Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2014, Article ID 236158, 9 pages
http://dx.doi.org/10.1155/2014/236158

Research Article

Sufficient Descent Polak-Ribière-Polyak Conjugate Gradient Algorithm for Large-Scale Box-Constrained Optimization

Qiuyu Wang[1] and Yingtao Che[2]

[1] *College of Software, Henan University, Kaifeng 475000, China*
[2] *College of Mathematics and Information Science, Henan University, Kaifeng 475000, China*

Correspondence should be addressed to Qiuyu Wang; wqy@henu.edu.cn

Received 7 December 2013; Accepted 14 February 2014; Published 13 April 2014

Academic Editor: Gaohang Yu

A practical algorithm for solving large-scale box-constrained optimization problems is developed, analyzed, and tested. In the proposed algorithm, an identification strategy is involved to estimate the active set at per-iteration. The components of inactive variables are determined by the steepest descent method at first finite number of steps and then by conjugate gradient method subsequently. Under some appropriate conditions, we show that the algorithm converges globally. Numerical experiments and comparisons by using some box-constrained problems from CUTEr library are reported. Numerical comparisons illustrate that the proposed method is promising and competitive with the well-known method—L-BFGS-B.

1. Introduction

We consider the box-constrained optimization problem

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

where $f:\mathbb{R}^n \to \mathbb{R}$ is continuously differentiable, and $l, u \in \mathbb{R}^n$ with $l < u$. The gradient of f at x^k is $\nabla f(x^k) \in \mathbb{R}^n$, and its ith component is $\nabla f_i(x^k)$ $(\in \mathbb{R})$ or g_i^k for the sake of simplicity. We define the feasible region of (1) as Ω; that is,

$$\Omega = \{x \in \mathbb{R}^n : l \le x \le u\}. \tag{2}$$

We say that a vector $\overline{x} \in \Omega$ is a stationary point for problem (1) if it satisfies

$$\begin{aligned} l_i = \overline{x}_i &\Longrightarrow \nabla f_i(\overline{x}) \ge 0, \\ l_i < \overline{x}_i < u_i &\Longrightarrow \nabla f_i(\overline{x}) = 0, \\ \overline{x}_i = u_i &\Longrightarrow \nabla f_i(\overline{x}) \le 0. \end{aligned} \tag{3}$$

Problem (1) is very important in practical optimization, because numerous practical problems can be converted into this form. In addition, problem (1) is often treated as a subproblem of augmented Lagrangian or penalty schemes for general constrained optimization. Hence, the development of numerical algorithms to efficiently solve (1), especially when the dimension is large, is important both in theory and application. Moreover, the box-constrained optimization problems have been received much attention in recent decades. We refer to the excellent paper [1] for a good review.

Many algorithms for solving this type of problems are based on active set strategies, for example, [2]. In this class of methods, a working set is used to estimate the set of active constraints at the solution and it is updated at per-iteration. The early active set methods are quite efficient for relatively lower dimensional problems but are typically unattractive for large-scale ones [3]. The main reason is that at most one constraint can be added to or dropped from the active set at each step. However, this slows down the rate of the convergence. Based on this consideration, more and more scholars are engaged in designing active set methods aiming to make rapid changes against/from incorrect predictions, for example, [3–8].

Let $\Omega = \mathbb{R}^n$. It is clear that problem (1) reduces to the following unconstrained optimization problem:

$$\min_{x\in\mathbb{R}^n} f(x). \tag{4}$$

It can be solved by almost any existing effective methods such as the nonlinear conjugate gradient method, limited memory BFGS method, and spectral gradient method. These methods are much suitable for solving (4) with n being assumed to be large due to their simplicity and low storage requirements.

In conjugate gradient methods scheme, Polak-Ribière-Polyak (PRP) method is generally believed to be the most efficient from computation point of view. But its global convergence for nonconvex nonlinear function is still uncertain [9]. To overcome this shortcoming, various techniques have been proposed. In particular, Zhang et al. [10] introduced a modified MPRP method in which the generated direction is descent independent of line search rule [11]. Global convergence is guaranteed by means of a variation of the Armijo-like line search strategy of Grippo and Lucidi [12]. Moreover, MPRP is also considered to be competent to solve the box-constrained optimization problem (1) based on projected gradient techniques [13]. However, to the best of our knowledge, papers on nonlinear conjugate gradient for solving box-constrained minimization problems are relatively fewer.

In this paper, from a different point of view, we further study the applications of the MPRP method in [10] for solving box-constrained minimization problems. First, motivated by the work of Facchinei et al. [3], we estimate a lower dimensional free subspace at per-iteration, which has the potencies of being numerically cheaper when applied to solve large-scale problems. The search direction at per-iteration consists of two parts: some of the components are defined simply; the others are determined by the nonlinear gradient method. The remarkable feather of the proposed method is that all the generated points are feasible without requiring gradient projection. Under some appropriate conditions, we show that the method converges globally. We present experimental results comparing the developed method with the well-known algorithm L-BFGS-B.

The paper is organized as follows. For easy comprehension of the proposed algorithm, we briefly recall an active set identification technique and then state the steps of our new algorithm in Section 2. In Section 3, we show that the proposed method converges globally. In Section 4, we test the performance of the algorithm and do some numerical comparisons. Finally, we draw some conclusions in Section 5. In the rest of this paper, the symbol $\|\cdot\|$ denotes the Euclidean norm of vectors.

