Received April 21, 2020, accepted April 30, 2020, date of publication May 4, 2020, date of current version May 19, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2992340

An Efficient Single-Parameter Scaling Memoryless Broyden-Fletcher-Goldfarb-Shanno Algorithm for Solving Large Scale Unconstrained Optimization Problems

JING LV, SONGHAI DENG, AND ZHONG WAN
School of Mathematics and Statistics, Central South University, Changsha 410083, China

Corresponding author: Zhong Wan (wanmath@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 71671190 and Grant 51535012.

ABSTRACT In this paper, a new spectral scaling memoryless Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm is developed for solving large scale unconstrained optimization problems, where the scaling parameter is chosen so as to minimize all the eigenvalues of search direction matrices. The search directions in this algorithm are proved to satisfy the approximate Dai-Liao conjugate condition. With this advantage of the search directions, a scaling memoryless BFGS update formula is constructed and an algorithm is developed by incorporating acceleration strategy of line search and restart criterion. Under mild assumptions, global convergence of the algorithm is proved. Numerical tests demonstrate that the developed algorithm is more robust and efficient in solving large scale benchmark test problems than the similar ones in the literature.

INDEX TERMS Computational efficiency, convergence of numerical methods, optimization methods, algorithm design and analysis.

I. INTRODUCTION

Optimization models have found wider applications in the fields of engineering and management sciences [1], [2]. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) method is one of the most efficient quasi-Newton algorithms for solving medium scale unconstrained optimization models [3]. Owing to its good local and global convergence and self-correcting quality [4]–[6], it is particularly suitable for solving the small-sized and medium-sized unconstrained optimization problems [7], [8]. Specifically, a mathematical model of unconstrained optimization problems can be written as

$$\min f(x), x \in R^n, \tag{1}$$

where $f : R^n \to R$ is continuously differentiable. The BFGS method produces an iterate format for solving (1) by generating a sequence $\{x_k\}$, specified by

$$x_{k+1} = x_k + \alpha_k d_k, k = 0, 1, \ldots, \tag{2}$$

The associate editor coordinating the review of this manuscript and approving it for publication was Tachun Lin.

where

$$\begin{cases} d_k = -H_k g_k, \quad H_k = B_k^{-1}, \\ B_{k+1} = B_k - \dfrac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}, \; B_0 = I, \\ s_k = x_{k+1} - x_k, \\ y_k = \nabla f(x_{k+1}) - \nabla f(x_k) \triangleq g_{k+1} - g_k, \end{cases} \tag{3}$$

and α_k is a step size computed by a line search. It has been shown that $B_k \in R^{n\times n}$ in formula (3) is an approximation of the Hessian matrix $\nabla^2 f(x_k)$ for $k \geq 1$. Most importantly, if $y_k^T s_k > 0$, then for all k, $B_k \in R^{n\times n}$ in formula (3) is symmetric and positive definite. Therefore, d_k in (3) is a descent direction of the objective function f, as well as being close to the Newton direction. In general, the condition $y_k^T s_k > 0$ is guaranteed by the Wolfe-Powell line search.

However, some numerical experiences [7]–[9] have showed that the BFGS algorithm may be not efficient enough due to a poor approximation to the Hessian matrix at the initial point, or due to the ill-conditioning of the approximate Hessian matrices in the iterate process. To overcome the drawbacks of the BFGS algorithm, some modified versions have been proposed in the literature to improve its numerical efficiency and robustness [10]–[14]. In order to obtain a more effective algorithm, two aspects are generally studied. The first focuses on innovation of line search rules. In addition to the classical Wolfe line search and Armijo search rules, many effective line search rules have been proposed to find suitable step lengths in recent years [15], [16]. Another is focused on determination of efficient search directions. Exactly due to meticulous choices of step lengths and search directions, this type of algorithms are often more efficient than the heuristic algorithms [17]–[20]. Particularly, as shown in the existing results [7], [9], [13], [14], [21], the scaling factor in the update formula of BFGS plays an important role in approximating the Hessian matrix and improving efficiency of algorithms. In this paper, we mainly investigate how to determine search directions by this scaling strategy.

Note that the scaling strategy in the BFGS update formula has been regarded as one of the main approaches to avoid an ill-conditional B_k [9] in (3). It includes two ways: one is to multiply the approximate Hessian matrix by an appropriate scalar before it is updated in the BFGS method. Another is by properly scaling one or two terms in the BFGS update formula. Specifically, if the third term in the second equality in (3) is multiplied by a scaling parameter γ_k, then it is called one-parameter scaling BFGS update formula:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \gamma_k \frac{y_k y_k^T}{y_k^T s_k}. \tag{4}$$

In this case, the inverse of B_{k+1}, H_{k+1}, reads

$$H_{k+1} = H_k - \frac{H_k y_k s_k^T + s_k y_k^T H_k}{y_k^T s_k} + \left(\frac{1}{\gamma_k} + \frac{y_k^T H_k y_k}{y_k^T s_k}\right) \frac{s_k s_k^T}{y_k^T s_k}. \tag{5}$$

Clearly, if $\gamma_k = 1$, then (4) is exactly the standard BFGS update formula. One of our goals in this paper is to find a better scaling parameter such that the corresponding algorithm is more efficient and robust.

For many large-scale practical optimization models [22], [23], the (scaling) BFGS algorithms like (3) and (4) are often powerless because they are associated with solution of a large-scale system of linear equations $B_k d_k = -g_k$, as well as computation and storage of matrices B_k with large sizes. Therefore, another goal in this paper is to modify the scaling BFGS algorithms such that only gradient information in the optimization problem (1) is employed to compute the search directions and the step sizes without needs of computing and storing matrices [15]. In summary, this paper intends to study a single-parameter scaling memoryless BFGS algorithm for solving large scale unconstrained optimization problems.

The rest of this paper is organized as follows. In next section, we review the literature related to this study. Section III is devoted to development of a scaling memoryless BGFS algorithm based on acceleration scheme and restart criterion. In Section IV, global convergence of the algorithm is established. Numerical tests and discussion are conducted in Section V to show advantages of our algorithm. Along with suggestions for future research, some conclusions are drawn in the last section.

II. LITERATURE REVIEW

The BFGS algorithm was first proposed in 1970 by Broyden, Fletcher, Goldfarb and Shanno [24]. Owing to its fast convergence speed, a great number of its variants have been studied. Biggs [13] proposed a modified BFGS algorithm by introducing a scaling parameter such that an improved estimate of the second directional derivative is obtained. Yuan [14] presented another modified BFGS algorithm such that the updated approximate Hessian matrix satisfies the most recent quasi-Newton condition: the gradient value of the local quadratic model matches that of the objective function at the previous iterate. Cheng and Li [7] proposed a spectral scaling BFGS method by scaling the quasi-Newton equation, which has a self correcting property such that its numerical behavior is improved. However, global convergence was proved in [7] only for uniformly convex optimization problems.

To make the BFGS methods applicable to large scale optimization models, many memoryless BFGS updating formulas have been proposed. For more details, one can see [3], [25]–[32] and the references therein. For example, Livieris *et al.* [25] presented a new hybrid conjugate gradient method based on convex hybridization of the conjugate parameters of DY and HS+ by adapting the quasi-Newton philosophy. The computation of the hybrization parameter is obtained by minimizing the distance between the hybrid conjugate gradient direction and the self-scaling memoryless BFGS direction. Babaie-Kafaki and Ghanbari [26] proposed a nonlinear conjugate gradient method by minimizing the distance between the search direction matrix of the Dai-Liao method and the scaled memoryless BFGS update matrix in the Frobenius norm. Andrei [27] presented an accelerated scaled memoryless BFGS preconditioned conjugate gradient algorithm for solving unconstrained optimization problems by using the Powell's nonnegative restriction of the conjugate gradient. The basic idea was combination of the scaled memoryless BFGS method with the preconditioning technique in the frame of the conjugate gradient method. Andrei [28] and Yao and Ning [29] proposed two conjugate gradient methods, where the search direction was given by a symmetrical Perry matrix and was associated with a positive parameter determined by minimizing the distance of this matrix and the self-scaling memoryless BFGS matrix in the Frobenius norm. Apostolopoulou *et al.* [30] presented a curvilinear algorithmic model for training neural networks which was based on modifications of the memoryless BFGS method by incorporating a curvilinear search. In summary, the above-mentioned algorithms have global convergence and satisfactory numerical efficiency, but stability of the algorithms cannot be guaranteed. In this paper, we will develop a more stable algorithm by clustering all the eigenvalues of the approximate Hessian matrices to obtain a better scaling parameter in the memoryless BFGS update formula.

