manuscript No.
(will be inserted by the editor)

A Limited Memory Subspace Minimization Conjugate Gradient Algorithm for Unconstrained Optimization

Zexian Liu[1,2] · Yu-Hong Dai[2,*] · Hongwei Liu[3]

the date of receipt and acceptance should be inserted later

Abstract Subspace minimization conjugate gradient (SMCG) methods are a class of high potential iterative methods for unconstrained optimization. The orthogonality is an important property of linear conjugate gradient method. It is however observed that the orthogonality of gradients in linear conjugate gradient method is often lost, which usually causes the slow convergence of conjugate gradient method. Based on SMCG_BB (J. Optim. Theory Appl. 180(3), 879-906, 2019), we combine the limited memory technique with subspace minimization conjugate gradient method and present a limited memory subspace minimization conjugate gradient algorithm for unconstrained optimization in this paper. The proposed method includes two types of iterations: SMCG iteration and quasi-Newton (QN) iteration. In the SMCG iteration, we determine the search direction by solving the quadratic approximation problem, in which the important parameter is estimated based on some properties of the objective function at the current iterative point. In the QN iteration, a modified quasi-Newton method in the subspace is exploited to improved the orthogonality. Additionally, a modified strategy for choosing the initial stepsize is exploited. The global convergence of the proposed method is established under weaker conditions. Some numerical results indicate that,for the CUTEr library, the proposed method has a great improvement over SMCG_BB, and is comparable to the latest limited memory conjugate gradient software package CG_DESCENT (6.8) (SIAM J. Optim. 23(4), 2150-2168, 2013) and is superior to the limited memory BFGS (L-BFGS) method.

Keywords Limited memory · Barzilai-Borwein conjugate gradient method · Subspace minimization · Orthogonality · Quasi-Newton method

Mathematics Subject Classification (2000) 49M37 · 65K05 · 90C30

Yu-Hong Dai (✉)
e-mail: dyh@lsec.cc.ac.cn

Zexian Liu
liuzexian2008@163.com

Hongwei Liu
hwliu@mail.xidian.edu.cn
1. School of Mathematics and Statistics, Guizhou University, Guiyang, 550025, China
2. LSEC, ICMSEC, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, 100190, China
3. School of Mathematics and Statistics, Xidian University, Xi'an, 710126, China

1 Introduction

Consider the following unconstrained optimization problem

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

where $f:\mathbb{R}^n\to\mathbb{R}$ is continuously differentiable and its gradient is denoted by g.

Throughout this paper, $g_k=g(x_k)$, $f_k=f(x_k)$, $s_{k-1}=x_k-x_{k-1}$ and $y_{k-1}=g_k-g_{k-1}$. If $x\in\mathbb{R}^n$ and $\mathcal{S}\subset\mathbb{R}^n$, then $\operatorname{dist}\{x,\mathcal{S}\}=\inf\{\|y-x\|,y\in\mathcal{S}\}$.

Due to the simplicity, limited storage and nice numerical performance, conjugate gradient (CG) methods are common and efficient methods for unconstrained optimization, especially when n is very large. Giving an initial point x_0, CG methods are the form of

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

where α_k is the stepsize and d_k is the search direction determined by

$$d_k=\begin{cases}-g_k, & \text{if } k=0,\\ -g_k+\beta_k d_{k-1}, & \text{if } k>0,\end{cases} \tag{3}$$

where β_k is often called conjugate parameter.

Different choices of β_k lead to different CG methods. Some well-known formulae for β_k are called the Fletcher-Reeves (FR) [1], Hestenes-Stiefel (HS) [2], Polak-Ribière-Polyak (PRP) [3] and Dai-Yuan (DY) [4] formulae, and are given by

$$\beta_k^{FR}=\frac{\|g_k\|^2}{\|g_{k-1}\|^2},\quad \beta_k^{HS}=\frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}},\quad \beta_k^{PRP}=\frac{g_k^T y_{k-1}}{\|g_{k-1}\|^2},\quad \beta_k^{DY}=\frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}}.$$

Subspace minimization conjugate gradient (SMCG) methods, which can be regarded as the generation of classical CG method [5], are a class of high potential iterate methods for unconstrained optimization. SMCG methods can be dated back to Yuan and Stoer's work [5], and its search direction is determined by solving the following problem:

$$\min_{d_k\in\bar{\Omega}_k} g_k^T d_k+\frac{1}{2}d_k^T B_k d_k, \tag{4}$$

where $\bar{\Omega}_k=Span\{g_k,s_{k-1}\}$ and B_k is a symmetric and positive definite approximation to the Hessian matrix. From then now, SMCG methods have received some attentions [6, 7]. In 2016, motivated by the Barzilai-Borwein (BB) method [8] and based on the SMCG method [5], Dai and Kou [9] presented some efficient Barzilai-Borwein CG methods (BBCG) for strictly convex quadratic minimization problem:

$$\min_{x\in\mathbb{R}^n} q(x)=\frac{1}{2}x^T Ax+b^T x, \tag{5}$$

where $A \in \mathbb{R}^{n\times n}$ is a symmetric and positive definite matrix and $b \in \mathbb{R}^n$; the numerical results in [9] indicated that BBCG3 is the most efficient. Recently, Liu and Liu [10] extended BBCG3 to general unconstrained optimization and presented an efficient Barzilai-Borwein conjugate gradient method with a generalized Wolfe line search (SMCG_BB), and the numerical results in [10] indicate that SMCG_BB is comparable to two well-known CG software packages CGOPT [11] and CG_DESCENT (5.3) [12]. Based on SMCG_BB [10], some efficient SMCG methods [13–17] were later proposed.

Some remarkable properties of linear CG method include the conjugacy of the search directions and the orthogonality of the gradients, which are also of great importance to the numerical performance of nonlinear CG methods. It is however observed that the orthogonality is often lost when linear CG method solves strictly convex minimization problems. Hager and Zhang [18] observed that, when solving the strictly convex quadratic minimization problem PALMER1C in the CUTEr library [21], with the same exact line search PRP^+ CG method [19] converges much slower than the L-BFGS method [20], although these two methods should generate exactly the same iterations theoretically; they thought that the slow convergence may be caused by the loss of orthogonality. To improve the orthogonality, Hager and Zhang [18] first tried to use the L-BFGS method to improve the orthogonality while detecting the loss of orthogonality, and presented a limited memory conjugate gradient method (CG_DESCENT(6.0)); the numerical results in [18] indicated that CG_DESCENT(6.0) has a great improvement over the memoryless version CG_DESCENT(5.3) and is also faster than the L-BFGS method for the test problems in CUTEr library [21].

Although the limited memory CG method [18] is pretty efficient, it also has some drawbacks: (i)The limited memory CG method [18] can be regarded as a preconditioned version of CG method (9) with

$$\beta_k^{HZ} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} - \frac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}} \frac{g_k^T d_{k-1}}{d_{k-1}^T y_{k-1}}. \tag{6}$$

Its convergence is established by imposing the assumptions:

$$\|P_k\| \le \gamma_0, \qquad g_{k+1}^T P_k g_{k+1} \ge \gamma_1 \|g_{k+1}\|^2, \quad d_k^T P_k^{-1} d_k \ge \gamma_2 \|d_k\|^2, \tag{7}$$

where $\gamma_0 > 0$, $\gamma_1 > 0$ and $\gamma_2 > 0$, on its preconditioners:

$$P_k = I, \quad P_k = Z_k \hat{B}_{k+1}^{-1} Z_k^T, \quad P_k = Z_k \hat{B}_{k+1}^{-1} Z_k^T + \sigma_k \overline{Z}_k \overline{Z}_k^T, \tag{8}$$

where σ_k is given by (4.2) of [18], $\hat{B}_{k+1}$ is an approximation to the Hessian matrix of f at the subspace spanned by the previous search directions, and Z_k and $\overline{Z}_k$ are the matrices whose columns are the orthogonal bases for the above subspace and its complement, respectively. It is obvious that the assumptions (7) are quite strict and the preconditioners (8) are pretty complicated. (ii)CG_DESCENT (6.0) with the AWolfe line search [22]:

$$\sigma g_k^T d_k \le g(x_k + \alpha_k d_k)^T d_k \le (2\delta - 1)\, g_k^T d_k,$$

where $0 < \delta < 0.5$ and $\delta \leq \sigma < 1$, has illustrated nice numerical performance, but there is no guarantee for the convergence of CG_DESCENT with the AWolfe line search [22]. While CG DESCENT (6.0) with the standard Wolfe line search is globally convergent, but it performs significantly worse than CG DESCENT (6.0) with the AWolfe line search.

What about the numerical behavior of the combination of SMCG method with the limited memory technique? Can one overcome the above drawbacks when combining SMCG method with the limited memory technique? The aim of our work is to answer the above questions. In this paper, based on BBCG3 [9] and SMCG_BB [10], we present a limited memory SMCG method for unconstrained optimization, which includes two types of iterations: SMCG iteration and quasi-Newton (QN) iteration. In the SMCG iteration, the search directions are determined by solving the quadratic approximation problem, in which the important parameter is estimated based on some important properties of f at x_k. In the QN iteration, a modified quasi-Newton method in the subspace spanned by some previous search directions is designed to improve the orthogonality, and some properties about the quasi-Newton matrices are analyzed. A modified strategy for choosing the initial stepsize is exploited. The global convergence of the proposed method is established under weak conditions in comparison to the limited memory CG method [18]. Some numerical experiments are conducted, which indicate that, for the test problems in the CUTEr library [21], the proposed method has a tremendous improvement over SMCG_BB, is superior to the latest limited memory CG software package CG_DESCENT (6.8) with the Wolfe line search, is also comparable to CG_DESCENT (6.8) with the default and very efficient line search—the hybridization of the Wolfe line search and AWolfe line search, and is superior to L-BFGS method.