2. Motivation and Algorithm

In this section, we introduce a new algorithm which is defined by

$$x^{k+1} = x^k + \alpha_k d^k, \tag{5}$$

where d^k is a search direction and α_k is the corresponding step length. To construct our algorithm, throughout this paper, we assume that the following assumptions hold.

Assumption 1. The level set $\mathcal{F} = \{x \in \mathbb{R}^n : f(x) \le f(x^0)\} \cap \Omega$ is compact.

Assumption 2. In some neighborhood $\mathcal{N}$ of $\mathcal{F}$, the gradient $\nabla f(x)$ is Lipschitz continuous; that is, there exists a constant $\mathcal{L} > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \le \mathcal{L} \|x - y\|, \quad \forall x, y \in \mathcal{N}. \tag{6}$$

Assumption 3. Strict complementarity condition holds at $\overline{x}$; that is, the strict inequalities hold in the first and the last implications of (3).

In [3], Facchinei et al. presented an active set Newton algorithm for solving problem (1). Their algorithm possesses some favorable properties, such as fast local convergence and feasibility of all iterations. In addition, only a lower dimensional quadratic program subproblem needs to be solved at per-iteration. We now simply recall the active set identification technique in [3]. Let $a_i(x)$ and $b_i(x)$ be nonnegative continuous and bounded functions defined on Ω, such that if $x_i = l_i$ or $x_i = u_i$, then $a_i(x) > 0$ or $b_i(x) > 0$, respectively. For any $x \in \Omega$, define the index set $L(x)$, $F(x)$ and $U(x)$ as

$$\begin{aligned} L(x) &= \{i : x_i \le l_i + a_i(x) \nabla f_i(x)\}, \\ U(x) &= \{i : x_i \ge u_i + b_i(x) \nabla f_i(x)\}, \\ F(x) &= \{1, \ldots, n\} \setminus (L \cup U). \end{aligned} \tag{7}$$

The set $L(x) \cup U(x)$ is an estimate of the active set at point x. Facchinei et al. [3] showed that with Assumption 3, when x is sufficiently close to $\overline{x}$, the index set $L(x) \cup U(x)$ is an exact identification of $L(\overline{x}) \cup U(\overline{x})$. This property will enable us to find the active set at the stationary point $\overline{x}$ in a finite number of steps. This active set identification technique (7) has motivated much additional studies on box-constrained optimization problems, for example, [14–18].

Using the active set estimation technique in (7), we deduce the search direction used in our algorithm in detail now. For the sake of simplicity, let $L^k = L(x^k)$, $U^k = U(x^k)$, $F^k = F(x^k)$, and let $|\cdot|$ be the number of elements in the corresponding vector let Z_k be the matrix whose columns are $\{e_i : i \in F^k\}$, where e_i is the ith column of the identity matrix. Additionally, we denote, respectively, the ith component of x^k and d^k by x_i^k and d_i^k.

Let $d^k \in \mathbb{R}^n$, in which

$$\begin{aligned} d_i^k &= l_i - x_i^k, \quad i \in L^k, \\ d_i^k &= u_i - x_i^k, \quad i \in U^k. \end{aligned} \tag{8}$$

Now, we restrict our attention to define components with index $i \in F^k$. Let

$$\tilde{d}^k = \begin{cases} -\tilde{g}^0, & \text{if } k = 0, \\ -\tilde{g}^k, & \text{if } k > 0,\ F^k \ne F^{k-1}, \\ -\tilde{g}^k + \beta_k \tilde{d}^{k-1} - \theta_k \tilde{y}^{k-1}, & \text{if } k > 0,\ F^k = F^{k-1}, \end{cases} \tag{9}$$

where

$$\beta_k = \frac{\left(\tilde{g}^k\right)^\top \tilde{y}^{k-1}}{\|\tilde{g}^{k-1}\|^2}, \qquad \theta_k = \frac{\left(\tilde{g}^k\right)^\top \tilde{d}^{k-1}}{\|\tilde{g}^{k-1}\|^2}, \tag{10}$$

in which $\tilde{g}^k = Z_k^\top g^k$, $\tilde{g}^{k-1} = Z_k^\top g^{k-1}$, and $\tilde{y}^{k-1} = \tilde{g}^k - \tilde{g}^{k-1}$. Clearly, $\tilde{d}^k \in \mathbb{R}^{|F^k|}$ and therefore $Z_k \tilde{d}^k \in \mathbb{R}^n$. It is not difficult to see that the definition of the search direction $\tilde{d}^k$ ensures the sufficient descent property; that is

$$\left(\tilde{g}^k\right)^\top \tilde{d}^k = -\left\|\tilde{g}^k\right\|^2. \tag{11}$$

The definition of the search direction (9) was designed by Zhang et al. [10] in full space $\mathbb{R}^n$. Here, it is stated for slightly different circumstances, but it is easy to verify that it is still valid in our context.

In order to guarantee the feasibility of all iterations and the descent of the objective function at each iteration, we define the positive scalar ξ_k^* by

$$\xi_k^* = \max\left\{\xi \mid \xi \le 1, l_i - x_i^k \le \xi\left(Z_k \tilde{d}^k\right)_i \le u_i - x_i^k, i \in F^k\right\}. \tag{12}$$

Note that when the strict complementarity condition holds and $l < x < u$, then ξ_k^* is always well defined unless the stationary point is achieved. Combining with (8) and (9), the compact form of d^k can be rewritten as

$$d_i^k = \begin{cases} l_i - x_i^k, & i \in L^k, \\ u_i - x_i^k, & i \in U^k, \\ \xi_k^*\left(Z_k \tilde{d}^k\right)_i, & i \in F^k. \end{cases} \tag{13}$$

Remark 4. Observe that the search direction of inactive variables in (9) contain both gradient and direction information at the current or the previous steps, while the set of inactive variables might be changed as the iteration progresses. In particular, the sets F^{k-1} and F^k might not be the same. In this situation, we set $\tilde{d}^{k-1} = -\tilde{g}^{k-1}$, that is, a negative gradient direction. Otherwise, we use the conjugate gradient formula to update the search direction in the free subspace.