It is noted that Babaie-Kafaki [3], [31] started with controlling the condition number of the BFGS update matrix to develop more efficient algorithms. Actually, it was shown that the algorithms in [3], [31] are more stable compared with the similar ones. Very recently, Babaie-Kafaki and Ghanbari [32] further suggested a linear combination of the search direction by the memoryless BFGS technique with that by the Hestenes-Stiefel method. As a result, a one-parameter extension of the Hestenes-Stiefel method was proposed in [32]. However, unlike the analysis of the condition number of matrices in [3], [31], [32], we will use a measure function in this paper to conduct clustering analysis of all the eigenvalues of the approximate Hessian matrix, not only the maximum and minimum eigenvalues.

In summary, as a new scaling and memoryless BFGS algorithm, our algorithm has advantages of lower storage, lower computational cost and more stable numerical performance. Especially, we will prove that the used scaling parameter in our method can minimize all the eigenvalues of search direction matrices. Then, such an advantage of this parameter will be incorporated in developing an efficient algorithm for solving large scale optimization problems. We will also prove that our algorithm is globally convergent and show by numerical tests that it is more efficient and stable than the similar ones available in the literature.

III. DEVELOPMENT OF ALGORITHM

In this section, our aim is to develop a single-parameter scaling memoryless BFGS algorithm to solve large scale optimization problems.

In order to analyze properties of the BFGS methods, Byrd and Nocedal [33] introduced a measure function:

$$\varphi(A) = tr(A) - \ln(\det(A)), \tag{6}$$

where A is a symmetric positive definite search direction matrix in the quasi-Newton method, tr denotes the trace of this matrix, and det represents its determinant. From the definition of φ, we know that for a given matrix A, $\varphi(A)$ is involved with all the eigenvalues of A, not only the smallest or the largest ones. In this paper, we are concerned with how to choose the parameter γ_k in (4) such that $\varphi(B_{k+1})$ is minimized for any k. We first answer whether B_{k+1} in (4) is positive definite or not in the case that B_k is positive definite.

Proposition 1: Suppose that the stepsize α_k is computed by the Wolfe line search (16). If B_k is symmetric positive definite and $\gamma_k > 0$, then B_{k+1} in (4) is symmetric positive definite.

Proof: Similar to the proof of Proposition 2.1 in [8]. □

By Proposition 1, in order to minimize all the eigenvalues of B_{k+1}, we can choose a parameter $\gamma_k > 0$ such that

$$\gamma_k = \arg\min_{\gamma_k > 0} \varphi(B_{k+1}). \tag{7}$$

We can prove the following result.

Proposition 2: Denote

$$\gamma_k = \frac{y_k^T s_k}{\|y_k\|^2}. \tag{8}$$

Then, γ_k in (8) solves Problem (7).

Proof: Notice that

$$tr(B_{k+1}) = tr(B_k) - \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} + \gamma_k \frac{\|y_k\|^2}{y_k^T s_k}, \tag{9}$$

and

$$\det(B_{k+1}) = \gamma_k \frac{y_k^T s_k}{s_k^T B_k s_k} \det(B_k). \tag{10}$$

Consequently,

$$\begin{aligned}\varphi(B_{k+1}) &= tr(B_{k+1}) - \ln(\det(B_{k+1}))\\ &= tr(B_k) - \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} + \gamma_k \frac{\|y_k\|^2}{y_k^T s_k}\\ &\quad - \ln\left(\gamma_k \frac{y_k^T s_k}{s_k^T B_k s_k} \det(B_k)\right)\\ &= tr(B_k) - \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} + \gamma_k \frac{\|y_k\|^2}{y_k^T s_k}\\ &\quad - \ln\gamma_k - \ln\left(\frac{y_k^T s_k}{s_k^T B_k s_k}\right) - \ln(\det(B_k)).\end{aligned} \tag{11}$$

Therefore,

$$\frac{d\varphi}{d\gamma_k} = \frac{\|y_k\|^2}{y_k^T s_k} - \frac{1}{\gamma_k}. \tag{12}$$

The first-order optimality condition of Problem (7) yields

$$\frac{d\varphi}{d\gamma_k} = 0, \quad \gamma_k > 0.$$

From $\frac{d\varphi}{d\gamma_k} = 0$, it follows that

$$\frac{\|y_k\|^2}{y_k^T s_k} - \frac{1}{\gamma_k} = 0,$$

i.e.

$$\gamma_k = \frac{y_k^T s_k}{\|y_k\|^2}.$$

Clearly, $\gamma_k > 0$. The proof has been completed. □

Remark 1: Cheng and Li [7] also obtained the same scaling parameter as in (8) by minimizing $\|s_k - \gamma_k y_k\|^2$. Proposition 2 shows that such a γ_k also minimizes $\varphi(B_{k+1})$. Thus, its condition number is also minimized.

Clearly, a smaller condition number of search direction matrices can theoretically ensure stability of algorithms.

With the advantages of γ_k stated in Remark 1, we modify the scaling BFGS update formula (4) as:

$$B_{k+1} = I - \frac{s_k s_k^T}{s_k^T s_k} + \gamma_k \frac{y_k y_k^T}{y_k^T s_k}. \tag{13}$$

Compared with (4), (13), (4) is more applicable to solve large scale optimization problems since it no longer needs to solve a large scale system of linear equations to compute a search direction.

Specifically, let γ_k be defined by (8). We set $H_k = I$ in (5) and rewrite the inverse matrix of B_{k+1} as

$$H_{k+1} = I - \frac{y_k s_k^T + s_k y_k^T}{y_k^T s_k} + \left(\frac{1}{\gamma_k} + \frac{y_k^T y_k}{y_k^T s_k}\right)\frac{s_k s_k^T}{y_k^T s_k}. \quad (14)$$

Consequently, at the iterate point x_{k+1}, we obtain a search direction:

$$d_{k+1} = -g_{k+1} + \left(\frac{y_k^T g_{k+1}}{y_k^T s_k} - 2\frac{\|y_k\|^2}{y_k^T s_k}\frac{s_k^T g_{k+1}}{y_k^T s_k}\right)s_k + \frac{s_k^T g_{k+1}}{y_k^T s_k}y_k. \quad (15)$$

Since the Wolfe line search can ensure that H_{k+1} is positive definite, it is used to develop an algorithm together with the search direction being computed by (15). Specifically, at the k-th iteration, we choose a step size α_k such that it satisfies the following conditions:

$$\begin{cases} f(x_k + \alpha_k d_k) \le f(x_k) + \rho\alpha_k g(x_k)^T d_k \\ g(x_k + \alpha_k d_k)^T d_k \ge \sigma g(x_k)^T d_k. \end{cases} \quad (16)$$

Furthermore, if the second inequality in (16) is replaced by

$$| g(x_k + \alpha_k d_k)^T d_k | \le \sigma | g(x_k)^T d_k |, \quad (17)$$

then the step size α_k satisfies the strong Wolfe conditions [34].

With the above preparation, we are in a position to state the overall framework of our algorithm.