The remainder of this paper is organized as follows. In Sect. 2, we describe the type of the iteration, determine the search directions in the SMCG iteration and design a modified quasi-Newton method to improve the orthogonality in the QN iteration, in which some important properties of the QN matrices are analyze; we also design a modified strategy for choosing the initial stepsize and describe a limited memory SMCG algorithm detailed. In Sect. 3, the convergence of the proposed method is established under weaker conditions. Some numerical experiments are conducted in Sect. 4. Conclusions are given in the last section.

2 The limited memory subspace minimization conjugate gradient algorithm

In the section, we describe the iteration types of the limited memory SMCG algorithm, determine the search direction in the SMCG iteration by solving a quadratic approximation problem, design a modified quasi-Newton method in the subspace $\mathcal{S}_k$ spanned by the previous $m > 0$ search directions

$$\mathcal{S}_k = span\{d_{k-1}, d_{k-1}, \cdots, d_{k-m}\}$$

to improve the orthogonality in the QN iteration. We also develop a modified strategy for choosing the initial stepsize and describe the limited memory SMCG algorithm detailed.

2.1 The iteration type

The limited memory SMCG algorithm mainly includes two types of iterations which are SMCG iteration and quasi-Newton (QN) iteration, respectively.

(1) SMCG iteration

The search directions in the SMCG iteration are generated by solving the problem (4), which are described detailed as follows.

We denote the search direction

$$d_k = ug_k + vs_{k-1}, \tag{9}$$

where $u, v \in \mathbb{R}$. Substituting (9) into (4) and using the standard secant equation $B_k s_{k-1} = y_{k-1}$ yield the following quadratic approximation problem:

$$\min_{u,v\in\mathbb{R}} \begin{pmatrix} \|g_k\|^2 \\ g_k^T s_{k-1} \end{pmatrix}^T \begin{pmatrix} u \\ v \end{pmatrix} + \frac{1}{2} \begin{pmatrix} u \\ v \end{pmatrix}^T \begin{pmatrix} \rho_k & g_k^T y_{k-1} \\ g_k^T y_{k-1} & s_{k-1}^T y_{k-1} \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}, \tag{10}$$

where $\rho_k = g_k^T B_k g_k$ is remained to be determined.

Based on some properties of the objective f at the current x_k, we determine ρ_k by the following cases.

(i) If the conditions

$$\frac{y_{k-1}^T y_{k-1}}{s_{k-1}^T y_{k-1}} \le \xi_2 \text{ and } \frac{s_{k-1}^T y_{k-1}}{s_{k-1}^T s_{k-1}} \ge \frac{\xi_3}{\sqrt{k}} \tag{11}$$

hold, where $\xi_2 > 10^4$ and $0 < \xi_3 < 10^{-4}$, then the condition number of the Hessian matrix might be not very large. Similar to BBCG3 [9], we also use $\frac{3}{2}\frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}} I$ to estimate B_k in the term ρ_k, which implies $\rho_k = \frac{3}{2}\frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}}\|g_k\|^2$. Substituting the resulting ρ_k into (10), we can easily obtain

$$u_k = \frac{1}{\Delta_k}\left(g_k^T y_{k-1} g_k^T s_{k-1} - s_{k-1}^T y_{k-1}\|g_k\|^2\right), \quad v_k = \frac{1}{\Delta_k}\left(g_k^T y_{k-1}\|g_k\|^2 - \rho_k g_k^T s_{k-1}\right), \tag{12}$$

where

$$\Delta_k = \begin{vmatrix} \rho_k & g_k^T y_{k-1} \\ g_k^T y_{k-1} & s_{k-1}^T y_{k-1} \end{vmatrix} = \rho_k s_{k-1}^T y_{k-1} - \left(g_k^T y_{k-1}\right)^2 > 0. \tag{13}$$

Therefore, if the conditions (11) hold, the search direction in the SMCG iteration is determined by

$$d_k = u_k g_k + v_k s_{k-1}, \tag{14}$$

where u_k and v_k are given by (12).

(ii) If the following conditions

$$\frac{y_{k-1}^T y_{k-1}}{s_{k-1}^T y_{k-1}} > \xi_2, \ \frac{s_{k-1}^T y_{k-1}}{s_{k-1}^T s_{k-1}} \ge \frac{\xi_3}{\sqrt{k}} \text{ and } \frac{\left|s_{k-1}^T g_k y_{k-1}^T g_k\right|}{s_{k-1}^T y_{k-1}\|g_k\|^2} \le \xi_1 \tag{15}$$

hold, where $0 < \xi_1 \leq 10^{-4}$, then the condition number of the Hessian matrix is likely to be very large. In the case, it seems that it is too simple to use the scalar matrix $\frac{3}{2}\frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}} I$ to estimate B_k in ρ_k. Here we set $B_k = I + \frac{y_{k-1}y_{k-1}^T}{s_{k-1}^T y_{k-1}}$, which implies $\rho_k = \|g_k\|^2 + \frac{\left(g_k^T y_{k-1}\right)^2}{s_{k-1}^T y_{k-1}}$. Substituting the resulting ρ_k into (10), we can easily obtain that

$$\bar{u}_k = -1 + \frac{g_k^T y_{k-1} g_k^T s_{k-1}}{s_{k-1}^T y_{k-1}\|g_k\|^2}, \quad \bar{v}_k = \left(1 - \frac{g_k^T y_{k-1} g_k^T s_{k-1}}{s_{k-1}^T y_{k-1}\|g_k\|^2}\right)\frac{g_k^T y_{k-1}}{s_{k-1}^T y_{k-1}} - \frac{g_k^T s_{k-1}}{s_{k-1}^T y_{k-1}}. \tag{16}$$

Therefore, if the conditions (15) hold, the search direction in the SMCG iteration is determined by

$$d_k = \bar{u}_k g_k + \bar{v}_k s_{k-1}, \tag{17}$$

where $\bar{u}_k$ and $\bar{v}_k$ are given by (16).

(iii) If neither (11) nor (15) holds, then $d_k = -g_k$.

In sum, the search directions in SMCG iteration are described as follows:

$$d_k = \begin{cases} u_k g_k + v_k s_{k-1}, & \text{if (11) holds,} \\ \bar{u}_k g_k + \bar{v}_k s_{k-1}, & \text{if (15) holds,} \\ -g_k, & \text{otherwise,} \end{cases} \tag{18}$$

where u_k and v_k are given by (12), and $\bar{u}_k$ and $\bar{v}_k$ are given by (16).

It is noted that the search direction (14) is similar to that in BBCG3 [9], and the search direction (17) is similar to that in Case III in SMCG_BB [10].

The SMCG iteration is continued until the loss of the orthogonality is detected. When the orthogonality is lost, the iteration will turn to the following QN iteration.

(2) QN iteration

When the orthogonality of the successive gradients is lost, we develop a modified quasi-Newton method in the subspace to improve the orthogonality. Specifically, while the loss of the orthogonality is detected, the SMCG iteration is terminated temporarily and then a modified quasi-Newton method described in Subsection 3.2 is used to improve the orthogonality. It is noted that the quasi-Newton direction in the subspace $\mathcal{S}_k$ is always transformed to the full space $\mathbb{R}^n$ at each QN iteration. Once the orthogonality is improved, the QN iteration is stopped and the SMCG iteration is evoked immediately.

2.2 A modified quasi-Newton method in the subspace for improving the orthogonality

We develop a modified quasi-Newton method in the subspace to improve the orthogonality in the subsection.

Theorem 2.1 *[23] Suppose that the iterate $\{x_k\}$ is generated by the linear conjugate gradient method with the exact line search for minimizing the strictly convex quadratic minimization problem (5), and x_k is not the*

solution point x^*. *Then,*

$$g_k^T d_i = 0, \qquad i = 0, 1, \cdots, k-1$$

and x_k *is the minimizer of* (5) *over the subset* $\Omega_k = \{x | x = x_0 + span\{d_0, d_1, \cdots, d_{k-1}\}\}$.

From Theorem 2.1, we observe that x_k is the minimizer of (5) in the subset Ω_k and thus each subset Ω_k has been fully utilized in this sense.

Let $S_k \in \mathbb{R}^{n\times m}$ be such a matrix whose columns are $d_{k-1}, d_{k-2}, \ldots, d_{k-m}$. We suppose that the columns of S_k are linearly independent. It is also observed that the case of linear dependence rarely occurs. Let the QR factorization of S_k be $S_k = Z_k \bar{R}_k$, where the columns of $Z_k \in \mathbb{R}^{n\times m}$ form the normal orthogonal basis of $\mathcal{S}_k$ and $\bar{R}_k \in \mathbb{R}^{m\times m}$ is the upper triangular matrix with positive diagonal entries.