Remark 5. Considering the case when the current iteration x^k is still not an optimal point, the value of $\tilde{g}^k$ might be sufficiently small. In this case, for practical purpose, it might be efficient to restrict $\|\tilde{g}^k\|$ bounded from zero.

To list our algorithm in detail, we first show some useful properties of the direction. The following result shows that whenever $d^k \ne 0$, it is a descent direction for objective function f at current point x^k. The property is very important to our algorithm.

Lemma 6. *If d^k is attained by* (13)*, it satisfies*

$$\left(g^k\right)^\top d^k \le 0, \tag{14}$$

and the equality holds if and only if $d^k = 0$.

Proof. By the definition of search direction d^k, we have

$$\begin{aligned} \left(d^k\right)^\top g^k &= \sum_{i \in L^k}\left(l_i - x_i^k\right) g_i^k \\ &\quad + \sum_{i \in U^k}\left(u_i - x_i^k\right) g_i^k - \left\|Z_k^\top g^k\right\|^2 \le 0. \end{aligned} \tag{15}$$

The above relation can be easily obtained from the definition of the direction d^k in (13). Notice that the strict complementary condition holds, which shows $(d^k)^\top g^k = 0$ if and only if $d^k = 0$. □

As an immediate consequence of the definition of the search direction, we have the following fact.

Lemma 7. *Let d^k be determined by* (13) *and assume that $d^k \ne 0$; then,*

$$x^k + d^k \in \Omega. \tag{16}$$

The above lemma shows that for all $0 \le \alpha \le 1$ such that

$$P_\Omega\left(x^k + \alpha d^k\right) = x^k + \alpha d^k. \tag{17}$$

Since the search direction d^k has been computed, a step length $\alpha_k \in (0, 1]$ ought to be determined in order to obtain the next iteration as (5). In this paper, we consider a backtracking line search procedure, where a decreasing sequence of $\{\rho^j\}$ is tried until the smallest j_k is found which satisfies

$$f\left(x^k + \rho^j d^k\right) \le f\left(x^k\right) - \delta\left\|\rho^j d^k\right\|^2, \tag{18}$$

where $\rho \in (0, 1)$, $\delta \in (0, 1/2)$. Then a step length is accepted corresponding to $\alpha^k = \rho^{j_k}$.

We are now ready to formally state the overall algorithm for solving the box-constrained optimization problems (1) as follows.

Algorithm 8 (SDPRP). The steps of the algorithm is given as follows.

Step 0. Given starting point $x^0 \in \Omega$, constants $\delta \in (0, 1/2)$, $\rho \in (0, 1)$ and $g_{\max} > g_{\min} > 0$. Set $k := 0$.

Step 1. Determine L^k, U^k, and F^k according to (7).

Step 2. If $F^k \ne F^{k-1}$, set $\tilde{d}^k = -\tilde{g}^k$. Otherwise, set

$$\left\|\tilde{g}^{k-1}\right\|^2 = \min\left\{g_{\max}, \max\left\{g_{\min}, \left\|\tilde{g}^{k-1}\right\|^2\right\}\right\}. \tag{19}$$

Determine d^k by (13).

Step 3. If $d^k = 0$, then stop.

Step 4. Find the smallest integer j (say j_k) satisfying (18). Let $\alpha_k = \rho^{j_k}$.

Step 5. Set $x^{k+1} := x^k + \alpha_k d^k$.

Step 6. Let $k := k + 1$, go to Step 1.

Remark 9. Now suppose that the active sets have been identified after a finite number of steps; then, we have $d_i^k = 0$ for all $i \in L^k \cup U^k$ and $d_i^k = \xi_k^*(Z_k \tilde{d}^k)_i$ for all $i \in F^k$. If $\alpha_k \ne 1$, by the line search process, we know that $\alpha_k' = \rho^{-1}\alpha_k$ does not satisfy (18). That is,

$$f\left(x^k + \rho^{-1}\alpha_k d^k\right) - f\left(x^k\right) > -\delta\rho^{-2}\alpha_k^2\left\|d^k\right\|^2. \quad (20)$$

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

$$\begin{aligned} & f\left(x^k + \rho^{-1}\alpha_k d^k\right) - f\left(x^k\right) \\ &= \rho^{-1}\alpha_k g\left(x^k + \zeta_k\rho^{-1}\alpha_k d^k\right)^\top d^k \\ &= \rho^{-1}\alpha_k\left(g^k\right)^\top d^k \\ &\quad + \rho^{-1}\alpha_k\left[g\left(x^k + \zeta_k\rho^{-1}\alpha_k d^k\right) - g^k\right]^\top d^k \\ &\le \rho^{-1}\alpha_k\left(g^k\right)^\top d^k + \mathscr{L}\rho^{-2}\alpha_k^2\left\|d^k\right\|^2, \end{aligned} \quad (21)$$

where $\mathscr{L}$ is the Lipschitz constant. It follows from [10, Lemma 3.1] that there exists a positive constant M such that $\|d^k\| \le M$. Substituting the above inequality into (20), we have

$$\alpha_k \ge \frac{\rho}{\mathscr{L}+\delta}\frac{\left(g^k\right)^\top d^k}{\left\|d^k\right\|^2} = \frac{\rho}{\mathscr{L}+\delta}\frac{\left\|\bar{g}^k\right\|^2}{\left\|\bar{d}^k\right\|^2} \ge \frac{\rho}{M(\mathscr{L}+\delta)}\left\|\bar{g}^k\right\|^2. \quad (22)$$

It follows that when $\|\bar{g}^k\|^2 \neq 0$, the line search step is well defined, so is the whole algorithm.