Algorithm 1 (Single-Parameter Scaling Memoryless BFGS Algorithm(SM-BFGS))

```
Step 0 (Initialization). Choose an initial point $x_0 \in R^n$ and an initial positive definite matrix $H_0$. Choose the constants $\sigma, \rho$ with $0 < \sigma < \rho < 1$ and $\varepsilon > 0$. Compute $g_0 = \nabla f(x_0)$, $d_0 = -g_0$. Set $k := 0$.
Step 1 (Termination). Test a criterion for stopping the iterations. If $\|g_k\| < \varepsilon$, then the algorithm stops; Otherwise, go to Step 2.
Step 2 (Line Search). Detemine a step size $\alpha_k > 0$, satisfying the Wolfe line search conditions (16), or the strong Wolfe condition (17).
Step 3 Compute $\bar{a}_k = \alpha_k g_k^T d_k$ and $\bar{b}_k = -\alpha_k y_k^T d_k$.
Step 4 (Acceleration). If $\bar{b}_k > 0$, then set $\xi := -\bar{a}_k/\bar{b}_k$ and update $x_{k+1} := x_k + \xi\alpha_k d_k$; Otherwise, update $x_{k+1} := x_k + \alpha_k d_k$. Compute $f_{k+1} = f(x_k)$ and $g_{k+1} = \nabla f(x_{k+1})$. Set $s_k := x_{k+1} - x_k$, $y_k := g_{k+1} - g_k$.
Step 5 (Search Direction). Compute the scaling parameter $\gamma_k$, $H_k$, and $d_{k+1}$ by (8), (14) and (15), respectively.
Step 6 (Powell restart criterion). If $|g_{k+1}^T g_k| > 0.2\|g_{k+1}\|^2$, then set $d_{k+1} := -g_{k+1}$.
Step 7 (Update). Set $k := k + 1$, and return to Step 1.
```

Remark 2: *Among the existing memoryless scaling methods, the scaling parameters, are obtained by the secant equation or by directly minimizing the condition number of the inverse Hessian matrix [3], [31], [35], [36]. In Algorithm 1, the scaling parameter is computed by (8), whose intrinsic features are given by Byrd and Nocedal's measure function. Since such a measure can achieve better eigenvalue clustering than the other methods, Algorithm 1 can be more stable than the similar ones.*

Remark 3: *Steps* 3 *and* 4 *are the same accelerating scheme as in [37].*

In the end of this section, we further prove that the search direction sequence $\{ d_{k+1} \}$ generated by Algorithm 1 has the following properties.

Proposition 3: *Suppose that the line search satisfies the Wolfe line search conditions (16). Then, d_{k+1} given by (15) is descent for any k.*

Proof: Since the step length α_k satisfies the Wolfe line search conditions, it follows that $y_k^T s_k > 0$. From (15), we have

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + 2\frac{y_k^T g_{k+1} \cdot s_k^T g_{k+1}}{y_k^T s_k} - 2\frac{\|y_k\|^2}{y_k^T s_k}\frac{(s_k^T g_{k+1})^2}{y_k^T s_k}.$$

Since for any $u, v \in R^n$, $u^T v \le \frac{1}{2}(\|u\|^2 + \|v\|^2)$, if we set

$$u = \frac{1}{\sqrt{2}}(y_k^T s_k)g_{k+1}, \quad v = \sqrt{2}\left(s_k^T g_{k+1}\right)y_k,$$

then

$$\begin{aligned} &\frac{y_k^T g_{k+1} \cdot s_k^T g_{k+1}}{y_k^T s_k} \\ &= \frac{y_k^T g_{k+1} \cdot y_k^T s_k \cdot s_k^T g_{k+1}}{(y_k^T s_k)^2} \\ &= \frac{\left(\frac{1}{\sqrt{2}}\left(y_k^T s_k\right) g_{k+1}\right)^T \left(\sqrt{2}(s_k^T g_{k+1})y_k\right)}{\left(y_k^T s_k\right)^2} \\ &\le \frac{\frac{1}{2}\left(\frac{1}{2}\left(y_k^T s_k\right)^2 \|g_{k+1}\|^2 + 2(s_k^T g_{k+1})^2\|y_k\|^2\right)}{(y_k^T s_k)^2} \\ &= \frac{1}{4}\|g_{k+1}\|^2 + \frac{(s_k^T g_{k+1})^2}{(y_k^T s_k)^2}\|y_k\|^2. \end{aligned}$$

Therefore,

$$\begin{aligned} g_{k+1}^T d_{k+1} &\le -\|g_{k+1}\|^2 - 2\frac{\|y_k\|^2}{y_k^T s_k}\frac{(s_k^T g_{k+1})^2}{y_k^T s_k} \\ &\quad + 2\left(\frac{1}{4}\|g_{k+1}\|^2 + \frac{(s_k^T g_{k+1})^2}{\left(y_k^T s_k\right)^2}\|y_k\|^2\right) \\ &\le -\frac{1}{2}\|g_{k+1}\|^2. \end{aligned}$$

We have proved that d_{k+1} is a sufficiently descent direction. □

Proposition 4: *Suppose that the line search satisfies the Wolfe line search condition (16). Then, d_{k+1} given by (15)* *satisfies an approximate Dai-Liao conjugate condition*:

$$y_k^T d_{k+1} = -t_k(s_k^T g_{k+1}). \tag{18}$$

Proof: Actually,

$$\begin{aligned} y_k^T d_{k+1} &= -y_k^T g_{k+1} - 2\frac{\|y_k\|^2}{y_k^T s_k}\frac{s_k^T g_{k+1}}{y_k^T s_k} \cdot y_k^T s_k \\ &\quad + \frac{y_k^T g_{k+1}}{y_k^T s_k} y_k^T s_k + \frac{s_k^T g_{k+1}}{y_k^T s_k} y_k^T y_k \\ &= -\frac{\|y_k\|^2}{y_k^T s_k} s_k^T g_{k+1} = -t_k(s_k^T g_{k+1}), \end{aligned}$$

where $t_k = \frac{\|y_k\|^2}{y_k^T s_k} > 0$. We have proved the result. □

The properties of Algorithm 1 in Propositions 3 and 4 are useful to its convergence analysis in next section.

IV. CONVERGENCE ANALYSIS

In this section, we will prove global convergence of Algorithm 1.

Since global convergence of the spectral scaling BFGS method was only proved for uniformly convex optimization problems in [7], we first simply prove that the scaling BFGS algorithm corresponding to the scaling parameter γ_k in (8) is globally convergent for any general smooth non-convex objective function, rather than a uniformly convex one. For readability, we state this spectral scaling BFGS algorithm as follows.

Algorithm 2 (Spectral Scaling BFGS Algorithm)

```
Step 0 (Initialization). Choose an initial point $x_0 \in R^n$ and an initial positive definite matrix $H_0$. Choose the constants $\sigma$, $\rho$ with $0 < \sigma < \rho < 1$ and stop tolerance $\varepsilon > 0$. Compute $g_0 = \nabla f(x_0)$. $d_0 = -g_0$. Set $k := 0$.
Step 1 (Termination). Test a criterion for stopping the iterations. If $\|g_k\| < \varepsilon$, then the algorithm stops. Otherwise, go to Step 2.
Step 2 (Line search). Compute a step size $\alpha_k > 0$ such that the Wolfe line search conditions (16) is satisfied.
Step 3. Compute the scaling parameter $\gamma_k$ using (8).
Step 4 (Update). Updating the inverse approximate Hessian $H_k$ by (5).
Step 5 (Search direction). Compute the search direction by $d_{k+1} = -H_{k+1} g_{k+1}$.
Step 6 (Update). Set $k := k + 1$. Go to Step 1.
```

As done in [7], we also need the following assumption.

Assumption 1: Assume that the level set $S = \{x : f(x) \leq f(x_0)\}$ is bounded, i.e., there exists a positive constant B such that for all $x \in S$, $\|x\| < B$.

Under Assumption 1, it follows from the first Wolfe condition (16) that the sequences $\{f(x_k)\}$ is not increasing. Thus,

$$\lim_{k\to\infty} f(x_k)$$

exists. Before statement of convergence result, we first prove the following results.

Lemma 1: Suppose that the inverse approximate Hessian matrix is computed by (5). Then, the search direction $d_{k+1} = -H_k g_{k+1}$ in Step 5 of Algorithm 2 is descent.

