Conjugate Gradient Method: A Developed Version to Resolve Unconstrained Optimization Problems

1,2*Ahmad Alhawarat, 1,3Nguyen-Thoi Trung and 4Zabidin Salleh

1Division of Computational Mathematics and Engineering, Institute for Computational Science, Ton Duc Thang University, Ho Chi Minh City, Vietnam
2Faculty of Mathematics and Statistics, Ton Duc Thang University, Ho Chi Minh City, Vietnam
3Faculty of Civil Engineering, Ton Duc Thang University, Ho Chi Minh City, Vietnam
4Department of Mathematics, Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu, 21030 Kuala Nerus, Terengganu, Malaysia

Abstract: One of the important methods that are widely utilized to resolve unconstrained optimization problems is the Conjugate Gradient (CG) method. This paper aims to propose a new version of the CG method on the basis of Weak Wolfe-Powell (WWP) line search. The assumption is bounded below optimization problems with the Lipschitz continuous gradient. The new parameter obtains global convergence properties when the WWP line search is used. The descent condition is established without using any line search. The performance of the proposed CG method is tested by obtaining some unconstrained optimization problems from the CUTEst library. Testing results show that the proposed version of the CG method outperforms CG-DESCENT version 5.3 in terms of CPU time, function evaluations, gradient evaluations and number of iterations.

Keywords: Unconstrained Optimization, Conjugate Gradient, Line Search, Convergence Analysis

Introduction

The Conjugate Gradient (CG) method is utilized to resolve unconstrained optimization problems in the form of:

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

where $f: \mathbb{R}^n \to \mathbb{R}$ represents the smooth function and denotes that the gradient is available. Using the CG method does not require a second derivative or its approximation as Newtons method or its modifications. Thus, this method is computationally inexpensive. The CG method is used to obtain a solution for the optimization problem:

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

By generating a sequence of points $x_{k+1}$ (Equation (1)), start from initial point $x_0$, where $x_k$ denotes the current iteration and $\alpha_k > 0$ indicates a step length obtained from a line search (Equation (2)-(6)).

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 1, 2, ...$$

The search direction $d_k$ of the CG method is defined in Equation (2):

$$d_k = \begin{cases} g_k & \text{if } k = 1 \\ g_k + \beta_k d_{k-1} & \text{if } k \geq 2 \end{cases}$$

(2)

where, $g_k = g(x_k)$ and $\beta_k$ is the CG formula.

The exact line search, which is expressed in Equation (3), can be used to obtain the step length:

$$\phi(\alpha) = \min f (x_k + \alpha d_k), \quad \alpha > 0$$

(3)

However, this type of line search is computationally expensive because numerous iterations are required to obtain the step length. Moreover, if the initial point is far from the optimum and/or the dimension of the problem is large, then an even greater number of iterations is required. With high-speed processors, sufficient memory and an appropriate choice of $\beta_k$, Equation (3) may be computationally acceptable for some functions. The inexact line search uses an approximation of the function and a reduced search space to find the step length. Therefore, the inexact line search is not as computationally expensive as...
the exact line search. The Strong Wolfe–Powell (SWP) line search is the most popular type of inexact line search and is calculated in Equation (4)-(6):

\[ f(x_k + \alpha d_k) \leq f(x_k) + \delta \alpha g_k^T d_k \quad (4) \]

and:

\[ g(x_k + \alpha d_k)^T d_k \leq \sigma \|g_k^T d_k\| \quad (5) \]

where:

\[ 0 < \delta < \sigma < 1. \]

The WWP line search is given by Equation (4) and (6):

\[ g(x_k + \alpha d_k)^T d_k \geq \sigma g_k^T d_k \quad (6) \]