3. Convergence Analysis

In this section, we show that Algorithm 8. converges globally. The following lemma gives a sufficient and necessary condition for the global convergence of Algorithm 8.

Lemma 10. *Let $\{x^k\}$ be a sequence of iterations generated by Algorithm 8, and let d^k be a search direction defined by* (13). *Then x^k is a stationary point of* (1) *if and only if $d^k = 0$.*

Proof. Let $d^k = 0$. If $i \in L^k$, then we have

$$0 = d_i^k = l_i - x_i^k \ge -a_i\left(x^k\right)\nabla f_i\left(x^k\right). \quad (23)$$

Since $x_i^k = l_i$ and $a_i(x^k) > 0$, the right inequality implies $\nabla f_i(x^k) \ge 0$. If $\in U^k$, we will prove it similarly. And if $\in F^k$, we can get from (11) that $\nabla f_i(x^k) = 0$. Now suppose that x^k is a stationary point of (1); it gets from (3) and (7) that

$$\begin{aligned} L^k &= \left\{i : x_i^k = l_i\right\}, \\ F^k &= \left\{i : l_i < x_i^k < u_i\right\}, \\ U^k &= \left\{i : x_i^k = u_i\right\}. \end{aligned} \quad (24)$$

Then it follows from (9) and (11) that $d_{L^k}^k = 0$, $d_{U^k}^k = 0$ and $d_{F^k}^k = 0$. Therefore $d^k = 0$. □

Lemma 11. *Let $\{x^k\}$ be a sequence of iterations generated by Algorithm 8 and let d^k be a search direction defined by* (13). *Suppose that there are subsequences $\{x^k\}_{\mathscr{K}} \to \bar{x}$ and $\{d^k\}_{\mathscr{K}} \to 0$ as $k \to \infty$. Then $\bar{x}$ is a stationary point of problem* (1).

Proof. Since the number of distinct sets L^k, U^k, and F^k is finite and Assumptions 1–3 hold, Assumption 3 ensures that there exists k_0 such that for all $k > k_0$ [3],

$$L^k = \bar{L}, \qquad U^k = \bar{U}, \qquad F^k = \bar{F}. \quad (25)$$

Since $\{x^k\} \subseteq \mathscr{F}$, we obviously have

$$l \le \bar{x} \le u. \quad (26)$$

Furthermore, the fact that $\{d_L^k\}_{\mathscr{K}} \to 0$ and $\{d_U^k\}_{\mathscr{K}} \to 0$ implies

$$\begin{aligned} \bar{x}_L &= l_L, \qquad \nabla f_L(\bar{x}) \ge 0, \\ \bar{x}_U &= u_U, \qquad \nabla f_U(\bar{x}) \le 0. \end{aligned} \quad (27)$$

The proof of $\nabla f_{\bar{F}}(\bar{x}) = 0$ can be obtained once we notice that (11) and $d_{F^k}^k \to 0$ as $k \to \infty$. □

The similar proof of Lemmas 10 and 11 can also be found in [3]. To end of this section, now we are ready to establish global convergence of Algorithm 8.

Theorem 12. *Suppose that Assumptions 1–3 hold. Then the sequence $\{x^k\}$ generated by Algorithm 8 has at least a limit point, and every limit point of this sequence is a stationary point of problem* (1).

Proof. It easily follows from Lemma 7 that the sequence $\{x^k\}$ of points generated by the Algorithm 8 is contained in the compact set $\mathscr{F}$. Hence, by Assumption 1, there exists at least a limit point of this sequence.

If the sequence $\{x^k\}$ is finite with last point $\bar{x}$, then, by Lemma 10, $\bar{x}$ is a stationary point of problem (1). So we assume that the sequence is infinite.

Regarding the first part of proof in Lemma 11, L^k, U^k, and F^k are constants starting from some iteration index k_0; that is,

$$L^k = \bar{L}, \qquad U^k = \bar{U}, \qquad F^k = \bar{F}. \quad (28)$$

It is easy to see that

$$\begin{aligned} d_i^k &= l_i^k - x_i^k = 0, \quad \forall i \in L^k, \\ d_i^k &= u_i^k - x_i^k = 0, \quad \forall i \in U^k, \end{aligned} \quad (29)$$

as $k \to \infty$. Now, it remains to prove $d_i^k \to 0$ $(i \in F^k)$ as $k \to \infty$. According to the global convergence theorem in [10], we have

$$\liminf_{k\to\infty}\left\|\bar{g}_i^k\right\| = 0, \quad \forall i \in F^k. \quad (30)$$

By the definition of $\widetilde{d}_k$ in (9), it is equivalent to

$$\liminf_{k\to\infty} \left\| \widetilde{d}_i^k \right\| = 0, \quad \forall i \in F^k. \tag{31}$$

It follows from (29) and (31) that $\{d^k\}_{k\in\mathcal{K}} \to 0$, which shows our claims by Lemma 11. The proof is complete. □

4. Numerical Experiments

Now, let us report some numerical results attained by our Sufficient Descent Polak-Ribière-Polyak Algorithm—SDPRP. The algorithm is implemented by Fortran77 code in double precision arithmetic. All experiments are run on a PC with CPU Intel Pentium Dual E2140 1.6 GHz, 512 M bytes of SDRAM memory, and Red Hat Linux 9.03 operating system. Our experiments are performed on a set of the nonlinear box-constrained problems from the CUTEr [19] library that have second derivatives available. Since we are interested in large problems, we refine this selection by considering only problems where the number of variables is at least 50. Altogether, we solve 54 problems. The type of objective function and the character of the problems being tested are listed in Table 1.