If g_k is almost in the subspace $\mathcal{S}_k$, which can be measured by the distance of g_k and $\mathcal{S}_k$, namely,

$$\text{dist}\{g_k, \mathcal{S}_k\} \leq \widetilde{\eta}_0 \|g_k\|, \tag{19}$$

where $0 < \widetilde{\eta}_0 < 1$ and $\widetilde{\eta}_0$ is small, then it is obvious that the orthogonality of the successive gradients has lost. Since the columns of Z_k form the normal orthogonal basis of $\mathcal{S}_k$, it is not difficult to know from the definition of dist $\{g_k, \mathcal{S}_k\}$ that (19) can be written as

$$\left(1 - \widetilde{\eta}_0^2\right) \|g_k\|^2 \leq \left\|Z_k^T g_k\right\|^2. \tag{20}$$

In the case, according to Theorem 2.1, it is possible that x_k is possible to be far away from the minimizer of the objective function over the subset Ω_k. Based on the above observation, it seems that when the inequality (20) holds, it is better to optimize in the subspace $\mathcal{S}_k$ as the subspace $\mathcal{S}_k$ has not been fully utilized. As a result, we stop the SMCG iteration and turn to optimize the objective function over $\mathcal{S}_k$, namely,

$$\min_{z\in\mathcal{S}_k} f(x_k + z). \tag{21}$$

If the gradient g_{k+1} becomes sufficiently orthogonal to the subspace, which can be measured by

$$\text{dist}\{g_{k+1}, \mathcal{S}_k\} \geq \widetilde{\eta}_1 \|g_{k+1}\|, \tag{22}$$

where $0 < \widetilde{\eta}_0 < \widetilde{\eta}_1 < 1$, then the iteration will leave the subspace $\mathcal{S}_k$. Similar to (20), the inequality (22) can also be written as

$$\left(1 - \widetilde{\eta}_1^2\right) \|g_{k+1}\|^2 \geq \left\|g_{k+1}^T Z_k\right\|^2. \tag{23}$$

In the limited memory CG method [18], the L-BFGS method is used to solve the subproblem (21) for restoring the orthogonality. It is, however, observed that it is required to impose the strict assumptions (7) on the preconditioners (8) in the convergence analysis of the limited memory CG method [18]. Additionally,

it is well-known that the L-BFGS method only enjoys linear convergence rate. Since the dimension m of the subspace $\mathcal{S}_k$ is usually small, it may be more suitable to use a quasi-Newton method to solve (21).

Now we develop a modified quasi-Newton method in the subspace $\mathcal{S}_k$ for solving the subproblem (21).

The search direction of quasi-Newton method [23] for solving (1) is the form: $d_k = B_k^{-1} g_k$, where B_k is an approximation to the Hessian matrix. Li and Fukushima [24] presented a cautious quasi-Newton method for nonconvex unconstrained optimization, where B_k is updated by

$$B_{k+1} = \begin{cases} B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{s_k^T y_k}, & \text{if } \frac{s_k^T y_k}{\|s_k\|^2} > \upsilon \|g_k\|^\alpha, \\ B_k, & \text{otherwise,} \end{cases}$$

where $\upsilon > 0$ and $\alpha > 0$.

In what follows, the symbol with hat means that it belongs to the subspace $\mathcal{S}_k$, distinguishing from the corresponding symbol in the full space $\mathbb{R}^n$.

Let $\hat{x} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_m)^T \in \mathbb{R}^m$. The subproblem (21) can be stated as

$$\min_{\hat{x} \in \mathbb{R}^m} \ \hat{f}(\hat{x}) = f\left(x_k + \hat{x}_1 d_{k-1} + \hat{x}_2 d_{k-2} + \cdots + \hat{x}_m d_{k-m}\right). \tag{24}$$

The search direction of the modified quasi-Newton method for solving subproblem (24) takes the form of

$$\hat{d}_{k+1} = -\hat{B}_{k+1}^{-1} \hat{g}_{k+1}, \tag{25}$$

where $\hat{B}_{k+1} \in \mathbb{R}^{m \times m}$ is a symmetric and positive definite approximation to the Hessian matrix of $\hat{f}$. To ensure good convergence, the matrix $\hat{B}_k$ will be reset to the identity matrix $\hat{I} \in \mathbb{R}^{m \times m}$ after it is updated l times, where

$$l = \max\left(m^2, 45\right). \tag{26}$$

Motivated by [24] , the search direction of the modified quasi-Newton method for subproblem is (25) with

$$\hat{B}_{k+1} = \begin{cases} \hat{B}_k - \frac{\hat{B}_k \hat{s}_k \hat{s}_k^T \hat{B}_k}{\hat{s}_k^T \hat{B}_k \hat{s}_k} + \frac{\hat{y}_k \hat{y}_k^T}{\hat{s}_k^T \hat{y}_k}, & \text{if mod } (k,l) \neq 0 \text{ and } \frac{\hat{s}_k^T \hat{y}_k}{\hat{s}_k^T \hat{s}_k} \geq \upsilon, \\ \hat{I}, & \text{otherwise,} \end{cases} \tag{27}$$

where mod (k,l) denotes the residue for k modulo n and $\upsilon > 0$. It is not difficult to verify that $\hat{B}_{k+1}$ is symmetric and positive definitive when $\hat{B}_k$ is symmetric and positive definitive and $\hat{s}_k^T \hat{y}_k > 0$.

In the practice, the search direction $\hat{d}_{k+1} \in \mathbb{R}^m$ of the modified quasi-Newton method is always transformed to the full space $\mathbb{R}^n$ at each QN iteration. According to the QR factorization of S_k, it is not difficult to obtain the search direction in the full space $\mathbb{R}^n$:

$$d_{k+1} = -P_k g_{k+1}, \tag{28}$$

where

$$P_k = Z_k \hat{B}_{k+1}^{-1} Z_k^T \tag{29}$$

and $\hat{B}_{k+1}$ is given by (27).

Remark 1 It is noted that, when the orthogonality is improved, the limited memory CG method [18] first performs the iteration with the complicated preconditioner corresponding to the third term in (8) and then takes the iteration with $P_k = I$, while our method performs the SMCG iteration immediately. It is clear that our method is simpler.

We do the following assumptions and then study some important properties of the QN matrices (27) and (29).

Assumption 2.1 *(i) The objective function $f(x)$ is continuously differentiable on $\mathbb{R}^n$ (ii)The objective function $f(x)$ is bounded below on $\mathbb{R}^n$; (iii) The gradient $g(x)$ is Lipschitz continuous on $\mathbb{R}^n$, namely, there exists $L > 0$ such that*

$$\| g(x) - g(y) \| \leq L \| x - y \|, \quad \forall x, y \in \mathbb{R}^n.$$

Lemma 2.1 *Assume that f satisfies Assumption 2.1. Then, for $\hat{B}_{k+1}$ in (27), there exist three constants $\bar{\xi}_1 > 0$, $\bar{\xi}_2 > 0$ and $\bar{\xi}_3 > 0$ such that*

$$\lambda_{\max}\left(\hat{B}_{k+1}\right) \leq \bar{\xi}_1, \quad \lambda_{\max}\left(\hat{B}_{k+1}^{-1}\right) \leq \bar{\xi}_2, \quad \left\|\hat{B}_{k+1}^{-1}\right\| \leq \bar{\xi}_3.$$

Proof Since the columns of Z_k forms the normal orthogonal basis for $\mathcal{S}_k$ and $m < +\infty$, it follows that there exists $\xi_0 > 0$ such that $\|Z_k\| \leq \xi_0$. According to Assumption 2.1 (iii), (27) and the property of the maximum eigenvalue function $\lambda_{\max}(\cdot)$: $\lambda_{\max}(A_1 + A_2) \leq \lambda_{\max}(A_1) + \lambda_{\max}(A_2)$, where $A_1 \in \mathbb{R}^{m\times m}$ and $A_2 \in \mathbb{R}^{m\times m}$ are symmetric matrices, we have $\lambda_{\max}\left(\hat{B}_{k+1}\right) = 1$ or

$$\begin{aligned}
\lambda_{\max}\left(\hat{B}_{k+1}\right) &\leq \lambda_{\max}\left(\hat{B}_k\right) + \lambda_{\max}\left(-\frac{\hat{B}_k \hat{s}_k \hat{s}_k^T \hat{B}_k}{\hat{s}_k^T \hat{B}_k \hat{s}_k}\right) + \lambda_{\max}\left(\frac{\hat{y}_k \hat{y}_k^T}{\hat{s}_k^T \hat{y}_k}\right) \\
&\leq \lambda_{\max}\left(\hat{B}_k\right) + \frac{\hat{y}_k^T \hat{y}_k}{\hat{s}_k^T \hat{y}_k} \\
&\leq \hat{\lambda}_{\max}\left(\hat{B}_k\right) + L^2 \xi_0^2 \frac{\|\hat{s}_k\|^2}{\hat{s}_k^T \hat{y}_k} \\
&\leq \hat{\lambda}_{\max}\left(\hat{B}_k\right) + \frac{L^2 \xi_0^2}{\upsilon}.
\end{aligned}$$

Since $\hat{B}_k$ will be set to $\hat{I}$ after updating at most l times, we obtain $\lambda\left(\hat{B}_{k+1}\right) \leq 1 + \frac{lL^2\xi_0^2}{\upsilon} \triangleq \bar{\xi}_1$.

Let $\hat{P}_k = \hat{B}_{k+1}^{-1}$. According to (27), after some matrix operations we obtain $\hat{P}_k = \hat{I}$ or

$$\hat{P}_k = \left(\hat{I} - \frac{\hat{y}_k \hat{s}_k^T}{\hat{s}_k^T \hat{y}_k}\right)^T \hat{P}_{k-1} \left(\hat{I} - \frac{\hat{y}_k \hat{s}_k^T}{\hat{s}_k^T \hat{y}_k}\right) + \frac{\hat{s}_k \hat{s}_k^T}{\hat{s}_k^T \hat{y}_k}. \tag{30}$$