The SWP line search forces the step length to be near a stationary point or the local minimum of the function, as the step length in the WWP line search may stratify without this advantage. The popular formulas for \( \beta_k \) are illustrated in Equation (7)-(11) (Fletcher and Reeves, 1964; Polak and Ribiere, 1969; Fletcher, 1987; Liu and Storey, 1991; Dai and Yuan, 1999):

\[ \beta_k^{\text{PRP}} = \frac{\|g_k^T\|}{\|g_{k-1}\|} \quad (7) \]

\[ \beta_k^{\text{WD}} = \frac{g_k^T (g_k - g_{k-1})}{\|g_{k-1}\|^2} \quad (8) \]

\[ \beta_k^{\text{CD}} = \frac{|g_k|^2}{d_k^T g_{k-1}} \quad (9) \]

\[ \beta_k^{\text{LS}} = \frac{g_k^T (g_k - g_{k-1})}{d_{k-1}^T g_{k-1}} \quad (10) \]

\[ \beta_k^{\text{PY}} = \frac{|g_k|^2}{(g_k - g_{k-1})^T d_{k-1}} \quad (11) \]

Wei et al. (2006) proposed a new positive CG method, which is relatively similar to the original Polak-Ribiere-Polyak (PRP) formula that has a global convergence under exact and inexact line search (Equation 12):

\[ \beta_k^{\text{WPL}} = \frac{g_k^T (g_k - \|g_k\| g_{k-1})}{\|g_{k-1}\|^2} \quad (12) \]

Theoretically, when \( \beta_k^{\text{PRP}} \approx 0 \), the search direction restarts automatically. However, (Powell, 1984) presented an example showing that \( \beta_k^{\text{PRP}} \) has no global convergence properties, whereas \( \beta_k^{\text{WS}} \) has a full global convergence property and satisfies the descent condition. However, the \( \beta_k^{\text{PRP}} \) formula is not as efficient as \( \beta_k^{\text{PRP}} \). Powell examined the \( \beta_k^{\text{PRP}} \) formula and found that this method cycle does not reach a solution when \( x_{k+1} \approx x_k \), which implies that \( \|g_k\|/\|g_{k-1}\| \approx 1 \). Hager and Zhang (2005) proposed a new CG method with a descent property for any inexact line search with \( g_k^T d_k \leq -7/8\|g_k\|^2 \). This method is globally convergent whenever the line search fulfills the Wolfe conditions. This formula is presented in Equation (13):

\[ \beta_k^{\text{WS}} = \max \{ \beta_k^{\text{WS}}, \eta_k \} \quad (13) \]

where:

\[ \beta_k^{\text{WS}} = \frac{1}{d_k^T g_k} \left( y_k - 2d_k \right) g_k^T \]

\[ \eta_k = \frac{1}{\|g_k\|} \min \{ \eta, \|g_k\| \} \] and \( \eta > 0 \) is a constant.

In the numerical experiments, they set \( \eta = 0.01 \). \( \beta_k^{\text{WS}} \) is called the CG-DESCENT method. Numerous versions for the CG-DESCENT code have appeared recently. Additional details will be discussed in the section of numerical results. In addition, (Hager and Zhang, 2005) proposed an approximate WWP line search as in Equation (14).

\[ \delta < \min \left\{ \frac{1}{2}, \sigma \right\} \]

Equation (14) is matched to the second Wolfe condition (Equation (6)). The first inequality in Equation (14) is matched to the first Wolfe condition (Equation (4)) when the function is quadratic. The new version of this method, called CG-DESCENT 6.3, was proposed in (Hager and Zhang, 2013).

One of important conditions in CG method called sufficient descent condition which proposed b Al-Baali (1985), which given as follows if there exists a constant \( c > 0 \) such that:

\[ g_k^T d_k \leq -c\|g_k\|^2, \forall k \in N \]
However, the concern regarding memory requirements and CPU time to solve unconstrained optimization problems has encouraged the development of the CG method. Over the years, numerous new formulas of the CG method have been proposed. Some of these formulas are difficult to use in different application fields, such as neural network, engineering and medical science. This restriction motivates us to construct a new version of the CG method, which is simple and relatively easy to understand. For more information the reader can read the following papers Alhawarat and Salleh (2017), Alhawarat et al. (2015), Hestenes and Stiefel (1952), Gilbert and Nocedal (1992) and Salleh and Alhawarat (2016).