Table 1: Problem set according to the CUTEr classification.

Number of sets	Objective type	Problem character	Classification
1	Others	Academic	OBR2-AN-∗-∗
2	Others	Modeling	OBR2-MN-∗-∗
3	Others	Real application	OBR2-RN-∗-∗
4	Sum of squares	Academic	SBR2-AN-∗-∗
5	Sum of squares	Modeling	SBR2-AN-∗-∗
6	Quadratic	Academic	QBR2-AN-∗-∗
7	Quadratic	Modeling	QBR2-MN-∗-∗
8	Quadratic	Real application	QBR2-RN-∗-∗
9	Others	Modeling	OBI2-MY-∗-∗
10	Quadratic	Academic	QBR2-AY-∗-∗

In the experiments, for easily comparing with other codes, we use the projected gradient errors to measure the quality of the solutions instead of $\|d^k\|$; that is, we force the iteration stopped when

$$\left\| P_\Omega(x^k - \nabla f(x^k)) - x^k \right\|_\infty \le 10^{-5}, \tag{32}$$

where $P_\Omega(\cdot)$ is the projected gradient of objective function and $\|\cdot\|_\infty$ denotes the maximum absolute component of a vector. Clearly, the stopping criteria (32) are equivalent to $\|\nabla f(x^k)\|_\infty \le 10^{-5}$ in our method. We also stop the execution of SDPRP when 10000 iterations or 20000 function evaluations are completed without achieving convergence. We choose the initial parameters $g_{\min} = 10^{-7}$, $g_{\max} = 10^{20}$, $\rho = 0.29$, and $\delta = 10^{-1}$. Moreover, we also test our proposed methods with different parameters $a_i(x)$ and $b_i(x)$ to see that $a_i(x) = b_i(x) = 10^{-6}\|P_\Omega(\nabla f(x^0)\|$ is the best choice. In order to assess the performance of the SDPRP, we then test the well-known method L-BFGS-B (available at http://www.ece.northwestern.edu/~nocedal/lbfgsb.html) [20, 21] for comparison.

When running the codes L-BFGS-B, default values are used for all parameters. For L-BFGS-B, besides (32), the algorithm has another built-in stopping test based on the parameter factr. It is designed to terminate the run when the change in the objective function f is sufficiently small. The default value factr = 1.0d + 7 for moderate accuracy. Additionally, it can occur sometimes that the line search cannot make any progress, or the value of Nfg reached a prefixed number (=10000); in these cases, the run is also terminated.

The numerical results of the algorithms SDPRP and L-BFGS-B are listed in Tables 2 and 3, respectively. We used horizontal lines in both tables to divide the selected problems into 10 classes according to Table 1. The columns in Tables 2 and 3 have the following meanings:

Problem: name of the problem;

Dim: dimension of the problem;

Iter: number of iterations;

Nf: number of function evaluations;

Ng: number of gradient evaluations;

Time: CPU time in seconds;

Fv: final function value;

Pgnorm1: maximum-norm of the final gradient projection.

The symbol "—" indicates that the CUTEr system becomes nonrespondent when calculating the corresponding problem. For L-BFGS-B, we record the number of function and gradient evaluations Nfg because L-BFGS-B always evaluates the function and gradient at the same time.

Some general observations on the results in Tables 2 and 3 are the following. From the last column of Table 3, we see that, for 41 problems, L-BFGS-B has terminated abnormally without being able to satisfy the termination condition (32). SDPRP fails to reach a stationary point based on the stopping criteria (32) only on the 12 problems: BDEXP, EXPQUAD, SOND1LS, QR3DLS, BIGGS1, BQPGAUSS, CHENHARK, JNLBRNGB, NCVXBQP1, ODNAMUR, NCVXBQP2, GRIDGENA. It is an interesting fact that L-BFGS-B obtains at least as good function value as SDPRP on some problems but the projected gradient did not meet the stooping condition. This reason is not clear as pointed out by Zhu et al. in [21]. In analyzing the performance of both methods on the 13 problems at which at least one method works successfully, we observe that, on these problems, L-BFGS-B requires less iterations, less function evaluations than SDPRP. The sterical data of both algorithms are summarized in Table 4, where "Scueed" is the number of successes and "Failure" is the number of failures of tested problems for both algorithms.

Taking everything together, the preliminary numerical comparisons indicate that our proposed method is efficient and competitive with the well-known method L-BFGS-B.

Table 2: Performance of SDPRP.