It is not difficult to see that $\lambda_{\max}\left(\left(\hat{I}-\frac{\hat{y}_k\hat{s}_k^T}{\hat{s}_k^T y_k}\right)^T\left(\hat{I}-\frac{\hat{y}_k\hat{s}_k^T}{\hat{s}_k^T\hat{y}_k}\right)\right)=\frac{\|\hat{y}_k\|^2\|\hat{s}_k\|^2}{\left(\hat{s}_k^T\hat{y}_k\right)^2}$. For any $\hat{z}\neq 0\in\mathbb{R}^m$ and $\hat{P}_k$ in (30), we have

$$\begin{aligned}
\hat{z}^T\hat{P}_k\hat{z} &= \hat{z}^T\left(\hat{I}-\frac{\hat{y}_k\hat{s}_k^T}{\hat{s}_k^T\hat{y}_k}\right)^T\hat{P}_{k-1}\left(\hat{I}-\frac{\hat{y}_k\hat{s}_k^T}{\hat{s}_k^T\hat{y}_k}\right)\hat{z}+\frac{\left(\hat{s}_k^T\hat{z}\right)^2}{\hat{s}_k^T\hat{y}_k}\\
&\leq \hat{\lambda}_{\max}\left(\hat{P}_{k-1}\right)\hat{z}^T\left(\hat{I}-\frac{\hat{y}_k\hat{s}_k^T}{\hat{s}_k^T\hat{y}_k}\right)^T\left(\hat{I}-\frac{\hat{y}_k\hat{s}_k^T}{\hat{s}_k^T\hat{y}_k}\right)\hat{z}+\frac{\left(\hat{s}_k^T\hat{z}\right)^2}{\hat{s}_k^T\hat{y}_k}\\
&\leq \lambda_{\max}\left(\hat{P}_{k-1}\right)\lambda_{\max}\left(\left(\hat{I}-\frac{\hat{y}_k\hat{s}_k^T}{\hat{s}_k^T\hat{y}_k}\right)^T\left(I-\frac{\hat{y}_k\hat{s}_k^T}{\hat{s}_k^T\hat{y}_k}\right)\right)\|\hat{z}\|^2+\frac{\left(\hat{s}_k^T\hat{z}\right)^2}{\hat{s}_k^T\hat{y}_k}\\
&\leq \hat{\lambda}_{\max}\left(\hat{P}_{k-1}\right)\frac{\|\hat{y}_k\|^2\|\hat{s}_k\|^2}{\left(\hat{s}_k^T\hat{y}_k\right)^2}\|\hat{z}\|^2+\frac{\|\hat{s}_k\|^2}{\hat{s}_k^T\hat{y}_k}\|\hat{z}\|^2.
\end{aligned}$$

Dividing the above inequality by $\|\hat{z}\|^2$ and maximizing the resulting inequality, we can obtain

$$\begin{aligned}
\lambda_{\max}\left(\hat{P}_k\right) &\leq \lambda_{\max}\left(\hat{P}_{k-1}\right)\frac{\|\hat{y}_k\|^2\|\hat{s}_k\|^2}{\left(\hat{s}_k^T\hat{y}_k\right)^2}+\frac{\|\hat{s}_k\|^2}{\hat{s}_k^T\hat{y}_k}\\
&\leq \lambda_{\max}\left(\hat{P}_{k-1}\right)L^2\xi_0^2\frac{\|\hat{s}_k\|^4}{\left(\hat{s}_k^T\hat{y}_k\right)^2}+\frac{\|\hat{s}_k\|^2}{\hat{s}_k^T\hat{y}_k}\\
&\leq \frac{L^2\xi_0^2}{v^2}\lambda_{\max}\left(\hat{P}_{k-1}\right)+\frac{1}{v}.
\end{aligned}$$

The above second inequality comes from Assumption 2.1 (iii). Since $\hat{P}_k$ will be set to $\hat{I}$ after updating at most l times, we can easily know that there exists a constant $\bar{\xi}_2>0$ such that $\lambda\left(\hat{B}_{k+1}^{-1}\right)=\lambda\left(\hat{P}_k\right)\leq\bar{\xi}_2$.

Since $\hat{B}_{k+1}^{-1}$ is a symmetric and positive definite matrix, we have that $\left\|\hat{B}_{k+1}^{-1}\right\|_2^2=\lambda_{\max}\left(\hat{B}_{k+1}^{-1}\right)\leq\bar{\xi}_2$. Therefore, by the equivalence of matrix norm in finite dimensional space and $\hat{B}_{k+1}^{-1}\in\mathbb{R}^{m\times m}$, we know that there exists $\bar{\xi}_3>0$ such that $\left\|\hat{B}_{k+1}^{-1}\right\|<\bar{\xi}_3$. The proof is completed. □

Lemma 2.2 *Assume that f satisfies Assumption 2.1. Then, for P_k in (29), there exist three constants $\gamma_0>0$, $\gamma_1>0$ and $\gamma_2>0$ such that*

$$\|P_k\|\leq\gamma_0,\quad g_{k+1}^TP_kg_{k+1}\geq\gamma_1\|g_{k+1}\|^2,\quad d_k^TP_k^{-1}d_k\geq\gamma_2\|d_k\|^2,$$

where P_k^{-1} denotes the pseudoinverse of P_k.

Proof By (23) and Lemma 2.1, we obtain

$$\|P_k\|=\left\|Z_k\hat{B}_{k+1}^{-1}Z_k^T\right\|=\left\|\hat{B}_{k+1}^{-1}\right\|\leq\bar{\xi}_3\triangleq\gamma_0,$$

$$g_{k+1}^TP_kg_{k+1}=g_{k+1}^TZ_k\hat{B}_{k+1}^{-1}Z_k^Tg_{k+1}=\hat{g}_{k+1}^T\hat{B}_{k+1}^{-1}\hat{g}_{k+1}\geq\lambda_{\min}\left(\hat{B}_{k+1}^{-1}\right)\|\hat{g}_{k+1}\|^2\geq\frac{1}{\bar{\xi}_1}\left(1-\tilde{\eta}_1^2\right)\|g_{k+1}\|^2\triangleq\gamma_1\|g_{k+1}\|^2,$$

$$d_k^TP_k^{-1}d_k=d_k^TZ_k\hat{B}_{k+1}Z_k^Td_k=\hat{d}_k^T\hat{B}_{k+1}\hat{d}_k\geq\frac{1}{\bar{\xi}_2}\left\|\hat{d}_k\right\|^2=\frac{1}{\bar{\xi}_2}\|d_k\|^2\triangleq\gamma_2\|d_k\|^2.$$

Therefore, we can obtain the conclusions. The proof is completed. □

2.3 A modified strategy for choosing the initial stepsize

As we know, the choice of the initial stepsize is of great importance to the numerical performance of an optimization method. Based on the strategy in SMCG_BB [10], we design a modified strategy for choosing the initial stepsize in this subsection.

As same as SMCG_BB, the initial stepsize $\alpha_k^0 (k=0)$ at the first iteration is determined by

$$\alpha_0^0 = \begin{cases} 1, & \text{if } |f| \leq 10^{-30} \text{ and } \|x_0\|_\infty \leq 10^{-30}, \\ 2|f|/\|g_0\|, & \text{if } |f| > 10^{-30} \text{ and } \|x_0\|_\infty \leq 10^{-30}, \\ \min\left\{1, \|x_0\|_\infty / \|g_0\|_\infty\right\}, & \text{if } \|g_0\|_\infty < 10^7 \text{ and } \|x_0\|_\infty > 10^{-30}, \\ \min\left\{1, \max\left\{1, \|x_0\|_\infty\right\} / \|g_0\|_\infty\right\}, & \text{if } \|g_0\|_\infty \geq 10^7 \text{ and } \|x_0\|_\infty > 10^{-30}. \end{cases} \tag{31}$$

As for the initial stepsize $\alpha_k^0 (k>0)$, we determine it based on the following observations.

According to [25],

$$\mu_k = \left| \frac{2\left(f_{k-1} - f_k + g_k^T s_{k-1}\right)}{s_{k-1}^T y_{k-1}} - 1 \right| \tag{32}$$

is a quantity showing how $f(x)$ is close to a quadratic function on the line segment between x_{k-1} and x_k. If the following condition [26,27] holds, i.e.,

$$\mu_k \leq \xi_5 \quad \text{or} \quad \max\{\mu_k, \mu_{k-1}\} \leq \xi_6, \tag{33}$$

where $0 < \xi_5 < \xi_6$, then f may be close to a quadratic function on the line segment between x_{k-1} and x_k.

It is universally acknowledged that the linear CG method with the exact line search enjoys the quadratic termination for strictly convex quadratic functions. Additionally, Andrei [28] thought that the higher accuracy of the stepsize, the faster convergence rate of a CG method. Based on the above observations, if f is close to a quadratic function on the line segment between x_{k-1} and x_k, then it is reasonable to choose the minimizer of the interpolation function $q\left(\phi_k(0), \phi_k'(0), \phi_k(\alpha)\right)$ as the initial stepsize for a CG method, where $\phi_k(\alpha) = f(x_k + \alpha d_k)$ and $\alpha > 0$ is a trial stepsize.

We first consider the initial stepsize for the search direction in the QN iteration.

(i)The initial stepsize for the search direction (28) with $\hat{B}_k \neq \hat{I}$.

Since the search direction $\hat{d}_k$ is a quasi-Newton direction in the subspace, it is natural to choose $\alpha_k^0 = 1$ as the trial initial stepsize.

Let

$$\bar{\alpha}_k = \min\ q\left(\phi_k(0), \phi_k'(0), \phi_k(1)\right). \tag{34}$$

If the condition (33) holds and $\bar{\alpha}_k > 0$, then we set the initial stepsize as

$$\hat{\alpha}_k = \min\left\{\max\left\{\bar{\alpha}_k, \alpha_{\min}\right\}, \alpha_{\max}\right\}, \tag{35}$$

where $\alpha_{\max} > 0$ is very large and $\alpha_{\min} > 0$ is very small. Therefore, the initial stepsize for the search dirction (28) with $\hat{B}_k \neq \hat{I}$ is determined by

$$\alpha_k^0 = \begin{cases} \hat{\alpha}_k, & \text{if (33) holds and } \bar{\alpha}_k > 0, \\ 1, & \text{otherwise.} \end{cases} \tag{36}$$

(ii)The initial stepsize for the search direction (28) with $\hat{B}_k = \hat{I}$.