Proof: Left-multiplying g_{k+1}^T on both sides of $d_{k+1} = -H_k g_{k+1}$ yields

$$g_{k+1}^T d_{k+1} = -g_{k+1}^T H_k g_{k+1}.$$

By Proposition 1, B_k is positive definite. Since H_k is the inverse of B_k, H_k is positive definite. It is seen that d_k is a descent direction. □

Lemma 2: Suppose that the scaled B_{k+1} is determined by (4), where γ_k is computed by (8). Then,

$$tr(B_{k+1}) \leq tr(B_0) + (k+1), \tag{19}$$

and

$$\sum_{i=0}^{k} \frac{\|B_i s_i\|^2}{s_i^T B_i s_i} < tr(B_0) + (k+1). \tag{20}$$

Proof: From (9), we have

$$\begin{aligned} tr(B_{k+1}) &= tr(B_k) - \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} + \gamma_k \frac{\|y_k\|^2}{y_k^T s_k} \\ &= tr(B_0) - \sum_{i=0}^{k} \frac{\|B_i s_i\|^2}{s_i^T B_i s_i} + \sum_{i=0}^{k} \gamma_i \frac{\|y_i\|^2}{y_i^T s_i} \\ &= tr(B_0) - \sum_{i=0}^{k} \frac{\|B_i s_i\|^2}{s_i^T B_i s_i} + \sum_{i=0}^{k} \frac{y_i^T s_i}{\|y_i\|^2} \frac{\|y_i\|^2}{y_i^T s_i} \\ &= tr(B_0) - \sum_{i=0}^{k} \frac{\|B_i s_i\|^2}{s_i^T B_i s_i} + (k+1). \end{aligned}$$

Since B_{k+1} is positive definite and $tr(B_{k+1}) > 0$, we have

$$\sum_{i=0}^{k} \frac{\|B_i s_i\|^2}{s_i^T B_i s_i} < tr(B_0) + (k+1),$$

which ends the proof of the desired result. □

Remark 4: If $B_0 = I$, then $tr(B_{k+1}) \leq n + (k+1)$ and

$$\sum_{i=0}^{k} \frac{\|B_i s_i\|^2}{s_i^T B_i s_i} < n + (k+1).$$

Remark 5: Note that the inequality (19) reveals that the largest eigenvalue of B_{k+1} is strictly less than $tr(B_0)+(k+1)$. Therefore, the spectral scaling BFGS method with γ_k in (8) has a good self-correcting property subject to the trace. In other words, it can be more efficient than the other BFGS algorithms available in the literature by correcting the largest eigenvalue.

Lemma 3: If $\gamma_k \geq m$, for $k = 1, 2, \ldots$, where $m > 0$ is a constant, then there is a constant $c > 0$ such that for all k sufficiently large,

$$\prod_{i=0}^{k} \alpha_i \geq c^k. \tag{21}$$

Proof: Similar to the proof of Lemma 3.3 in [8].

Remark 6: If $B_0 = I$, then

$$\prod_{i=0}^{k} \alpha_i \geq \frac{m^{k+1}(1-\sigma)^{k+1}}{\left(\frac{1}{n}(n+k+1)\right)^n}.$$

With the results in Lemmas 1, 2 and 3, we can prove global convergence of Algorithm 2.

Theorem 1: Let $\{x_k\}$ be any sequence generated by Algorithm 2. Under Assumption 1, it holds that

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{22}$$

Proof: Assume that for all k, $\|g_k\| > \gamma > 0$. Note that f is bounded from below and $B_k s_k = \alpha_k B_k d_k = -\alpha_k g_k$. Thus, $\alpha_k = \frac{\|B_k s_k\|}{\|g_k\|}$. From the first inequality in the Wolfe conditions (16), it follows that

$$\sum_{k=0}^{\infty} \left(-s_k^T g_k\right) < \infty.$$

The following proof is similar to that of Theorem 3.1 in [38]. For completeness, we present this proof in detail.

$$\begin{aligned}
\infty &> \sum_{k=0}^{\infty} \left(-s_k^T g_k\right) \\
&= \sum_{k=0}^{\infty} \frac{1}{\alpha_k} s_k^T B_k s_k = \sum_{k=0}^{\infty} \frac{\|g_k\|}{\|B_k s_k\|} s_k^T B_k s_k \\
&= \sum_{k=0}^{\infty} \frac{s_k^T B_k s_k}{\|B_k s_k\|^2} \|g_k\| \|B_k s_k\| \\
&\geq \gamma^2 \sum_{k=0}^{\infty} \alpha_k \frac{s_k^T B_k s_k}{\|B_k s_k\|^2}.
\end{aligned}$$

By geometric arithmetic mean inequality, for any $\zeta > 0$, there exists an integer $k_0 > 0$ such that for any positive integer q, it holds that

$$q \left(\prod_{k=k_0+1}^{k_0+q} \alpha_k \frac{s_k^T B_k s_k}{\|B_k s_k\|^2} \right)^{1/q} \leq \sum_{k=k_0+1}^{k_0+q} \alpha_k \frac{s_k^T B_k s_k}{\|B_k s_k\|^2} \leq \zeta.$$

Thus,

$$\begin{aligned}
\left(\prod_{k=k_0+1}^{k_0+q} \alpha_k \right)^{1/q} &\leq \frac{\zeta}{q} \left(\prod_{k=k_0+1}^{k_0+q} \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} \right)^{1/q} \\
&\leq \frac{\zeta}{q^2} \sum_{k=k_0+1}^{k_0+q} \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} \\
&\leq \frac{\zeta}{q^2} \sum_{k=0}^{k_0+q} \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} \\
&\leq \frac{\zeta}{q^2} (tr(B_0) + (k_0+q+1)),
\end{aligned} \tag{23}$$

where the last inequality follows from Lemma 2. As $q \to \infty$, the last part of (23) converges to zero. Therefore,

$$\left(\prod_{k=k_0+1}^{k_0+q} \alpha_k \right)^{1/q} \leq 0,$$

which contradicts the result in Lemma 3. The proof of global convergence for a general non-convex problem is completed. □

We now come back to establish the global convergence of Algorithm 1, the core algorithm developed in this paper.

We first make the following mild assumption.

Assumption 2: In some neighborhood N of Ω, f is continuously differentiable and its gradient is Lipschitz continuous, namely, there exists a constant $L > 0$ such that

$$\|g(x) - g(y)\| \leq L\|x - y\|, \quad \forall x, y \in \Omega. \tag{24}$$

Under Assumptions 1 and 2, there exist constants $B > 0$ and $\Gamma \geq 0$ such that

$$\|s_k\| \leq B, \quad \|g(x)\| \leq \Gamma, \ \forall\, x \in \Omega.$$

Lemma 4: Under Assumptions 1 and 2, if the line search satisfies the Wolfe conditions (16), then for all $k > 0$, the following inequality holds:

$$\alpha_k \geq \frac{(1-\sigma)|g_k^T d_k|}{L\|d_k\|^2}. \tag{25}$$

Proof: From the Wolfe conditions (16), it follows that

$$(\sigma - 1) g_k^T d_k \leq (g_{k+1} - g_k)^T d_k \leq \alpha_k L \|d_k\|^2.$$

Since d_k is a descent direction and $\sigma < 1$, the inequality (25) has been proved. □

Lemma 5: Let $\{d_k\}$ be any sequence generated by Algorithm 1. Under Assumptions 1 and 2, it holds that

$$\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < +\infty. \tag{26}$$

Proof: From the Wolfe condition (16) and Lemma 4, we get

$$f(x_k) - f(x_{k+1}) \geq -\rho \alpha_k g_k^T d_k \geq \rho \frac{(1-\sigma)(g_k^T d_k)^2}{L\|d_k\|^2}.$$

Therefore, from Assumption 2, we get the Zoutendijk condition (26) [39]. □

Lemma 6: Let $\{d_k\}$ be any sequence generated by Algorithm 1. Under Assumptions 1 and 2, if

$$\sum_{k\geq 0} \frac{1}{\|d_k\|^2} = \infty, \tag{27}$$

then

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{28}$$

Proof: Similar to the proof of Lemma 3.1 in [40], we can prove Lemma 6. □

Theorem 2: Let $\{x_k\}$ be any sequence generated by Algorithm 1. Under Assumptions 1 and 2, if there exists $\tau > 0$ such that $y_k^T s_k > \tau$, then