Problem	Dim	Iter	Nf	Time	Fv	Pgnorm1
SINEALI	1000	19	64	0.030	$-0.99860E+05$	$0.54718E-05$
BDEXP	1000	10001	10003	6.030	$0.92538E-03$	$0.13189E-04$
EXPLIN	120	76	260	0.010	$-0.72376E+06$	$0.95740E-05$
EXPLIN2	120	59	183	0.000	$-0.72446E+06$	$0.50532E-05$
EXPQUAD	120	4007	20001	0.230	$-0.36260E+07$	$0.28861E-03$
MCCORMCK	5000	32	72	0.130	$-0.45666E+04$	$0.69662E-05$
PROBPENL	500	18	173	0.010	$0.39919E-06$	$0.71380E-05$
QRTQUAD	120	949	4240	0.030	$-0.36246E+07$	$0.89349E-05$
S368	100	54	136	0.240	$-0.14025E+03$	$0.42537E-05$
HADAMALS	1024	113	479	0.390	$0.30658E+05$	$0.31465E-05$
SCOND1LS	5002	5061	20001	81.330	$0.11463E+03$	$0.10058E+00$
CHEBYQAD	50	735	4514	6.440	$0.53863E-02$	$0.98736E-05$
HS110	200	1	2	0.000	$-0.99601E+40$	$0.00000E+00$
LINVERSE	1999	1527	1565	2.220	$0.68200E+03$	$0.89735E-05$
NONSCOMP	5000	67	128	0.110	$0.23525E-10$	$0.42102E-05$
QR3DLS	610	2760	20000	5.530	$0.17615E+00$	$0.29073E-01$
DECONVB	61	3353	11830	0.260	$0.35269E-07$	$0.95952E-05$
QUDLIN	5000	1	2	0.00	$-0.12500E+10$	$0.00000E+00$
BIGGSB1	5000	10001	11535	10.93	$0.21870E-01$	$0.74163E-04$
BQPGABIM	50	45	213	0.00	$-0.37903E-04$	$0.75229E-05$
BQPGASIM	50	57	223	0.00	$-0.55198E-04$	$0.62637E-05$
BQPGAUSS	2003	3945	20001	6.78	$-0.18077E+00$	$0.98485E-01$
CHENHARK	5000	6335	20001	9.56	$-0.19998E+01$	$0.36782E-04$
CVXBQP1	10000	14	15	0.05	$0.22502E+07$	$0.00000E+00$
HARKERP2	100	10	47	0.00	$-0.50000E+00$	$0.20897E-07$
JNLBRNG1	10000	4001	7427	38.40	$-0.18057E+00$	$0.96930E-05$
JNLBRNG2	10000	1911	6880	23.35	$-0.41486E+01$	$0.85547E-05$
JNLBRNGA	10000	2182	5955	20.17	$-0.27109E+00$	$0.94192E-05$
JNLBRNGB	10000	4981	20001	54.23	$-0.62068E+01$	$0.23366E-01$
NCVXBQP1	10000	1	2	0.01	$-0.19855E+11$	$0.27756E-16$
NCVXBQP2	10000	3180	20000	16.18	$-0.13340E+11$	$0.13402E+01$
NCVXBQP3	10000	3184	20000	17.11	$-0.65583E+10$	$0.92309E+01$
NOBNDTOR	5476	893	1462	3.54	$-0.44993E+00$	$0.51489E-05$
OBSTCLAE	10000	5261	5794	40.60	$0.18865E+01$	$0.97210E-05$
OBSTCLAL	10000	813	1121	6.29	$0.18865E+01$	$0.83060E-05$
OBSTCLBL	10000	3509	3872	26.90	$0.72722E+01$	$0.56157E-05$
OBSTCLBM	10000	3362	3730	25.97	$0.72722E+01$	$0.87301E-05$
OBSTCLBU	10000	2126	2454	16.71	$0.72722E+01$	$0.86796E-05$
PENTDI	1000	11	22	0.00	$-0.75000E+00$	$0.31759E-05$
TORSION1	10000	1550	2401	12.51	$-0.42726E+00$	$0.72463E-05$
TORSION2	10000	3227	4745	26.11	$-0.42725E+00$	$0.90789E-05$
TORSION3	10000	394	712	3.23	$-0.12138E+01$	$0.62159E-05$
TORSION4	10000	2183	2957	17.40	$-0.12138E+01$	$0.76428E-05$
TORSION5	10000	114	228	1.26	$-0.28604E+01$	$0.86795E-05$
TORSION6	10000	1639	1847	12.37	$-0.28604E+01$	$0.54347E-05$
TORSIONA	10000	1410	2273	13.26	$-0.41838E+00$	$0.97648E-05$
TORSIONB	10000	2662	4351	25.30	$-0.41838E+00$	$0.97029E-05$
TORSIONC	10000	433	747	4.41	$-0.12044E+01$	$0.65687E-05$
TORSIOND	10000	2269	3019	20.06	$-0.12044E+01$	$0.69877E-05$
TORSIONE	10000	97	194	0.95	$-0.28507E+01$	$0.68035E-05$
TORSIONF	10000	1589	1775	13.42	$-0.28507E+01$	$0.94166E-05$
ODNAMUR	11130	10001	16448	44.78	$0.11336E+05$	$0.21497E+02$
GRIDGENA	6218	620	20000	16.58	$-0.19243E+16$	$0.99274E+01$
NOBNDTOR	5476	893	1462	3.69	$-0.44993E+00$	$0.51489E-05$

Table 3: Performance of L-BFGS-B.