The rest of this paper is organized in five sections. The new version of the CG method (MCG) is illustrated in section 2. Section 3 demonstrates the global convergence analysis for the new formula. Efficiency analysis based on numerical results are discussed and evaluated in section 4. We concluded in section 5.

**The Modified Conjugate Gradient (MCG) Method**

Alhawarat et al. (2016) presented a new CG formula with new restart criteria (Equation (15)):

$$
\beta_k^{\text{Alhawarat}} = \left\{ \begin{array}{ll}
|g_k|^2 - \lambda_k |g_k^T g_{k-1}|, & \text{if } |g_k| > \lambda_k |g_k^T g_{k-1}|, \\
0, & \text{otherwise},
\end{array} \right.
$$

where:

$$
\lambda_k = \frac{x_k - x_{k-1}}{|y_{k-1}|}.
$$

In the present study, we modified the formula as follows (Equation (16)):

$$
\beta_k^{\text{Alhawarat}} = \left\{ \begin{array}{ll}
|g_k|^2 - \lambda_k |g_k^T g_{k-1}|, & \text{if } |g_k| > \lambda_k |g_k^T g_{k-1}|, \\
0, & \text{otherwise},
\end{array} \right.
$$

where $\| \cdot \|$ represents the Euclidean norm, $\mu_k$ is defined by

$$
\lambda_k = \frac{x_k - x_{k-1}}{|y_{k-1}|} \quad \text{and} \quad m > 1.
$$

We note that Equation (16) satisfies the descent property without using any line search. In addition, we note that:

$$
0 \leq \beta_k^{\text{Alhawarat}} \leq \frac{|g_k|^2}{|g_{k-1}|^2} = \beta_k^{\text{CG}}
$$

The main steps of the MCG method are illustrated in algorithm (1).

**Algorithm (1): MCG**

Let $\epsilon \leq 10^{-6}$, $k = 1$; $d_1 = -g_1$

*Step 1:* Input $x_0$.

*Step 2:* If $|g_k| \leq \epsilon$ is satisfied, then stop.

*Step 3:* Compute the search direction $d_k$ according to (2) (with (16)).

*Step 4:* Compute the steplength $\alpha_k$ using (4) and (20).

*Step 5:* Update $x_{k+1}$ according to (1).

*Step 6:* Increment $k$ and go to Step 2.

**MCG: Global Convergence Analysis**

The following assumption is required to establish the convergence properties of the new formula $\beta_k^{\text{Alhawarat}}$.

**Assumption 1**

A. The level set $\Omega = \{ x | f(x) \leq f(x_0) \}$ is bounded and a positive constant $W$ exists such that $\| x \| \leq W, \forall x \in \Omega$.

B. In some neighborhood $T$ of $\Omega$, $f$ is continuously differentiable and the gradient is Lipschitz continuous. Then, for all $x, y \in T$, there exists a constant $L > 0$, which presents $\| g(x) - g(y) \| \leq L \| x - y \|$.

This case implies that a positive constant $R$ exists such that $\| g(u) \| \leq R, \forall u \in T$.

The descent condition is important in the study of the CG method; it is given by:

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

where $\epsilon \in (0,1)$.

**Global Convergence for $\beta_k^{\text{Alhawarat}}$ with the Modified WWP Line Search**

**Theorem 1**

Let the sequences $\{g_k\}$ and $\{d_k\}$ be generated using Equation (1), (2) and (16), where $\alpha_k$ is computed by any line search; then, the sufficient descent condition holds.

**Proof**

We use the proof by induction. By multiplying Equation (2) by $g_k^T$, we obtain:

$$
g_k^T d_k = g_k^T (-g_k + \beta_k d_{k-1}) = -|g_k|^2 + \beta_k g_k^T d_{k-1}
$$

(19)

$$
g_k^T d_k \leq -|g_k|^2 + \frac{|g_k|^2 - \mu_k |g_k^T g_{k-1}|}{|g_{k-1}|^2} |g_k^T d_{k-1}|
$$

(20)

$$
g_k^T d_k \leq -|g_k|^2 + \frac{|g_k|^2 - \mu_k |g_k^T g_{k-1}|}{m |g_k^T d_{k-1}|} |g_k^T d_{k-1}|
$$

(21)
\[ g_i^T d_i \leq -\|g_i\|^2 + \frac{\|g_i\|^2}{m} \quad (22) \]

When \( m > l \), we have \( g_i^T d_i < 0 \).