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{29}$$

Proof: By Assumptions 1 and 2, we have

$$\|y_k\| = \|g_{k+1} - g_k\| \leq L\|s_k\| \leq BL. \tag{30}$$

Suppose that $g_k \neq 0$ for all $k \geq 1$. Otherwise, a stationary point is obtained. From Assumptions 1 and 2, it follows that

$$\begin{aligned}\|d_{k+1}\| &\leq \|g_{k+1}\| + \frac{|y_k^T g_{k+1}|}{|y_k^T s_k|}\|s_k\| \\ &\quad + 2\frac{\|y_k\|^2}{|y_k^T s_k|}\frac{|s_k^T g_{k+1}|}{|y_k^T s_k}|\|s_k\| + \frac{\|s_k^T g_{k+1}|}{|y_k^T s_k|}\|y_k\| \\ &\leq \|g_{k+1}\| + \frac{\|y_k\|\|g_{k+1}\|}{\tau}\|s_k\| \\ &\quad + 2\frac{\|y_k\|^2}{\tau}\frac{\|s_k\|^2\|g_{k+1}\|}{\tau} + \frac{\|s_k\|\|g_{k+1}\|\|y_k\|}{\tau} \\ &\leq \Gamma + 2\frac{L\Gamma B^2}{\tau} + 2\frac{L^2\Gamma B^4}{\tau^2} = M. \end{aligned} \tag{31}$$

Consequently, the condition (27) is true. By Lemma 6, we know that (29) is true and the global convergence is proved. □

V. NUMERICAL TESTS AND DISCUSSION

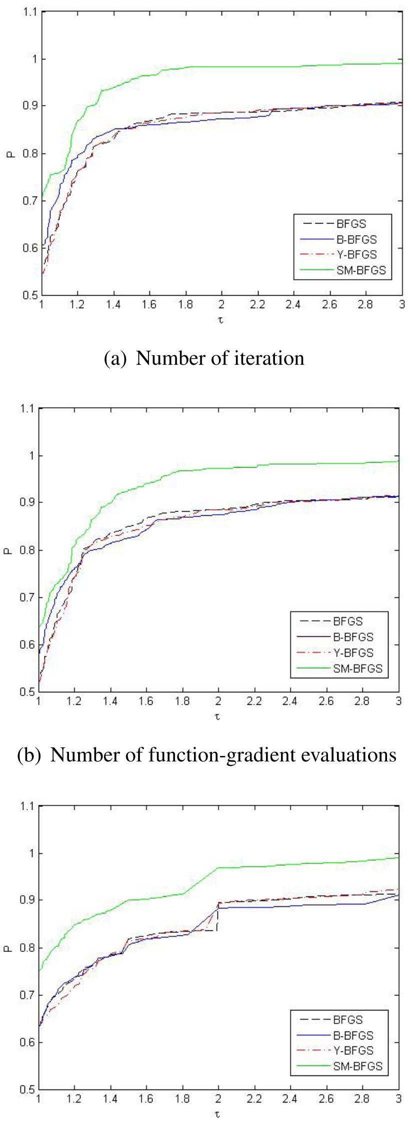
In this section, we report the numerical performance of Algorithm 1 (SM-BFGS), in comparison with similar algorithms available in the literature.

We test all these algorithms by using them to solve 750 large-scale test problems from [41], where the dimension of each benchmark problem changes from 1000 to 10000 with a step length 1000. To further validate global convergence of the algorithms, the same initial point, as given in the literature, is used. To show the advantages of our search directions in Algorithm 1, the step length in all the tested algorithms are determined by the Wolfe line search for a fair comparison, where we take $\rho = 0.0001$ and $\sigma = 0.8$. Each algorithm stops if the condition $\|g_k\| \leq 10^{-6}$ is satisfied or if the number of iterations exceeds 10^4. It is noted that in Algorithm 1, a strategy of the line search acceleration is used to improve its numerical performance.

All the numerical results are presented by a frequently-used approach to comparison of algorithm's performance [42]. Specifically, let f_i^A and f_i^B be the optimal value found by Algorithms A and B for the i-th test problem $(i = 1, 2, \ldots, p)$, respectively. For the i-th test problem, we say the performance of Algorithm A is better than that of Algorithm B if:

$$|f_i^A - f_i^B| < 10^{-3},$$

and the number of iterations, or the number of function-gradient evaluations, or the CPU time of Algorithm A is less than that of Algorithm B.



(a) Number of iteration

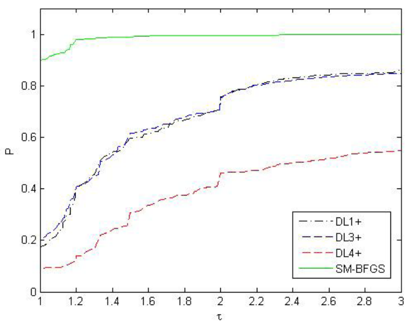
(b) Number of function-gradient evaluations

(c) Consumed CPU time

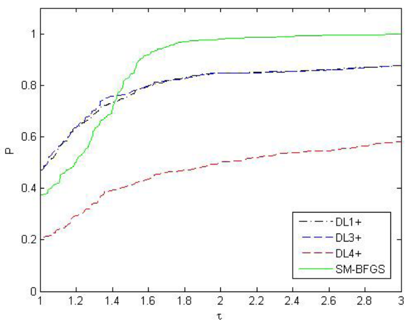
FIGURE 1. Comparison of similar BFGS-type algorithms.

To intuitively display numerical performance of all the tested algorithms, we use the Dolan and Moré performance profile graphs to analyze their numerical results, including the number of iterations, the total number of evaluating the objective function and its gradient, and the consumed CPU time when each algorithm stops. Specifically, in these performance profile graphs (see, for example, Figures 1 and 2), the horizontal axis represents the performance analysis factor τ, which can reflect efficiency of each algorithm in solving all the 750 test problems. The vertical axis is the probability P of each algorithm (measured by the proportion of the test problems) that a certain performance ratio is within a factor $\tau \in R$ of the best possible ratio.

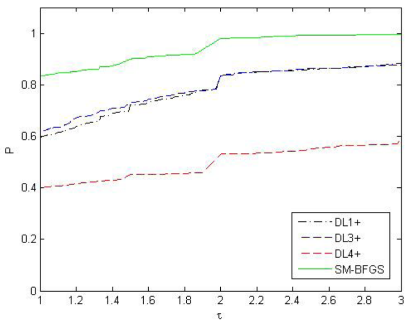
All codes of the computer procedures are written in Fortran 90, and are implemented in Windows system with 2.4 GHz CPU processor, 4 GB RAM memory.



(a) Number of iteration



(b) Number of function-gradient evaluations



(c) Consumed CPU time

FIGURE 2. **Comparison of similar conjugate gradient algorithms.**

In the first round of tests, since Algorithm 1 can be regarded as an extended BFGS algorithm, we compare our algorithm with the standard BFGS, B-BFGS in [13] and Y-BFGS in [14] with the memoryless technique. For readability, we present the scaling parameters in [13], [14] as follows.

$$\begin{aligned}\gamma_k^B &= \frac{6}{y_k^T s_k}(f(x_k) - f(x_{k+1}) + s_k^T g_{k+1}) - 2,\\ \gamma_k^Y &= \frac{2}{y_k^T s_k}(f(x_k) - f(x_{k+1}) + s_k^T g_{k+1}).\end{aligned} \tag{32}$$

In all the four algorithms, the initial matrix $H_0 = I$ for solving any test problem. In Figure 1, we show the efficiency comparison among SM-BFGS and the other three types of BFGS algorithms.