It is well-known that the BB stepsizes [8], especially the adaptive BB stepsizes [30], are very efficient for gradient method. Here we design an adaptive BB stepsize for the search direction (28) with $\hat{B}_k = \hat{I}$. If the exact line search is adopted, then $g_k^T s_{k-1} = 0$, which implies that if $g_k^T s_{k-1} > 0$, then the stepsize α_{k-1} is larger than the exact stepsize $\alpha_{k-1}^{es} = \arg\min_{\alpha>0} f(x_{k-1} + \alpha d_{k-1})$. In order to compensate the gap, the initial stepsize α_k^0 should be determined by the short BB stepsize $\alpha_k^{BB_2} = \frac{s_{k-1}^T y_{k-1}}{\|y_{k-1}\|^2}$. Similarly, if $g_k^T s_{k-1} \leq 0$, the initial stepsize α_k^0 should be determined by the long BB stepsize $\alpha_k^{BB_1} = \frac{\|s_{k-1}\|^2}{s_{k-1}^T y_{k-1}}$. Therefore, the initial trial stepsize is determined by

$$\bar{\bar{\alpha}}_k = \begin{cases} \left\{\min\left\{\alpha_k^{BB_2}, \alpha_{\max}\right\}, \alpha_{\min}\right\}, & \text{if } g_k^T s_{k-1} > 0, \\ \left\{\min\left\{\alpha_k^{BB_1}, \alpha_{\max}\right\}, \alpha_{\min}\right\}, & \text{if } g_k^T s_{k-1} \leq 0. \end{cases} \tag{37}$$

Let

$$\tilde{\bar{\alpha}}_k = \min\ q\left(\phi_k(0), \phi_k'(0), \phi_k\left(\max\left(\bar{\bar{\alpha}}_k, 5\alpha_{k-1}\right)\right)\right). \tag{38}$$

If $\tilde{\bar{\alpha}}_k > 0$, then the initial stepsize is determined by

$$\tilde{\alpha}_k = \min\left\{\max\left\{\tilde{\bar{\alpha}}_k, \alpha_{\min}\right\}, \alpha_{\max}\right\}. \tag{39}$$

Therefore, the initial stepsize for the search direction (28) with $\hat{B}_k = \hat{I}$ is determined by

$$\alpha_k^0 = \begin{cases} \tilde{\alpha}_k, & \text{if (33) holds, and } \tilde{\bar{\alpha}}_k > 0, \\ \bar{\bar{\alpha}}_k, & \text{otherwise.} \end{cases} \tag{40}$$

As for the initial stepsize for the search direction in SMCG iteration, it is mainly determined by the way similar to that in SMCG_BB [10].

Therefore, the initial stepsize $\alpha_k^0 (k > 0)$ can be described as follows. If the search direction d_k is computed by (18) with $d_k \neq -g_k$ or by (28) with $\hat{B}_k \neq \hat{I}$, then the initial stepsize is determined by

$$\alpha_k^0 = \begin{cases} \hat{\alpha}_k, & \text{if (iteration = SMCG, } \bar{\alpha}_k > 0 \text{ and (33)) or} \\ & \quad \text{(iteration = QN, } \bar{\alpha}_k > 0, \ \ w_k < c_3 \text{ and (33)) hold,} \\ 1, & \text{otherwise,} \end{cases} \tag{41}$$

where $w_k = \frac{\phi_k(1)-\phi_k(0)}{0.001+|\phi(0)|}$, $c_3 > 0$ and $\hat{\alpha}_k$ is given by (35). If the search direction d_k is computed by (18) with $d_k = -g_k$ or by (28) with $\hat{B}_k = \hat{I}$, then the initial stepsize is given by

$$\alpha_k^0 = \begin{cases} \widetilde{\alpha}_k, & \text{if } \Big(\text{iteration} = \text{SMCG},\ \|g_k\|^2 \le 1,\ \widetilde{\widetilde{\alpha}}_k > 0 \text{ and (33)}\Big) \text{ or} \\ & \quad \Big(\text{iteration} = \text{QN},\ w_k < c_3,\ \widetilde{\widetilde{\alpha}}_k > 0 \text{ and (33)}\Big) \text{ hold}, \\ \bar{\bar{\alpha}}_k, & \text{otherwise}, \end{cases} \tag{42}$$

where $\widetilde{\alpha}_k$, $\widetilde{\widetilde{\alpha}}_k$ and $\bar{\bar{\alpha}}_k$ are determined by (39), (38) and (37), respectively.

2.4 Description of the limited memory subspace minimization conjugate gradient algorithm

As commented by Dai and Kou [9], it is important to design a suitable line search when extending BBCG3 to unconstrained optimization. A generalized Wolfe line search is developed in SMCG_BB [10]. As same as SMCG_BB, we also use the generalized Wolfe line search in the limited memory SMCG algorithm, which is described here for completeness.

The generalized Wolfe line search in SMCG_BB is

$$f(x_k + \alpha_k d_k) \le f(x_k) + \eta_k + \sigma\alpha_k g_k^T d_k \tag{43}$$

$$g_{k+1}^T d_k \ge \delta g_k^T d_k, \tag{44}$$

where $0 < \sigma < \delta < 1$ and η_k is required to satisfy $\lim\limits_{k\to+\infty} k\eta_k = 0$. Here we take

$$\eta_k = \begin{cases} 0, & \text{if } k = 0, \\ \min\left\{\frac{1}{k\lg(k/n+12)}, C_k - f(x_k)\right\}, & \text{if } k \ge 1, \end{cases} \tag{45}$$

where C_k is given by

$$C_0 = f(x_0),\ Q_0 = 1,\ Q_{k+1} = t_k Q_k + 1,\ C_{k+1} = \frac{t_k Q_k C_k + f_{k+1}}{Q_{k+1}} \tag{46}$$

and $0 \le \eta_{\min} \le t_k \le \eta_{\max} \le 1$.

We describe the limited memory SMCG algorithm in detail.

Algorithm 1. The limited memory SMCG algorithm (LMSMCG_BB)

Step 0. Given $x_0 \in \mathbb{R}^n$, $\varepsilon > 0$, $\alpha_{\min}, \alpha_{\max}$, σ, δ, $\widetilde{\eta}_0$, $\widetilde{\eta}_1$, ξ_2, ξ_3, ξ_4, ξ_5, ξ_6, ξ_7, υ, MaxRestart and MinQuad. Set IterRestart :=0, Numgrad :=0, IterQuad :=0, Numcongrad :=0, iteration= "SMCG iteration" and $k := 0$.

Step 1. If $||g_k||_\infty \le \varepsilon$, then stop.

Step 2. Compute the search direction.

If (**iteration** ="SMCG iteration"), then

If $k = 0$, then $d_0 = -g_0$ and Numgrad :=1, and go to Step 3.

elseif (neither (11) nor (15) holds), or (Numcongrad=MaxRestar) or (IterQuad=MinQuad and IterRestart $\neq$ IterQuad), compute the search direction by $d_k = -g_k$. Set Numgrad := Numgrad+1, Numcongrad := 0 and IterRestart := 0, and go to Step 3.

elseif the conditions (11) hold, compute d_k by (14). Set Numgrad := 0 and Numcongrad:= Numcongrad+1, and go to Step 3.

elseif the conditions (15) hold, compute d_k by (17). Set Numgrad := 0 and Numcongrad:= Numcongrad+1, and go to Step 3.

end

elseif (**iteration** ="QN iteration"), then

Compute $\hat{B}_k$ by (27) and P_k by (29), and determine the search direction d_k by (28).

end

Step 3. Compute the initial stepsize. If $k = 0$, then compute α_0^0 by (31) and go to Step 4, otherwise, compute the initial stepsize by (41) or (42), and go to Step 4.

Step 4. Line search. Determine α_k satisfying the generalized Wolfe line search (43) and (44) with α_k^0.

Step 5. Set $x_{k+1} = x_k + \alpha_k d_k$.

Step 6. Update IterRestart and IterQuad. Set IterRestart :=IterRestart+1. If $\left|\frac{2(f_{k+1}-f_k)}{(g_{k+1}+g_k)^T s_k} - 1\right| \leq \xi_7$ [11] or $\left|f_{k+1} - f_k - 0.5\left(g_{k+1}^T s_k + g_k^T s_k\right)\right| \leq \xi_8$ [10] holds, then IterQuad := IterQuad+1, otherwise, IterQuad :=0.

Step 7. Update the type of the iteration.

if (iteration="SMCG iteration"), then

if (20) holds, then iteration="QN iteration".

elseif(iteration="QN iteration"), then

if (23) holds, then iteration="SMCG iteration".

end

Step 8. Set $k := k + 1$, go to Step 1.

3 Convergence analysis

In the section, under the Assumption 2.1, we establish the global convergence of Algorithm 1, and analyze the R-linear convergence of Algorithm 1 for uniformly convex functions .

We first study some important properties of the new search directions (18) and (28).

Lemma 3.1 *Assume that f satisfies Assumption 2.1. Then, there exists a constant $\bar{c}_1 > 0$ such that the search directions (18) and (28) satisfy the sufficient descent condition:*

$$g_k^T d_k \leq -\bar{c}_1 \|g_k\|^2. \tag{47}$$

Proof We prove it in the following two cases.

(i) The search direction is given by (18). Similar to the proof of Lemma 3.1 of [10], we can obtain

$$g_k^T d_k \leq -c_1 \|g_k\|^2,$$

where $c_1 = \dfrac{2}{3\xi_2}$ and ξ_2 is the same as that in (11).

(ii) The search direction is given by (28). By Lemma 2.2, we obtain $g_k^T d_k = -g_k^T P_{k-1} g_k \leq -\gamma_1 \|g_k\|^2$.