The proof is complete.

The following lemma is called (Zoutendijk, 1970), condition, which is useful for analyzing the global convergence property of the CG method.

**Lemma 1**

Suppose Assumption 1 holds. Let any method in the form of Equation (1) and (2) and \( \alpha_k \) satisfy the WWP line search (Equation (5) and (6)), in which the search direction is descent. Then, the following condition holds:

\[ \sum_{i=1}^{k} \| g_i \|^2 < \infty \quad (23) \]

In addition, Equation (23) holds for the exact and SWP line searches; the proof is presented in (Wei et al., 2006). Substituting Equation (18) into Equation (23) yields:

\[ \sum_{i=1}^{k} \| g_i \|^2 < \infty \quad (24) \]

The following theorem shows that \( \beta_k^i \) has a global convergence property with the SWP line search. To establish the convergence analysis for the modified CG method (Equation (16)) with the modified WWP condition, we need the following theorem.

**Theorem 2**

Let Assumption 1 hold. Consider any form of Equation (1) and (2) with Equation (17), in which \( \alpha_k \) is obtained from the WWP line search (Equation (4) and (14)). Then, Equation (16) satisfies the descent condition.

**Proof**

We use the proof by induction. By multiplying Equation (2) by \( g_i^T \), we obtain Equation (25):

\[ g_i^T d_i = g_i^T ( -g_i + \beta_k d_{k-1} ) = -\|g_i\|^2 + \beta_k g_i^T d_{k-1} \quad (25) \]

By dividing Equation (25) by \( \|g_i\|^2 \) (using Equation (5) and (6)), we obtain:

\[ (2\delta - 1)\varphi'(0) \geq \varphi'(\alpha_k) \geq \sigma \varphi'(0) \]

\[ -1 + \sigma \frac{g_{k-1}^T d_{k-1}}{\|g_{k-1}\|} \leq \frac{g_i^T d_i}{\|g_i\|^2} \leq -1 + (1 - 2\delta) \frac{g_{k-1}^T d_{k-1}}{\|g_{k-1}\|} \quad (26) \]

From Equation (2), we obtain \( g_i^T d_i = -\|g_i\|^2 \).

Suppose that the condition is true until \( k-1 \), i.e., \( g_i^T d_i < 0 \), for \( i = 1, 2, \ldots, k-1 \). By repeating the process for Equation (26), we obtain:

\[ -\sum_{j=0}^{k-1} \sigma^j \leq \frac{g_i^T d_i}{\|g_i\|^2} \leq -2 + \sum_{j=0}^{k-1} (1 - 2\delta)^j \]

As \( 0 < \delta < \sigma < 1 - \varepsilon \), where \( \varepsilon \to 0 \):

\[ \sum_{j=0}^{k-1} (1 - 2\delta)^j < \frac{1 - (1 - \sigma)^k}{1 - \sigma} \]

Then, for sufficient \( k \):

\[ \frac{1 - (1 - \sigma)^k}{1 - \sigma} < \varepsilon . \]

When \( \delta \min \left\{ \frac{1}{2 \sigma} \right\} \):

\[ \sum_{j=0}^{k-1} (1 - 2\delta)^j < \frac{1 - (1 - 2\delta)^k}{1 - (1 - \delta)} \]

Then, for large enough \( k \):

\[ \frac{1 - (1 - 2\delta)^k}{1 - (1 - \delta)} = \frac{1}{2 \delta} = \epsilon^* \]

\[ \frac{1 - (1 - \sigma)^k}{1 - \sigma} \leq \frac{1 - (1 - 2\delta)^k}{1 - (1 - \delta)} \leq \epsilon^* - 2. \]

When \( 0 < \delta < \sigma < 1 \), we obtain \( \frac{1 - (1 - \sigma)^k}{1 - \sigma} < \varepsilon \). Then:

\[ -\varepsilon < \frac{g_i^T d_i}{\|g_i\|^2} < \epsilon^* - 2 \]

(27)

The proof is complete.