TABLE 1. **Pairwise comparison between Algorithm 1 and another algorithm.**

Other algorithms		SM-BFGS is better	Both are the same	Another algorithm is better
BFGS	NI	274	294	167
	NE	288	214	233
	CT	259	305	171
B-BFGS	NI	214	239	182
	NE	230	195	210
	CT	239	235	161
Y-BFGS	NI	295	268	173
	NE	307	215	214
	CT	267	305	164
DL1+	NI	579	93	47
	NE	299	17	403
	CT	333	246	140
DL3+	NI	565	91	59
	NE	295	18	402
	CT	320	249	146
DL4+	NI	617	48	14
	NE	456	28	195
	CT	436	182	61

From the numerical results in Figure 1, it is clear that:

(1) In terms of the number of iteration shown in Figure 1(a), our algorithm SM-BFGS can solve about 71% of the test problems with the least number of iterations. In contrast, BFGS, B-BFGS and Y-BFGS only solve about 56%, 60% and 55% of these problems, respectively.

(2) In terms of the number of function-gradient evaluations shown in Figure 1(b), SM-BFGS performs the best in solving about 63% of the test problems, and for the other three algorithms, the proportions of the test problems with the least number of evaluating functions and gradients are 53%, 52% and 58%, respectively.

(3) With respect to the consumed CPU time shown in Figure 1(c), our algorithm spends the shortest CPU time in solving about 75% of the problems, while the other three algorithms have a shorter running time on about 64% of these problems.

In one word, our algorithm (SM-BFGS) can solve the test problems as many as possible within less number of iteration, less number of function-gradient evaluations and less CPU time. Therefore, it is concluded that SM-BFGS outperforms the similar three types of BFGS algorithms.

In the second round of tests, since Algorithm 1 can be regarded as a modified conjugate gradient algorithm, we compare Algorithm 1 (SM-BFGS) with the similar conjugate gradient algorithms available in the literature. Particularly, recall that in Proposition 4, we have proved that the directions in our algorithm also satisfy the approximate Dai-Liao conjugate condition (18). In the following, SM-BFGS is compared with other three Dai-Liao types of conjugate gradient algorithms, which were recently published in [26], [43], [44], respectively. For simplicity, we denote them DL1+ [43], DL3+ [44] and DL4+ [26], respectively. Efficiency comparison of the four algorithms is presented in Figure 2.

From the numerical results in Figure 2, it is easy to see that:

(1) With regard to the number of iterations shown in Figure 2(a), Algorithm 1 (SM-BFGS) can solve about 90%

TABLE 2. Advantages of OM-BFGS compared with other six algorithms.

Function name	SM-BFGS	BFGS	Y-BFGS	B-BFGS	DL1+	DL3+	DL4+
Extended	NI: 29	NI: 32	NI: 32	NI: 32	NI: 37	NI: 39	NI: 188
Rosenbrock	NE: 97	NE: 104	NE: 104	NE:104	NE: 81	NE: 91	NE: 429
DIM:20000	CT: 7	CT:7	CT: 8	CT: 8	CT:8	CT: 7	CT: 34
Extended	NI: 29	NI: 32	NI: 32	NI: 32	NI: 37	NI: 38	NI: 242
Rosenbrock	NE: 97	NE: 104	NE: 104	NE:104	NE: 80	NE: 90	NE: 483
DIM:25000	CT: 8	CT: 9	CT: 12	CT: 8	CT: 11	CT:8	CT: 52
Extended	NI: 30	NI: 32	NI: 32	NI: 32	NI: 36	NI:38	NI: 151
Rosenbrock	NE: 100	NE: 104	NE: 104	NE:104	NE: 79	NE:90	NE: 211
DIM:30000	CT: 12	CT: 10	CT: 10	CT: 11	CT: 11	CT: 10	CT: 33
Raydan 1	NI: 793	NI: 801	NI: 801	NI: 801	NI: 4918	NI: 1567	NI: 10001
DIM:15000	NE: 1630	NE: 1646	NE: 1646	NE: 1646	NE: 5633	NE: 2449	NE: 12451
	CT: 149	CT: 163	CT: 150	CT:153	CT:668	CT: 249	CT: 1393
Raydan 1	NI: 916	NI: 925	NI:925	NI: 925	NI: 5625	NI:4407	NI: 10001
DIM:20000	NE: 1878	NE: 1895	NE:1894	NE:1895	NE: 6991	NE: 5612	NE:12196
	CT: 235	CT: 239	CT: 229	CT: 242	CT: 1072	CT: 845	CT:1847
Hager	NI: 98	NI: 99	NI: 108	NI:101	NI: 1131	NI: 1147	NI: 1150
DIM:20000	NE: 633	NE: 653	NE: 784	NE: 706	NE:15018	NE:15003	NE: 15003
	CT: 67	CT: 70	CT: 82	CT: 75	CT: 1532	CT: 1498	CT: 1493
Generalized	NI: 239	NI: 256	NI: 340	NI: 257	NI: 542	NI:724	NI: 1730
PSC1	NE:706	NE: 685	NE: 1202	NE:743	NE: 1050	NE: 2521	NE: 3900
DIM:15000	CT: 98	CT: 84	CT: 140	CT: 92	CT: 142	CT: 312	CT: 500
Generalized	NI: 223	NI: 273	NI: 426	NI: 642	NI:437	NI: 730	NI: 1434
PSC1	NE: 688	NE: 1137	NE: 1583	NE: 1665	NE: 845	NE: 2179	NE: 2676
DIM:30000	CT: 164	CT: 263	CT: 396	CT: 429	CT:227	CT: 562	CT: 734
Extended	NI: 37	NI: 41	NI: 56	NI: 652	NI: 66	NI:64	NI: 186
Powell	NE: 104	NE: 117	NE: 161	NE: 1818	NE: 121	NE: 120	NE: 398
DIM:15000	CT: 7	CT: 8	CT: 11	CT: 121	CT: 10	CT:9	CT: 31
Extended	NI: 45	NI: 51	NI: 53	NI: 57	NI: 62	NI: 78	NI: 10001
Powell	NE: 132	NE: 143	NE: 154	NE:163	NE: 116	NE: 151	NE: 10637
DIM:30000	CT: 17	CT: 20	CT: 20	CT: 22	CT: 19	CT:24	CT: 2512
Extended	NI: 31	NI: 35	NI: 36	NI: 37	NI: 43	NI: 42	NI: 42
QP2	NE: 112	NE: 120	NE: 124	NE:127	NE: 110	NE: 104	NE: 102
DIM:15000	CT: 14	CT: 15	CT: 15	CT: 16	CT:14	CT: 13	CT: 14
Extended	NI: 33	NI: 38	NI:39	NI: 39	NI: 43	NI: 42	NI: 47
QP2	NE: 114	NE: 131	NE: 136	NE: 139	NE:101	NE:98	NE: 114
DIM:20000	CT: 19	CT: 22	CT: 22	CT: 23	CT: 18	CT: 17	CT: 20

of the test problems with the least number of iterations, while DL1+, DL3+ and DL4+ solve only about 18%, 20% and 10% of these problems, respectively.

(2) With regard to the number of function-gradient evaluation shown in Figure 2(b), the algorithms SM-BFGS, DL1+, DL3+, and DL4+ respectively solve about 37%, 47%, 21%, and 47% of the test problems with the least number of evaluating the objective function and its gradient.

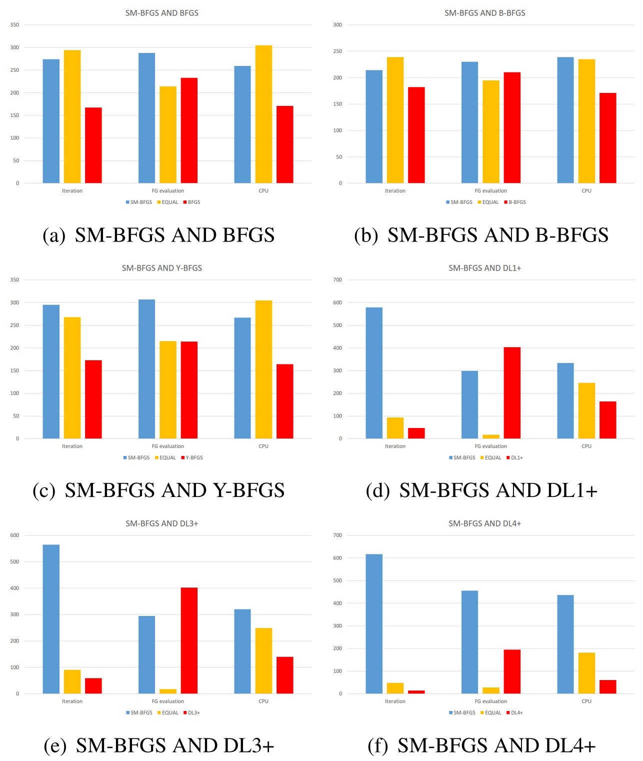
(3) In terms of performance profile of the consumed CPU time shown in Figures 2(c), SM-BFGS solves about 83% of the test problems with the least CPU time, while DL1+, DL3+ and DL4+ can solve about 60%, 62% and 40% of these problems with the least CPU time, respectively.