By setting $\bar{c}_1 = \min\{c_1, \gamma_1\}$, we can obtain (47). The proof is completed. □

Lemma 3.2 *Assume that f satisfies Assumption 2.1. Then, there exists two constants $\bar{c}_2 > 0$ and $\bar{c}_3 > 0$ such that the search directions (18) and (28) satisfy*

$$\|d_k\|^2 \leq (\bar{c}_2 + \bar{c}_3 k) \|g_k\|^2 . \tag{48}$$

Proof We prove it in the following two cases.

(i) The search direction is given by (18). Similar to the proof of Lemma 3.2 [10], we obtain that

$$\|d_k\|^2 \leq (c_2 + c_3 k) \|g_k\|^2 ,$$

where c_2 and c_3 are the same as those in Lemma 3.2 of [10].

(ii) The search direction is given by (28). By Lemma 2.2, we obtain $\|d_k\|^2 = \|-P_{k-1} g_k\|^2 \leq \gamma_0^2 \|g_k\|^2$.

By setting $\bar{c}_2 = \max\{\gamma_0^2, c_2\}$ and $\bar{c}_3 = c_3$, we can obtain (48). The proof is completed. □

The following lemma is used to prove the convergence of Algorithm 1.

Lemma 3.3 *Assume that f satisfies Assumption 2.1. Then,*

$$\alpha_k \geq \frac{(1-\delta)\,\bar{c}_1}{(\bar{c}_2 + \bar{c}_3 k)\, L},$$

where $\bar{c}_1$, $\bar{c}_2$, $\bar{c}_3$ and δ are given by (47), (48) and (44), respectively.

Proof By (44) and Assumption 2.1, we have that

$$(\delta - 1)\, g_k^T d_k \leq g\left(x_k + \alpha_k d_k\right)^T d_k - g_k^T d_k = \left(g\left(x_k + \alpha_k d_k\right) - g_k\right)^T d_k \leq L \alpha_k \|d_k\|^2,$$

which yields

$$\alpha_k \geq \frac{(\delta - 1)\, g_k^T d_k}{L \|d_k\|^2}. \tag{49}$$

By (49), Lemma 3.1 and Lemma 3.2, we obtain that

$$\alpha_k \geq \frac{(\delta - 1)\, g_k^T d_k}{L \|d_k\|^2} \geq \frac{\bar{c}_1 (1-\delta)}{L} \frac{\|g_k\|^2}{\|d_k\|^2} \geq \frac{(1-\delta)\,\bar{c}_1}{L\,(\bar{c}_2 + \bar{c}_3 k)}. \tag{50}$$

The proof is completed. □

Theorem 3.1 *Assume that f satisfies Assumption 2.1, and let $\{x_k\}$ be generated by Algorithm 1. Then,*

$$\liminf_{k\to+\infty} \|g_k\| = 0. \tag{51}$$

Proof By Lemmas 3.1 and 3.2, we know that the search directions (18) and (28) satisfy the sufficient descent condition (47) and (48). Therefore, by Lemma 3.3 and the generalized Wolfe line search (43) and (44), similar to the proof of Theorem 4.1 in [10], we can obtain (51). The proof is completed. □

The following theorem indicates that SMCG_BB is R-linearly convergent for uniformly convex functions.

Theorem 3.2 *Suppose that f is uniformly convex with unique minimizer x, $\eta_{max} < 1$, the gradient g is Lipschitz continuous on bounded sets, and there exists $\mu_{\max} > 0$ such that $\alpha_k \le \mu_{\max}$ for all k. Then, there exists $\theta \in]0,1[$ such that*

$$f_k - f\left(x^*\right) \le \theta^k \left(f_0 - f\left(x^*\right)\right).$$

Proof By Lemmas 3.1 and 3.2, we know that the search directions (18) and (28) satisfy the sufficient descent direction and (48). Therefore, similar to the proof of Theorem 4.3 in [10], we can obtain the desirable inequality $f_k - f\left(x^*\right) \le \theta^k \left(f_0 - f\left(x^*\right)\right)$. The proof is completed. □

Remark 2 It is noted that the convergence of the proposed method is established under Assumption 2.1 without the strict assumptions (7).

4 Numerical experiments

We compare the performance of LMSMCG_BB with that of SMCG_BB [10], the latest limited memory conjugate gradient software package CG_DESCENT (6.8) [18] and the limited memory BFGS method (L-BFGS) [20]. The C codes of CG_DESCENT (6.8) and SMCG_BB can be downloaded from http://users.clas.ufl.edu/hager/papers/Software and http://web.xidian.edu.cn/xdliuhongwei/en/paper.html, respectively. LMSMCG_BB, the code of which will be available in our website finally, is implemented based on the C code of SMCG_BB. Setting the LBFGS parameter in CG_DESCENT (6.8) to TRUE yields to L-BFGS method [18]. The test collection includes 145 problems from the CUTEr library [21], and can be found in http://users.clas.ufl.edu/hager/papers/CG/results6.0.txt; the initial points and the dimensions of the test problems are default. The numerical experiments are done on Ubuntu 10.04 LTS.

In the numerical experiments, we choose the following parameter values for LMSMCG_BB: $\varepsilon = 10^{-6}$, $\sigma = 0.01$, $\delta = 0.9999$, $\upsilon = 10^{-8}$, $\alpha_{\min} = 10^{-30}$, $\alpha_{\max} = 10^{30}$, $t_k = 0.9999$, $\widetilde{\eta}_0 = 10^{-6}$, $\widetilde{\eta}_1 = 0.4$, $\xi_2 = 10^6$, $\xi_3 = 10^{-8}$, $\xi_5 = 5 \times 10^{-4}$, $\xi_6 = 5 \times 10^{-3}$, $\xi_7 = 5 \times 10^{-7}$, $\xi_8 = 10^{-8}$, $m = 11$, MinQuad=3 and MaxRestar= $4n$, and other parameter values in LMSMCG_BB are the same as SMCG_BB. SMCG_BB, L-BFGS method and CG_DESCENT (6.8) use all default parameter values but the stopping condition. It is noted that the number of memory for all test methods is 11. All test methods are terminated if $\| g_k\|_\infty \le 10^{-6}$ is satisfied.

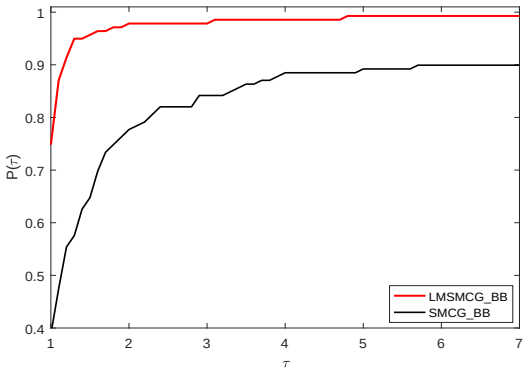

Fig. 1 Performance profile based on N_{iter}

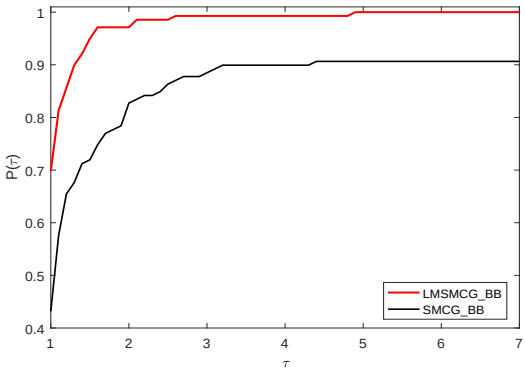

Fig. 2 Performance profile based on N_f

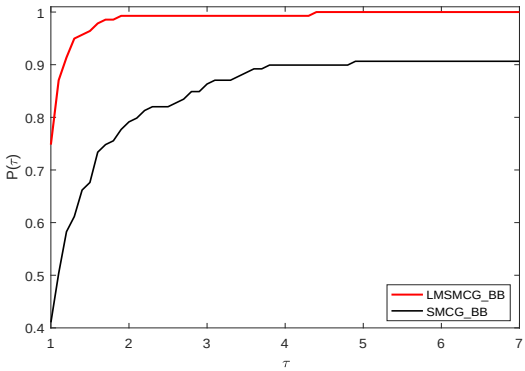

Fig. 3 Performance profile based on N_f

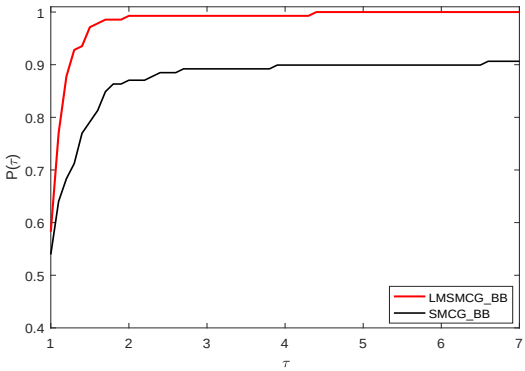

Fig. 4 Performance profile based on N_g

The performance profiles introduced by Dolan and Moré [32] are used to display the performances of the test methods. In the following figures, "N_{iter}", "N_f", "N_g" and "T_{cpu}" represent the number of iterations, the number of function evaluations, the number of gradient evaluations and CPU time (s), respectively.

We divide the numerical experiments into three groups.

In the first group of the numerical experiments, we compare LMSMCG_BB to SMCG_BB. Figs 1, 2, 3 and 4 illustrate the performance profiles of LMSMCG_BB and SMCG_BB in term of the number of iterations, the number of function evaluations, the number of gradient evaluations and CPU time. LMSMCG_BB and SMCG_BB both successfully solve 141 test problems, respectively. As shown in Figs 1, 2, 3 and 4, we observe that LMSMCG_BB is superior much to SMCG_BB in term of the number of iterations, the number of function evaluations, the number of gradient evaluations and CPU time. It indicates that the limited memory technique equipped in LMSMCG_BB indeed brings quite significant numerical improvements.