**Theorem 3**

Let Assumption 1 hold. Consider any form of Equation (1) and (2) with (9), in which \( \alpha_k \) is obtained from the modified WWP line search (Equation (4) and (20)) with \( 0 < \sigma < 1 - \varepsilon \). Where \( \varepsilon \to 0 \), then, \( \lim_{k \to \infty} \|g_k\| = 0 \).

**Proof**

The theorem is proven by contradiction. Let the conclusion be false. Then, a constant \( \varepsilon > 0 \) exists (Equation (28)):

\[ \|g_k\| > \varepsilon, \forall k \geq 1 \quad (28) \]

Ahmad Alhawarat et al. / Journal of Computer Science 2020, 16 (9): 1220-1228
DOI: 10.3844/jcssp.2020.1220.1228
By squaring both sides of Equation (2), Equation (29) is obtained:

$$\|d_i\|^2 = \|g_i\|^2 - 2\beta_i g_i^T d_{k+1} + \beta_i^2 \|d_{k+1}\|^2$$

(29)

By dividing Equation (29) by $\|g_i\|^2$, we obtain Equation (30):

$$\frac{\|d_i\|^2}{\|g_i\|^2} = \frac{1}{\|g_i\|^2} - 2\beta_i \frac{\|d_{k+1}\|}{\|g_i\|^2} + \frac{\beta_i^2 \|d_{k+1}\|^2}{\|g_i\|^2}$$

(30)

By using Equation (5), (30), (16) and (27), we obtain

$$\frac{\|d_i\|^2}{\|g_i\|^2} \leq \frac{\|d_{k+1}\|^2}{\|g_i\|^2} + \frac{1}{\|g_i\|^2} + \frac{2\sigma \|g_{k+1}\|^2}{\|g_i\|^2}$$

$$\leq \frac{\|d_{k+1}\|^2}{\|g_i\|^2} + \frac{1+2\sigma}{\|g_i\|^2}$$

(31)

Repeating the process for Equation (31) and using the relationship $\frac{1}{\|g_i\|^2} = \frac{1}{\|d_i\|^2}$ yield Equation (32):

$$g_i \frac{\|d_i\|^2}{\|g_i\|^2} \leq (1 + 2\sigma) \sum_{i=1}^{k} \frac{1}{\|g_i\|^2}$$

(32)

From Equation (28), we obtain Equation (33):

$$\frac{\|g_i\|^2}{\|d_i\|^2} \geq \frac{\varepsilon^2}{k(1+2\sigma \varepsilon)}$$

(33)

Therefore, $\sum_{i=0}^{k} \|g_i\|^2 = \infty$.

This result contradicts Equation (28). Thus, $\lim_{k \to \infty} \|g_i\| = 0$. The proof is complete.

**Efficiency Analysis: Numerical Result**

To analyze the efficiency of the new method, some test functions are selected from CUTE (Bongartz et al., 1995), as shown in Table A1 (Appendix A). These functions are obtained from the CCPForge website (Gould et al., 2018). The selected functions and dimensions are similar to that used in (Hager and Zhang, 2005). Furthermore, the modified CG method is compared with CG-DESCENT 5.3 (Hager and Zhang, 2005). The comparison is based on CPU time, function evaluations, number of iterations and gradient evaluations. In this study, WWP is modified (presented by the modified CG-DESCENT 5.3), where the memory equal to zero is used to obtain the result for $\beta_i$. The code can be downloaded from the webpage of (Hager and Zhang, 2018). The CG-DESCENT 5.3 results are obtained by running CG-DESCENT 6.3 with memory equal to zero. The minimum time of 0.2 second is used for all algorithms with memory equal to zero. The host computer has an Intel (R) Dual-Core CPU and 2GB of DDR2 RAM. Figures 1-4, in which a performance measure introduced by (Powell, 1977) is used, show the results. This performance measure is presented to compare a set of solvers $S$ with a set of problems $P$. Assume that $n_s$ solvers and $n_p$ problems are $s$ and $p$, respectively. The measure $r_{p,s}$ is defined as the computation time (e.g., number of iterations or the CPU time) required for solver $s$ to solve problem $p$. To produce a baseline for comparison, we scale the performance of solver $s$ on problem $p$ by the top performance of any solver $S$ on the problem using the following fraction:

$$r_{p,s} = \frac{t_{p,s}}{\min \{t_{p,s}, s \in S \}}.$$  

Assume that the parameter $r_m \geq r_{p,s}$ for all $p, s$ is selected and further assumed if and only if solver $s$ does not solve problem $p$. As we would like to obtain an overall assessment of the performance of a solver, we defined measure $P_s(t)$:

$$p_s(t) = \frac{1}{n_p} \text{size}\{p \in \rho : r_{p,s} \leq t \}.$$  

Thus, $P_s(t)$ is the probability for solver $s \in S$ that the performance ratio $r_{p,s}$ is within a factor $t \epsilon R$ of the best possible ratio. If the function $P_s$ is identified as the cumulative distribution function for the performance ratio, then the performance measure $P_s : \mathbb{R} \rightarrow [0, 1]$ for a solver is non-decreasing and piecewise continuous from the right. The value of $P_s(t)$ is the probability that the solver obtains the best performance among all solvers. In general, a solver with high values of $P_s(t)$, which would appear in the upper right corner of the figure, is preferable for all figures.

Figure 1 shows that the modified CG method (Alhawarat) out performs CG-DESCENT 5.3 in terms of gradient evaluations. Figure 2 illustrates that $r_k$ strongly outperforms CG-DESCENT 5.3 with regard to function evaluation. Figures 3 and 4 show that the $r_k$ formula strongly outperforms CG-DESCENT 5.3 in terms of CPU time and number of iterations.
Fig. 1: Performance measure based on the gradient evaluations

Fig. 2: Performance measure based on function evaluation
Ahmad Alhawarat et al. / Journal of Computer Science 2020, 16 (9): 1220-1228
DOI: 10.3844/jcssp.2020.1220.1228

Fig. 3: Performance measure based on CPU time

Fig. 4: Performance measure based on number of iterations
Conclusion

In this study, a modified version of the CG algorithm (Alhawarat) is suggested and its performance is investigated. The modified formula is restarted on the basis of the value of the Lipchitz constant. The modified WWP line search is used to obtain the step length. The global convergence is established by using WWP. In addition, the descent condition is satisfied without using any line search. Our numerical results show that the new coefficient produces efficient and competitive results compared with other methods, such as CG-DESCENT 5.3. As future work, the new version of CG (MCG) method will be combined with feed-forward neural network (Back-Propagation (BP) algorithm) to improve the training process and produce fast training multilayer algorithm. This will help in reducing time needed to train neural network when the training samples are massive.

Acknowledgement

The first author would like to thank the University Ton Duc Thang University for its support to continue his research under postdoctoral program. In addition, the first Author would like to thank University Malaysia Terengganu for supporting. All authors would like to thank the editor and the reviewers for their valuable comments.

Funding

This research has been partially supported by Universiti Malaysia Terengganu under the Postgraduate Research Grant (PGRG) vote no. 55193/3.

Author’s Contributions

All authors equally contributed in this work.

Ethics

The corresponding author confirms that the other authors has read and approved the manuscript and there is no ethical issue involved.

References

Al-Baali, M. (1985). Descent property and global convergence of the Fletcher-Reeves method with inexact line search. IMA Journal of Numerical Analysis, 5(1), 121-124.

Alhawarat, A., & Salleh, Z. (2017, March). Modification of nonlinear conjugate gradient method with weak Wolfe-Powell line search. In Abstract and Applied Analysis (Vol. 2017). Hindawi.