In summary, Figures 2(a) and 2(c) indicate that Algorithm 1 (SM-BFGS) can solves the test problems as many as possible with less number of iterations and less CPU time than the other three algorithm.

In order to further justify the advantages of Algorithm 1, we make pairwise comparison between Algorithm 1 and any one of the other six algorithms. The pairwise comparison results are presented in Table 1, where we denote NI, NE and CT the number of iteration, the number of evaluating the objective function and its gradient and the consumed CPU time after termination, respectively. More intuitively, Figure 3 displays the differences of numerical performance in Table 1 between Algorithm 1 and another algorithm. In Figure 3, each figure contains three groups of bars, which represent the three types of numerical performances: the number of iterations, the number of function-gradient evaluations and the consumed CPU time. Each group consists of the three bars with different colors: the blue bar represents the number of the test problems solved by Algorithm 1 with better numerical performance, the red bar represents the number of these problems solved by another algorithm with better numerical performance, and the orange bar represents the number of the test problems when the numerical performance of the compared algorithms is the same.

Figure 3 clearly demonstrates that in the pairwise comparison, Algorithm 1 (SM-BFGS) is better than any one of the other six algorithms. For example, between SM-BFGS and DL4+, it can be seen that:

(1) In terms of the number of iterations, SM-BFGS achieves less number of iterations for the 617 problems, DL4+ is better than SM-BFGS for the 14 problems, and for the rest of 48 problems, SM-BFGS and DL4+ have the same number of iterations.



(a) SM-BFGS AND BFGS (b) SM-BFGS AND B-BFGS

(c) SM-BFGS AND Y-BFGS (d) SM-BFGS AND DL1+

(e) SM-BFGS AND DL3+ (f) SM-BFGS AND DL4+

FIGURE 3. Comparison results between SM-BFGS and other six algorithms.

(2) In terms of the number of evaluating the objective function and its gradient, SM-BFGS wins for the 456 problems, DL4+ performs better for the 195 problems, and for the rest of 28 problems, SM-BFGS and DL4+ have the same numerical performance.

(3) With regard to the consumed CPU time, SM-BFGS runs faster than DL4+ for the 436 problems, DL4+ is faster only for the 61 problems, and for the rest of 182 problems, they perform the same.

On the whole, the second round of tests also shows Algorithm 1 is more stable and more efficient in solving the large scale benchmark test problems.

In the last round of tests, we report the numerical results in Table 2 as all the seven algorithms are used to solve the benchmark test problems with dimension of over 10000 (DIM).

The underlined results in Table 2 are the best ones among the seven algorithms. From the viewpoint of less number of iterations, less number of evaluating the objective function and its gradient, or less consumed CPU time after termination, our algorithm (SM-BFGS) also outperforms all the other six ones as they are used to solve large scale optimization problems with dimension of over 10000.

VI. CONCLUSION AND FUTURE RESEARCH

In this paper, we have proposed a new scaling memoryless BFGS algorithm for solving large scale unconstrained optimization problems. We have proved that the used scaling parameter can minimize all the eigenvalues of search direction matrices and the corresponding search directions satisfy the approximate Dai-Liao conjugate condition. In addition, a strategy of the line search acceleration is employed to improve numerical performance of this algorithm. Under mild assumptions, we have proved that the developed algorithm is globally convergent.

By numerical tests, we have demonstrated that our algorithm outperforms the similar ones available in the literature for solving large scale optimization problems, either as an extension of the BFGS-type algorithms or as a modified conjugate gradient algorithm.

In future research, it is valuable to study new two-parameter scaling memoryless BFGS algorithms such that the used scaling parameters can minimize all the eigenvalues of search direction matrices.

It is also interesting to further validate the proposed method in this paper by applying it in solving more practical optimization models from the fields of medical, engineering and management. For example, as done in [23], the developed algorithm may be useful to deeply mine the transcriptomic profile of the sub-genomes in hybrid fish lineage.

Since a nonlinear system of equations is closely related with an optimization model, it is significant to extend the developed algorithm into solving large scale nonlinear system of equations from engineering fields. Actually, it has been shown [20], [22] that recovering sparse signals and restoring blurred images can be formulated as a system of equations, and efficient optimization algorithms can be modified to solve these practical engineering problems.

REFERENCES

[1] Z. Wang, G. He, W. Du, J. Zhou, X. Han, J. Wang, H. He, X. Guo, J. Wang, and Y. Kou, "Application of parameter optimized variational mode decomposition method in fault diagnosis of gearbox," *IEEE Access*, vol. 7, pp. 44871–44882, 2019.

[2] D. Dabhi and K. Pandya, "Enhanced velocity differential evolutionary particle swarm optimization for optimal scheduling of a distributed energy resources with uncertain scenarios," *IEEE Access*, vol. 8, pp. 27001–27017, 2020.

[3] S. Babaie-Kafaki and Z. Aminifard, "Two–parameter scaled memoryless BFGS methods with a nonmonotone choice for the initial step length," *Numer. Algorithms*, vol. 82, no. 4, pp. 1345–1357, Dec. 2019.

[4] A. Griewank, "The global convergence of partitioned BFGS on problems with convex decompositions and lipschitzian gradients," *Math. Program.*, vol. 50, nos. 1–3, pp. 141–175, Mar. 1991.

[5] M. J. D. Powell, "On the convergence of the variable metric algorithm," *IMA J. Appl. Math.*, vol. 7, no. 1, pp. 21–36, 1971.

[6] J. Nocedal, "Theory of algorithms for unconstrained optimization," *Acta Numerica*, vol. 1, pp. 199–242, Jan. 1992.

[7] W. Y. Cheng and D. H. Li, "Spectral scaling BFGS method," *J. Optim. Theory Appl.*, vol. 146, no. 2, pp. 305–319, Aug. 2010.

[8] N. Andrei, "An adaptive scaled BFGS method for unconstrained optimization," *Numer. Algorithms*, vol. 77, no. 2, pp. 413–432, Feb. 2018.

[9] N. Andrei, "A double–parameter scaling Broyden–Fletcher–Goldfarb–Shanno method based on minimizing the measure function of Byrd and Nocedal for unconstrained optimization," *J. Optim. Theory Appl.*, vol. 178, no. 1, pp. 191–218, Jul. 2018.

[10] Z. Wan, K. L. Teo, X. Shen, and C. Hu, "New BFGS method for unconstrained optimization problem based on modified armijo line search," *Optimization*, vol. 63, no. 2, pp. 285–304, Feb. 2014.

[11] Y. Ou and X. Zhou, "A nonmonotone scaled conjugate gradient algorithm for large-scale unconstrained optimization," *Int. J. Comput. Math.*, vol. 95, no. 11, pp. 2212–2228, Nov. 2018.