In the second group of the numerical experiments, we compare LMSMCG_BB to CG_DESCENT (6.8) with different line search procedures. In the C code of CG_DESCENT (6.8), the Wolfe line search and AWolfe line search were both implemented, and the most efficient line search is the hybridization of the Wolfe line search and AWolfe line search, which is also the default line search in CG_DESCENT (6.8) [18]. We first compare LMSMCG_BB to CG_DESCENT(6.8) with the Wolfe line search (called CG_DESCENT(6.8)+Wolfe). LMSMCG_BB successfully solves 141 problems, while CG_DESCENT (6.8)+Wolfe only successfully solves **115** test problems, and all test problems that CG_DESCENT (6.8)+Wolfe failed are due to the failure of

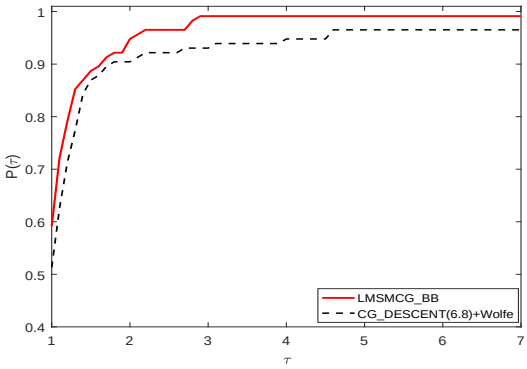

Fig. 5 Performance profile based on N_{iter}

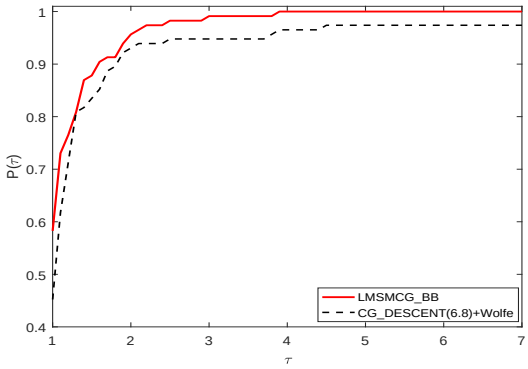

Fig. 6 Performance profile based on N_f

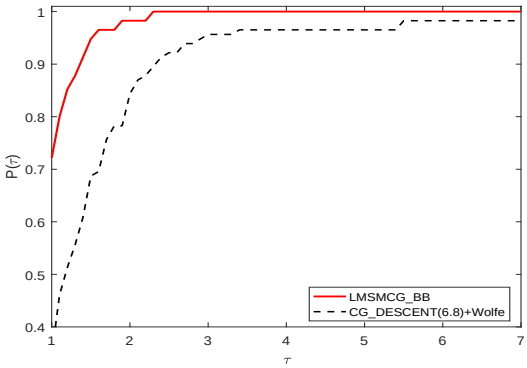

Fig. 7 Performance profile based on N_g

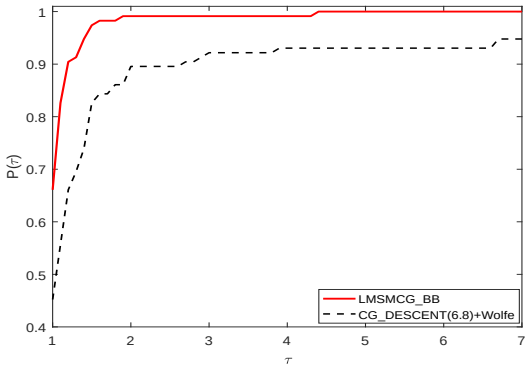

Fig. 8 Performance profile based on T_{cpu}

the Wolfe line search in the process of seeking suitable stepsize. As shown in Figs. 5 and 6, we observe that LMSMCG_BB performs slightly better than CG_DESCENT(6.8)+Wolfe in term of the number of iterations and the number of function evaluations, and Fig. 7 indicates that LMSMCG_BB has relatively large improvement over CG_DESCENT(6.8)+Wolfe in term of the number of gradient evaluations. We see from Fig. 8 that LMSMCG_BB is much faster than CG_DESCENT(6.8)+Wolfe for the CUTEr library.

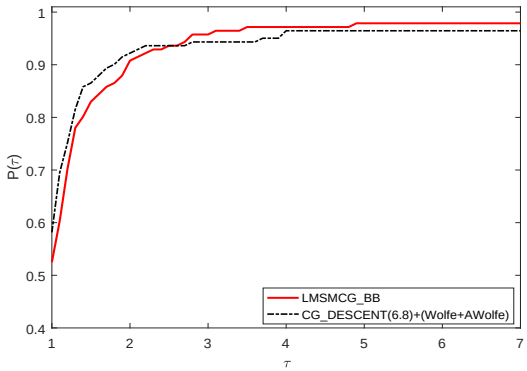

Fig. 9 Performance profile based on N_{iter}

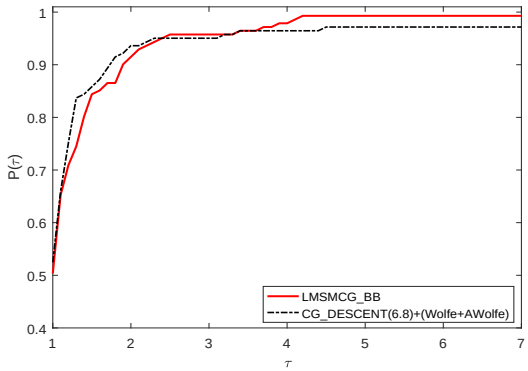

Fig. 10 Performance profile based on N_f

We then compare LMSMCG_BB to CG_DESCENT(6.8) with the hybridization of the Wolfe line search and AWolfe line search (called CG_DESCENT(6.8)+(Wolfe+AWolfe)). CG_DESCENT(6.8)+(Wolfe+AWolfe)

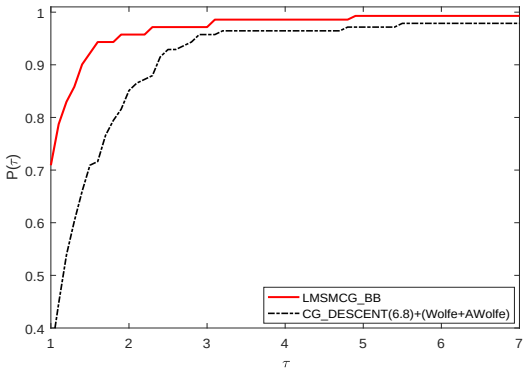

Fig. 11 Performance profile based on N_g

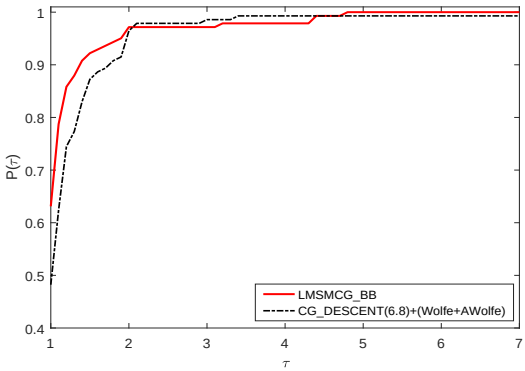

Fig. 12 Performance profile based on T_{cpu}

successfully solves 145 problems. Though Fig.9 and Fig.10 illustrate that LMSMCG_BB is at a little disadvantage over CG_DESCENT(6.8)+(Wolfe+AWolfe) in term of the number of iterations and the number of function evaluations, we can see from Fig.11 that LMSMCG_BB performs much better than CG_DESCENT(6.8)+(Wolfe+AWolfe) in term of the number of gradient evaluations, since LMSMCG_BB is better for about 71% test problems with the least gradient evaluations, while the percentage of CG_DESCENT (6.8)+(Wolfe+AWolfe) is only about 38%. As shown in Fig. 12, we can observe that LMSMCG_BB is slightly faster than CG_DESCENT(6.8)+(Wolfe+AWolfe).

Though the AWolfe line search is very efficient, there is no guarantee for the global convergence of CG_DESCENT with the AWolfe line search [29]. It follows from Theorem 3.1 in Sect. 3 that LMSMCG_BB with the generalized Wolfe line search (43) and (44) used is globally convergent. The second group of the numerical experiments indicates that LMSMCG_BB is superior much to CG_DESCENT(6.8) with the Wolfe line search which can keep the global convergence, and is also comparable with CG_DESCENT(6.8) with the hybridization of the Wolfe line search and AWolfe line search for which there is no guarantee for the global convergence.

In the third group of the numerical experiments, we compare LMSMCG_BB to L-BFGS method with the hybridization of the Wolfe line search and AWolfe line search (called L-BFGS+(Wolfe+AWolfe)). The Fortran code of L-BFGS method on Jorge Nocedal's web page was executed for solving the 145 test problems in the CUTEr library, and it failed in 33 of the 145 test problems [18], while the version of L-BFGS with the Wolfe line search contained in CG_DESCENT (6.8) can only solve successfully 116 test problems, and the version of L-BFGS with the hybridization of the Wolfe line search and AWolfe line search contained in CG_DESCENT (6.8) can successfully solve 144 test problems. So the version of L-BFGS with the hybridization of the Wolfe line search and AWolfe line search contained in CG_DESCENT (6.8) was used to compare with LMSMCG_BB. Though Figs.13 and 14 illustrate that LMSMCG_BB is at a little advantage over L-BFGS+(Wolfe+AWolfe) in term of the number of iterations and the number of function evaluations, we can see from Fig.15 that LMSMCG_BB is superior much to L-BFGS+(Wolfe+AWolfe) in term of the number of gradient evaluations, since LMSMCG_BB is better for about 69% test problems with the least gradient evaluations, while the percentage of L-BFGS+(Wolfe+AWolfe) is only about 38%. As we

know, the gradient evaluation generally requires much more computational cost compared with the function evaluation. In Fig.16, we can observe that LMSMCG_BB is much faster than L-BFGS+(Wolfe+AWolfe), one of the most efficient optimization methods for large unconstrained optimization.