Alhawarat, A., Mamat, M., Rivaie, M., & Salleh, Z. (2015). An efficient hybrid conjugate gradient method with the strong Wolfe-Powell line search. Mathematical Problems in Engineering, 2015.

Alhawarat, A., Salleh, Z., Mamat, M., & Rivaie, M. (2016). An efficient modified Polak–Ribière–Polyak conjugate gradient method with global convergence properties. Optimization Methods and Software, 32(6), 1299-1312.

Bongartz, I., Conn, A. R., Gould, N., & Toint, P. L. (1995). CUTE: Constrained and unconstrained testing environment. ACM Transactions on Mathematical Software (TOMS), 21(1), 123-160.

Dai, Y. H., & Yuan, Y. (1999). A nonlinear conjugate gradient method with a strong global convergence property. SIAM Journal on optimization, 10, 177-182.

Fletcher, R. (1987). Practical Methods of Optimization: Unconstrained Optimization. John Wiley & Sons.

Fletcher, R., & Reeves, C. M. (1964). Function minimization by conjugate gradients. The computer journal, 7(2), 149-154.

Gilbert, J. C., & Nocedal, J. (1992). Global convergence properties of conjugate gradient methods for optimization. SIAM Journal on optimization, 2, 21-42.

Gould, N. I., Orban, D., & Toint, P. L. (2018). CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. Computational Optimization and Applications, 60(3), 545-557.

Hager, W. W., & Zhang, H. (2005). A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM Journal on optimization, 16(1), 170-192.

Hager, W. W., & Zhang, H. (2013). The limited memory conjugate gradient method. SIAM Journal on Optimization, 23(4), 2150-2168.

Hager, W. W., & Zhang, H. (2018, July). Recent advances in bound constrained optimization. In IFIP Conference on System Modeling and Optimization (pp. 67-82). Springer, Boston, MA.

Hestenes, M. R., & Stiefel, E. (1952). Methods of conjugate gradients for solving linear systems. Journal of research of the National Bureau of Standards, 49(6), 409-436.

Liu, Y., & Storey, C. (1991). Efficient generalized conjugate gradient algorithms, part I: theory. Journal of optimization theory and applications, 69(1), 129-137.

Polak, E., & Ribiere, G. (1969). Note sur la convergence de méthodes de directions conjugées. ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique, 3(R1), 35-43.

Powell, M. J. (1984). Nonconvex minimization calculations and the conjugate gradient method. In Numerical analysis (pp. 122-141). Springer, Berlin, Heidelberg.
Powell, M. J. D. (1977). Restart procedures for the conjugate gradient method. Mathematical programming, 12(1), 241-254.
Salleh, Z., & Alhawarat, A. (2016). An efficient modification of the Hestenes-Stiefel nonlinear conjugate gradient method with restart property. Journal of Inequalities and Applications, 2016(1), 110.

Wei, Z., Yao, S., & Liu, L. (2006). The convergence properties of some new conjugate gradient methods. Applied Mathematics and computation, 183(2), 1341-1350.
Zoutendijk, G. (1970). Nonlinear programming, computational methods. Integer and nonlinear programming, 37-86.

Appendix (A)