Problem	Dim	Nf	Ng	Time	Fv	Pgnorm1
SINEALI	1000	44	44	0.03	−0.9990E + 05	0.8973E − 03
BDEXP	1000	18	18	0.01	0.4808E − 03	0.6140E − 05
EXPLIN	120	40	40	0.00	−0.5150E + 08	0.3206E + 07
EXPLIN2	120	43	43	0.00	−0.7833E + 08	0.5921E + 07
EXPQUAD	120	8	8	0.00	−0.3163E + 07	0.1181E + 05
MCCORMCK	5000	10150	10150	40.29	−0.1221E + 12	0.1426E + 03
PROBPENL	500	4	4	0.00	0.3992E − 06	0.3071E − 06
QRTQUAD	120	13349	13349	0.63	−0.3561E + 08	0.3110E + 06
S368	100	21	21	0.07	−0.4084E + 02	0.2220E + 02
HADAMALS	1024	20	20	0.04	0.3165E + 05	0.4539E − 05
SCOND1LS	5002	25	25	0.15	0.6374E + 05	0.7813E + 03
CHEBYQAD	50	21	21	0.15	0.1395E − 01	0.1642E + 01
HS110	200	21	21	0.02	−0.1478E + 39	0.3285E + 37
LINVERSE	1999	195	195	0.32	0.6810E + 03	0.4605E − 04
NONSCOMP	5000	42	42	0.09	0.7536E − 11	0.6216E − 05
QR3DLS	610	11054	11054	10.24	0.4538E − 02	0.1914E − 01
DECONVB	61	291	291	0.02	0.5845E − 07	0.8405E − 05
QUDLIN	5000	314	314	0.16	−0.1315 + 301	0.1380 + 151
BIGGSB1	5000	4452	4452	8.36	0.1481E − 03	0.8917E − 05
BQPGABIM	50	41	41	0.00	−0.1839E − 03	0.7289E − 05
BQPGASIM	50	41	41	0.00	−0.1839E − 03	0.7289E − 05
BQPGAUSS	2003	9873	9873	11.62	−0.7222 + 301	0.1179 + 154
CHENHARK	5000	10323	10323	21.54	−0.8985E + 10	0.1722E + 03
CVXBQP1	10000	6798	6798	30.27	0.1339E − 07	0.5419E − 03
HARKERP2	100	11329	11329	1.26	−0.3318E + 02	0.3309E + 00
JNLBRNG1	10000	333	333	2.99	−0.1146E + 01	0.9604E − 05
JNLBRNG2	10000	998	998	9.02	−0.1065E + 03	0.9553E − 05
JNLBRNGA	10000	440	440	3.60	−0.2179E + 01	0.7975E − 05
JNLBRNGB	10000	6375	6375	52.41	−0.9552E + 03	0.9473E − 05
NCVXBQP1	10000	21	21	0.03	−0.4922E + 08	0.5250E + 05
NCVXBQP2	10000	21	21	0.03	−0.2812E + 08	0.3750E + 05
NCVXBQP3	10000	21	21	0.02	0.7034E + 07	0.3750E + 05
NOBNDTOR	5476	658	658	2.86	−0.3683E + 14	0.3672E + 00
OBSTCLAE	10000	2780	2780	23.21	−0.4897E + 12	0.1952E + 00
OBSTCLAL	10000	2274	2274	19.01	−0.4304E + 12	0.1289E + 00
OBSTCLBL	10000	4281	4281	35.78	−0.4744E + 12	0.1875E + 00
OBSTCLBM	10000	3998	3998	33.56	−0.5849E + 12	0.6435E − 01
OBSTCLBU	10000	1551	1551	13.03	−0.5328E + 11	0.4544E − 01
PENTDI	1000	14	14	0.01	−0.4267E + 02	0.2967E − 05
TORSION1	10000	811	811	6.80	−0.2733E + 14	0.2905E + 00
TORSION2	10000	1644	1644	13.76	−0.5354E + 13	0.1543E + 00
TORSION3	10000	712	712	5.98	−0.7571E + 11	0.4588E − 01
TORSION4	10000	2079	2079	17.47	−0.3908E + 14	0.7154E + 00
TORSION5	10000	744	744	6.20	−0.8637E + 13	0.1263E + 00
TORSION6	10000	730	730	6.07	−0.1361E + 15	0.5556E + 00
TORSIONA	10000	831	831	7.65	−0.3461E + 13	0.8291E − 01
TORSIONB	10000	1049	1049	9.61	−0.4526E + 10	0.3910E − 01
TORSIONC	10000	1165	1165	10.65	−0.6297E + 14	0.5303E + 00
TORSIOND	10000	980	980	9.01	−0.3573E + 14	0.4092E + 00
TORSIONE	10000	704	704	6.48	−0.2645E + 15	0.2775E + 01
TORSIONF	10000	646	646	5.91	−0.1976E + 15	0.1459E + 01
ODNAMUR	11130	10692	10692	57.16	0.8793E + 04	0.1247E + 01
GRIDGENA	6218	1145	1145	6.92	0.2352E + 05	0.9992E − 03
NOBNDTOR	5746	—	—	—	—	—

Table 4: Statical data of SDPRP and L-BFGS-B.

	Success	Failure
SDPRP	42	12
L-BFGS-B	13	41

Moreover, we conclude that the method provides a valid approach for solving large-scale box-constrained problems.