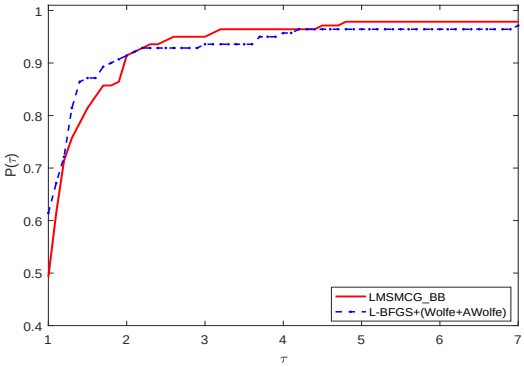

Fig. 13 Performance profile based on N_{iter}

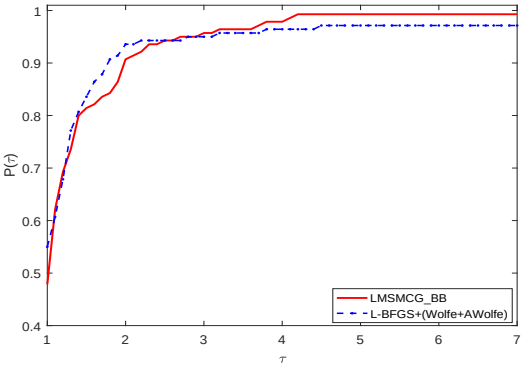

Fig. 14 Performance profile based on N_f

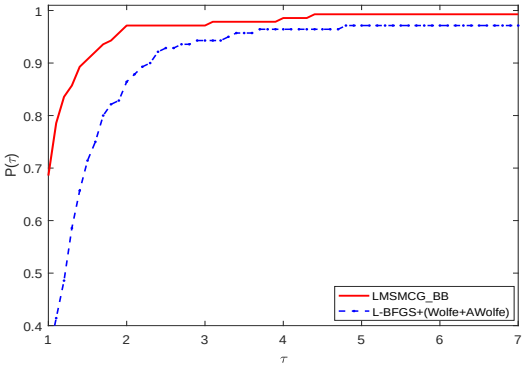

Fig. 15 Performance profile based on N_g

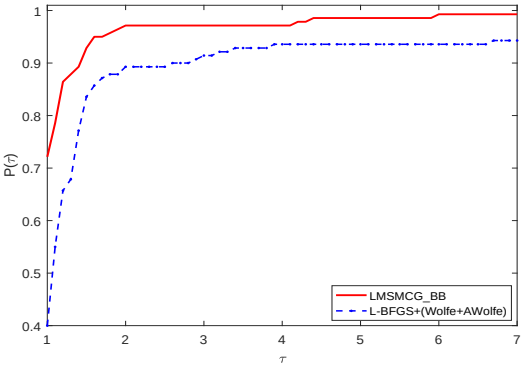

Fig. 16 Performance profile based on T_{cpu}

5 Conclusions

We present a limited memory subspace minimization conjugate gradient method (LMSMCG_BB) for unconstrained optimization in this paper. LMSMCG_BB only includes two types of iterations, which is different from the limited memory CG method [18] that uses three preconditioners (8). Unlike the limited memory CG method [18], the convergence of LMSMCG_BB with the generalize Wolfe line search used is established under the mild assumption without the strict assumption (7). The numerical results indicate that, for the CUTEr library, LMSMCG_BB has a quit great improvement over SMCG_BB, is superior to the latest limited memory CG software package CG_DESCENT (6.8) with the Wolfe line search, is also comparable to CG_DESCENT (6.8) with the efficient the hybridization of the Wolfe line search and AWolfe line search, and is superior to L-BFGS method with the hybridization of the Wolfe line search and AWolfe line search, one of the most efficient optimization methods for large unconstrained optimization.

Acknowledgements We would like to thank Professors Hager, W.W. and Zhang, H. C. for their C code of CG_DESCENT (6.8). This second author's work was partly supported by the Chinese NSF grants (Nos. 11631013,11331012) and Key Project of Chinese National Programs for Fundamental Research and Development (No. 2015CB856002). The first author's work was supported by the National Natural Science Foundation of China (No.11901561) and the Natural Science Foundation of Guangxi (No.2018GXNSFBA281180).

References

1. Fletcher, R., Reeves,C.: Function minimization by conjugate gradients. Comput. J., **7(2)**, 149-154(1964)
2. Hestenes, M. R., Stiefel,E. L. Methods of conjugate gradients for solving linear systems. J. Res. Nat. Bur. Stands., **49(6)**, 409-436(1952)
3. Polyak, B. T.: The conjugate gradient method in extreme problems. USSR Comp. Math. Math. Phys., **9**, 94-112(1969)
4. Dai, Y. H., Yuan, Y. X.: A nonlinear conjugate gradient method with a strong global convergence property. SIAM J. Optim., **10(1),** 177-182(1999)
5. Yuan Y.X., Stoer J.: A subspace study on conjugate gradient algorithms. Z. Angew. Math. Mech., **75(1)**, 69-77(1995)
6. Andrei, N.: An accelerated subspace minimization three-term conjugate gradient algorithm for unconstrained optimization. Numer. Algorithms, **65(4)**, 859-874(2014)
7. Yang, Y. T., Chen, Y. T., Lu, Y. L.: A subspace conjugate gradient algorithm for large-scale unconstrained optimization. Numer. Algorithms, **76(3)**, 813-828(2017)
8. Barzilai, J., Borwein, J.M.: Two-point step size gradient methods. IMA J. Numer. Anal., **8(1)**, 141-148(1988)
9. Dai, Y. H, Kou, C. X.: A Barzilai-Borwein conjugate gradient method. Sci. China. Math., **59(8)**, 1511-1524(2016)
10. Liu, H. W., Liu, Z. X. An efficient Barzilai-Borwein conjugate gradient method for unconstrained optimization. J. Optim. Theory Appl., **180(3)**, 879-906(2018)
11. Dai, Y. H., Kou, C. X.: A nonlinear conjugate gradient algorithm with an optimal property and an improved Wolfe line search. SIAM J. Optim., **23(1)**, 296-320(2013)
12. Hager, W. W., Zhang, H. C.: A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J. Optim., **16(1)**, 170-192(2005)
13. Li M, Liu H W, Liu Z X.: A new subspace minimization conjugate gradient method with nonmonotone line search for unconstrained optimization. Numer. Algorithms, **79(1)**, 195-219(2018)
14. Wang T, Liu Z X, Liu H W.: A new subspace minimization conjugate gradient method based on tensor model for unconstrained optimization. Int. Comput, Math., **96(10)**, 1924-1942(2019)
15. Li, Y F, Liu Z X, Liu H W.: A subspace minimization conjugate gradient method based on conic model for unconstrained optimization. Comut. Appl. Math., **83(1)**, (2019) https://doi.org/10.1007/s40314-019-0779-7
16. Zhang K K, Liu H W, Liu Z X.: A new adaptive subspace minimization three-term conjugate gradient algorithm for unconstrained optimization. Journal of Computational Mathematics, **39(2)**, 159-177(2021).
17. Zhao T., Liu H W, Liu Z X. New subspace minimization conjugate gradient methods based on regularization model for unconstrained optimization. Numerical Algorithms. DOI: 10.1007/s11075-020-01017-1.
18. Hager, W.W., Zhang, H. C.: The limited memory conjugate gradient method. SIAM J. Optim., **23(4),** 2150-2168(2013)
19. Gilbert, J. C., Nocedal, J.: Global convergence properties of conjugate gradient methods for optimization, SIAM J. Optim., **2**, 21-42(1992)
20. Liu, D. C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Math. Program., **45(1-3)**, 503-528(1989)
21. Gould, N. I. M., Orban, D., Toint, Ph.L.: CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. ACM Trans. Math. Softw., **29(4)**, 373-394(2003)
22. Hager, W. W., Zhang, H. C.: Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent. ACM Trans. Math. Softw., **32(1)**, 113-137(2006)

23. Yuan, Y. X., Sun, W. Y.: Theory and methods of optimization. Science Press of China (1999)
24. Li, D. H., Fukushima, M.: On the global convergence of BFGS method for nonconvex unconstrained optimization problems. SIAM J. Optim., **11(4)**, 1054-1064(2001)
25. Yuan Y. X.: A modified BFGS algorithm for unconstrained optimization. IMA J. Numer. Anal., **11(3)**, 325-332(1991)
26. Liu, Z. X., Liu, H. W.: An efficient gradient method with approximate optimal stepsize for large-scale unconstrained optimization. Numer. Algorithms, **78(1)**, 21-39(2018)
27. Liu, Z. X., Liu, H. W.: An efficient gradient method with approximately optimal stepsize based on tensor model for unconstrained optimization. J. Optim. Theory Appl., **181(2)**, 608-633(2019)
28. Andrei, N.: Open problems in nonlinear conjugate gradient algorithms for unconstrained optimization. Bull. Malays. Math. Sci. Soc., **34(2)**, 319-330(2011)
29. Hager, W.W., Zhang, H. C.: Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent. Acm Transactions on Mathematical Software, **32(1)**, 113-137(2006).
30. Zhou, B., Gao, L., Dai, Y. H.: Gradient methods with adaptive stepsizes. Comput. Optim. Appl., **35(1)**, 69-86(2006)
31. Dai, Y. H., Yuan,Y.X.: Nonlinear Conjugate Gradient Methods. Shanghai Scientific and Technical Publishers, Shanghai(2000)
32. Dolan, E. D., Moré, J. J.: Benchmarking optimization software with performance profiles. Math. Program., **91(2)**, 201-213(2002)