[12] A. Ebrahimi and G. Barid Loghmani, "Shape modeling based on specifying the initial B-spline curve and scaled BFGS optimization method," *Multimedia Tools Appl.*, vol. 77, no. 23, pp. 30331–30351, Dec. 2018.
[13] M. C. Biggs, "Minimization algorithms making use of non-quadratic properties of the objective function," *IMA J. Appl. Math.*, vol. 9, no. 2, pp. 123–123, 1972.
[14] Y.-X. Yuan, "A modified BFGS algorithm for unconstrained optimization," *IMA J. Numer. Anal.*, vol. 11, no. 3, pp. 325–332, 1991.
[15] T. Li and Z. Wan, "New adaptive Barzilar-Borwein step size and its application in solving large scale optimization problems," *ANZIAM J.*, vol. 61, no. 1, pp. 76–98, 2019.
[16] S. Huang, Z. Wan, and J. Zhang, "An extended nonmonotone line search technique for large-scale unconstrained optimization," *J. Comput. Appl. Math.*, vol. 330, pp. 586–604, Mar. 2018.
[17] Q. Al-Tashi, S. J. Abdul Kadir, H. M. Rais, S. Mirjalili, and H. Alhussian, "Binary optimization using hybrid grey wolf optimization for feature selection," *IEEE Access*, vol. 7, pp. 39496–39508, 2019.
[18] Q. Liu, X. Shen, and Y. Gu, "Linearized ADMM for nonconvex nonsmooth optimization with convergence analysis," *IEEE Access*, vol. 7, pp. 76131–76144, 2019.
[19] S. H. Deng, J. Lv, and Z. Wan "A new Dai–Liao type of conjugate gradient algorithm for unconstrained optimization problems," *Pacific J. Optim.*, vol. 15, no. 2, pp. 237–248, 2019.
[20] Z. Wan, J. Guo, J. Liu, and W. Liu, "A modified spectral conjugate gradient projection method for signal recovery," *Signal, Image Video Process.*, vol. 12, no. 8, pp. 1455–1462, Nov. 2018.
[21] I. E. Livieris and P. Pintelas, "An improved weight-constrained neural network training algorithm," *Neural Comput. Appl.*, vol. 32, no. 9, pp. 4177–4185, May 2020.
[22] J. Guo and Z. Wan, "A modified spectral PRP conjugate gradient projection method for solving large-scale monotone equations and its application in compressed sensing," *Math. Problems Eng.*, vol. 2019, Apr. 2019, Art. no. 5261830, doi: 10.1155/2019/5261830.
[23] Z. Wan and J. Y. Tang "Optimization techniques to deeply mine the transcriptomic profile of the sub-genomes in hybrid fish lineage," *Frontiers Genet.*, vol. 10, p. 911, Oct. 2019.
[24] C. G. Broyden, "The convergence of a class of double-rank minimization algorithms," *J. Inst. Math. Appl.*, vol. 6, no. 1, pp. 76–90, 1970.
[25] I. E. Livieris, V. Tampakas, and P. Pintelas, "A descent hybrid conjugate gradient method based on the memoryless BFGS update," *Numer. Algorithms*, vol. 79, no. 4, pp. 1169–1185, Dec. 2018.
[26] S. Babaie-Kafaki and R. Ghanbari, "A class of adaptive Dai–Liao conjugate gradient methods based on the scaled memoryless BFGS update," *4OR*, vol. 15, no. 1, pp. 85–92, Mar. 2017.
[27] N. Andrei, "Accelerated scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization," *Eur. J. Oper. Res.*, vol. 204, no. 3, pp. 410–420, Aug. 2010.
[28] N. Andrei, "Accelerated adaptive perry conjugate gradient algorithms based on the self-scaling memoryless BFGS update," *J. Comput. Appl. Math.*, vol. 325, pp. 149–164, Dec. 2017.
[29] S. Yao and L. Ning, "An adaptive three-term conjugate gradient method based on self-scaling memoryless BFGS matrix," *J. Comput. Appl. Math.*, vol. 332, pp. 72–85, Apr. 2018.
[30] M. S. Apostolopoulou, D. G. Sotiropoulos, I. E. Livieris, and P. Pintelas, "A memoryless BFGS neural network training algorithm," in *Proc. 7th IEEE Int. Conf. Ind. Informat.*, Jun. 2009, pp. 216–221.
[31] S. Babaie-Kafaki, "A modified scaling parameter for the memoryless BFGS updating formula," *Numer. Algorithms*, vol. 72, no. 2, pp. 425–433, Jun. 2016.
[32] S. Babaie-Kafaki and R. Ghanbari, "A linear hybridization of the Hestenes–Stiefel method and the memoryless BFGS technique," *Medit. J. Math.*, vol. 15, no. 3, p. 86, Jun. 2018.
[33] R. H. Byrd and J. Nocedal, "A tool for the analysis of quasi-Newton methods with application to unconstrained minimization," *SIAM J. Numer. Anal.*, vol. 26, no. 3, pp. 727–739, Jun. 1989.
[34] P. Wolfe, "Convergence conditions for ascent methods. II: Some corrections," *SIAM Rev.*, vol. 13, no. 2, pp. 185–188, Apr. 1971.
[35] S. Babaie-Kafaki, "On optimality of the parameters of self-scaling memoryless quasi-Newton updating formulae," *J. Optim. Theory Appl.*, vol. 167, no. 1, pp. 91–101, Oct. 2015.
[36] H. Yabe, H. J. Martinez, and R. A. Tapia, "On sizing and shifting the BFGS update within the sized-broyden family of secant updates," *SIAM J. Optim.*, vol. 15, no. 1, pp. 139–160, Jan. 2004.
[37] N. Andrei, "Acceleration of conjugate gradient algorithms for unconstrained optimization," *Appl. Math. Comput.*, vol. 213, no. 2, pp. 361–369, Jul. 2009.
[38] D.-H. Li and M. Fukushima, "A modified BFGS method and its global convergence in nonconvex minimization," *J. Comput. Appl. Math.*, vol. 129, nos. 1–2, pp. 15–35, Apr. 2001.
[39] G. Zoutendijk, "Nonlinear programming, computational methods," in *Integer and Nonlinear Programming*, J. Abadie, Ed. Amsterdam, The Netherlands: North-Holland, 1970, pp. 37–86.
[40] K. Sugiki, Y. Narushima, and H. Yabe, "Globally convergent three-term conjugate gradient methods that use secant conditions and generate descent search directions for unconstrained optimization," *J. Optim. Theory Appl.*, vol. 153, no. 3, pp. 733–757, Jun. 2012.
[41] N. Andrei, "An unconstrained optimization test functions collection," *Adv. Model. Optim.*, vol. 10, no. 1, pp. 147–161, 2008.
[42] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Math. Program.*, vol. 91, no. 2, pp. 201–213, Jan. 2002.
[43] S. Babaie-Kafaki and R. Ghanbari, "The Dai–Liao nonlinear conjugate gradient method with optimal parameter choices," *Eur. J. Oper. Res.*, vol. 234, no. 3, pp. 625–630, May 2014.
[44] S. Babaie-Kafaki and R. Ghanbari, "Two optimal Dai–Liao conjugate gradient methods," *Optimization*, vol. 64, no. 11, pp. 2277–2287, Nov. 2015.

JING LV received the B.S. degree from the School of Mathematics and Statistics, Shandong University of Technology, Zibo, China, in 2017. She is currently pursuing the master's degree with the School of Mathematics and Statistics, Central South University (CSU), Changsha, China. Her research interest is in large-scale optimization algorithms.

SONGHAI DENG received the B.S. degree in mathematics from Hunan Normal University, Changsha, China, in 1991, the M.S. degree in mathematical education from Hunan Normal University, Changsha, in 1994, and the Ph.D. degree in applied mathematics from Central South University, Changsha, in 2013. Since 2011, he has been an Associate Professor with the School of Mathematics and Statistics, Central South University. His research interests are in efficient optimization algorithms in industrial engineering and mathematical modeling.

ZHONG WAN received the B.S. degree in mathematics from Jiangxi Normal University, Nanchang, China, in 1988, the M.S. degree in mathematical education from Hunan Normal University, Changsha, China, in 1993, and the Ph.D. degree in applied mathematics from Hunan University, Changsha, in 2001. Since 2004, he has been a Chair Professor of operational research and control with the School of Mathematics and Statistics, Central South University, Changsha. His research interests include the development of efficient algorithms for solving optimization models and equilibrium problems and mathematical modeling in the fields of engineering and management sciences. He received several honors, including the Excellent Talent of New Century, Ministry of Education, China, and the Excellent Young Teacher of Hunan province, China.