5. Conclusions

In this paper, we have developed a subspace nonlinear conjugate gradient method for solving box-constrained optimization problems. For most of the tested optimization problems (42 out of 54), the algorithm works successfully to terminate at the solution. However, in the most cases, the number of function evaluations seems large. A common feather of this method is that all the generated points are feasible, without requiring gradient projection as usual methods in this scheme. Therefore, this may be the reason why the function evaluation numbers are higher than those of projected gradient or trust-region methods. However, we also believe that SDPRP is a valid approach for box-constrained problems. In our view, there are at least three issues that could lead to improvements. The first point that should be considered is probably the choice of the parameters in the active set identification technique and the value of the used parameters is not the only choice. Another important point that should be further investigated is the adoption of gradient projection technique. The third assumption is strong; how can we modify this algorithm so as to avoid the strict complementary assumption? Additionally, it is worthwhile to investigate that some existing conjugate gradient methods, such as [12, 22], whether it is possible to embed an active set framework to solve box-constrained problems. To this end, although the proposed method does not obtain significant development as we have expected, we think that the enhancement of this proposed method is still noticeable. Hence, we believe that the proposed algorithm is a valid approach for the problems and possibility and it may have its own potency.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors thank the associate editor and two anonymous referees for their constructive suggestions which improved the paper greatly. This author's work is supported by Naturel Science Foundation of Henan Province Grants GGJS2011030 and 132300410168.

References

[1] N. Gould, D. Orban, and P. Toint, "Numerical methods for large-scale nonlinear optimization," *Acta Numerica*, vol. 14, pp. 299–361, 2005.

[2] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*, chapter 11, Addison-Wesley, Reading, Mass, USA, 1973.

[3] F. Facchinei, J. Júdice, and J. Soares, "An active set Newton algorithm for large-scale nonlinear programs with box constraints," *SIAM Journal on Optimization*, vol. 8, no. 1, pp. 158–186, 1998.

[4] E. G. Birgin and J. M. Martínez, "Large-scale active-set box-constrained optimization method with spectral projected gradients," *Computational Optimization and Applications*, vol. 23, no. 1, pp. 101–125, 2002.

[5] F. Facchinei, S. Lucidi, and L. Palagi, "A truncated Newton algorithm for large scale box constrained optimization," *SIAM Journal on Optimization*, vol. 12, no. 4, pp. 1100–1125, 2002.

[6] W. W. Hager and H. Zhang, "A new active set algorithm for box constrained optimization," *SIAM Journal on Optimization*, vol. 17, no. 2, pp. 526–557, 2006.

[7] Q. Ni and Y. Yuan, "A subspace limited memory quasi-Newton algorithm for large-scale nonlinear bound constrained optimization," *Mathematics of Computation*, vol. 66, no. 220, pp. 1509–1520, 1997.

[8] D. P. Bertsekas, "Projected Newton methods for optimization problems with simple constraints," *SIAM Journal on Control and Optimization*, vol. 20, no. 2, pp. 221–246, 1982.

[9] W. W. Hager and H. Zhang, "A survey of nonlinear conjugate gradient methods," *Pacific Journal of Optimization*, vol. 2, no. 1, pp. 35–58, 2006.

[10] L. Zhang, W. Zhou, and D.-H. Li, "A descent modified Polak-Ribière-Polyak conjugate gradient method and its global convergence," *IMA Journal of Numerical Analysis*, vol. 26, no. 4, pp. 629–640, 2006.

[11] J. Zhang, Y. Xiao, and Z. Wei, "Nonlinear conjugate gradient methods with sufficient descent condition for large-scale unconstrained optimization," *Mathematical Problems in Engineering*, vol. 2009, Article ID 243290, 16 pages, 2009.

[12] L. Grippo and S. Lucidi, "A globally convergent version of the Polak-Ribière conjugate gradient method," *Mathematical Programming*, vol. 78, no. 3, pp. 375–391, 1997.

[13] L. Zhang, *Nonlinear conjugate gradient methods for optimization problems [Ph.D. thesis]*, College of Mathematics and Econometrics, Hunan University, Hunan, China, 2006.

[14] L. Qi, X. J. Tong, and D. H. Li, "Active-set projected trust-region algorithm for box-constrained nonsmooth equations," *Journal of Optimization Theory and Applications*, vol. 120, no. 3, pp. 601–625, 2004.

[15] Y. Xiao and D.-H. Li, "An active set limited memory BFGS algorithm for large-scale bound constrained optimization," *Mathematical Methods of Operations Research*, vol. 67, no. 3, pp. 443–454, 2008.

[16] Y. Xiao and Q. Hu, "Subspace Barzilai-Borwein gradient method for large-scale bound constrained optimization," *Applied Mathematics and Optimization*, vol. 58, no. 2, pp. 275–290, 2008.

[17] Y.-H. Xiao, Q.-J. Hu, and Z. Wei, "Modified active set projected spectral gradient method for bound constrained optimization," *Applied Mathematical Modelling*, vol. 35, no. 7, pp. 3117–3127, 2011.

[18] Y.-F. Yang, D.-H. Li, and L. Qi, "A feasible sequential linear equation method for inequality constrained optimization," *SIAM Journal on Optimization*, vol. 13, no. 4, pp. 1222–1244, 2003.

[19] I. Bongartz, A. R. Conn, N. Gould, and P. L. Toint, "CUTE: constrained and unconstrained testing environment," *ACM Transactions on Mathematical Software*, vol. 21, no. 1, pp. 123–160, 1995.

[20] R. H. Byrd, P. Lu, J. Nocedal, and C. Y. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[21] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-BFGS-B: fortran subroutines for large-scale bound-constrained optimization," *ACM Transactions on Mathematical Software*, vol. 23, no. 4, pp. 550–560, 1997.

[22] J. C. Gilbert and J. Nocedal, "Global convergence properties of conjugate gradient methods for optimization," *SIAM Journal on Optimization*, vol. 2, no. 1, pp. 21–42, 1992.