Table 1: The test functions

| Test functions | Dim | Test functions | Dim | Test functions | Dim | Test functions | Dim |
|----------------|-----|----------------|-----|----------------|-----|----------------|-----|
| AKIVA          | 2   | DIXMAANJ       | 3000| LIARWHD        | 5000| STRATEC        | 10  |
| ALLINITU       | 4   | DIXMAANK       | 3000| LOGHAIRY       | 2   | TESTQUAD       | 5000|
| ARGLINA        | 200 | DIXMAANL       | 3000| MANCINO        | 100 | TOINTGOR       | 50  |
| ARGLINB        | 200 | DIXON3DQ       | 10000| MARATOSB      | 2   | TOINTGSS       | 5000|
| ARWHEAD        | 5000| DJTL           | 2   | MEXHAT         | 2   | TOINTPSP       | 50  |
| BARD           | 3   | DQDRTIC        | 5000| MOREBV         | 5000| TOINTG         | 50  |
| BDKQTIC        | 5000| DQRTIC         | 5000| MSQRTALS       | 1024| TQUARTIC       | 5000|
| BEALE          | 2   | EDENSCH        | 2000| MSQRTBLS      | 1024| TRIDIA         | 5000|
| BIGGS6         | 6   | EG2            | 1000| NCB20B         | 5000| VARDIM         | 200 |
| BOX3           | 3   | EIGENALS       | 2550| NONDIA         | 5000| VAREIGVL       | 50  |
| BOX            | 10000| EIGENCLS       | 2652| NONDQUAR       | 5000| VIBRBEAM       | 8   |
| BRKMCC         | 200 | ENGVAL1        | 5000| OSBORNEA       | 5   | WATSON         | 12  |
| BROWNAL        | 200 | ERRINROS       | 50  | OSCIPATH       | 10  | YFITU          | 3   |
| BROWNDE        | 4   | EXPFIT         | 2   | PALMER1C       | 8   | ZANGWIL2       | 2   |
| BRODYND7D      | 5000| FLETCBV2       | 5000| PALMER1D       | 7   | STRATEC        | 10  |
| BRYBNBD        | 5000| FLETCHCR       | 1000| PALMER2C       | 8   | TESTQUAD       | 5000|
| CHAINWOO       | 4000| FMINSRF2       | 5625| PALMER3C       | 8   | TOINTGOR       | 50  |
| CHNROSNB       | 50  | FMINSURF       | 5625| PALMER4C       | 8   | TOINTGSS       | 5000|
| CLIFF          | 2   | FREUROTH       | 5000| PALMER5C       | 6   | TOINTPSP       | 50  |
| COSINE         | 10000| GENHUMPS       | 5000| PALMER6C       | 8   | TOINTG         | 50  |
| CRAGGLVY       | 5000| GENROSE        | 5000| PALMER7C       | 8   | TQUARTIC       | 5000|
| CUBE           | 2   | GROWTHLS       | 3   | PALMER8C       | 8   | TRIDIA         | 5000|
| CURLY10        | 10000| GULF           | 3   | PARKCH         | 15  | VARDIM         | 200 |
| CURLY20        | 10000| HAIRY          | 2   | PENALTY1       | 1000| VAREIGVL       | 50  |
| CURLY30        | 10000| HATFLDD        | 3   | PENALTY2       | 200 | VIBRBEAM       | 8   |
| DECONVU        | 63  | HATFLDE        | 3   | PENALTY3       | 200 | WATSON         | 12  |
| DENSCHNA       | 2   | HATFLDL       | 3   | POWELLSG       | 5000| WOODS          | 4000|
| DENSCHNB       | 2   | HEARTSLS       | 6   | POWER          | 10000| YFITU         | 3   |
| DENSCHNC       | 2   | HEARTSLS       | 8   | QUARTC         | 5000| ZANGWIL2       | 2   |
| DENSCHND       | 3   | HELIX          | 3   | ROSENB         | 2   | STRATEC        | 10  |
| DENSCHNE       | 3   | HIELOW         | 3   | S308           | 2   | TESTQUAD       | 5000|
| DENSCHNF       | 2   | HILBERTA       | 2   | SCHMVETT       | 5000| TOINTG         | 50  |
| DIXMAANA       | 3000| HILBERTB       | 10  | SENSORS        | 100 | TOINTGSS       | 5000|
| DIXMAANB       | 3000| HIMELBB        | 2   | SINEVAL        | 2   |                |     |
| DIXMAANC       | 3000| HIMELBF        | 4   | SINKUAD        | 5000|                |     |
| DIXMAAND       | 3000| HIMELBG        | 2   | SISHER         | 2   |                |     |
| DIXMAANE       | 3000| HIMELBH        | 2   | SNAIL          | 2   |                |     |
| DIXMAANF       | 3000| HUMPS          | 2   | SPARSINE       | 5000|                |     |
| DIXMAANG       | 3000| JENSMP         | 2   | SPARSQR        | 10000|                |     |
| DIXMAANH       | 3000| JIMACK         | 3549| SPMRTL         | 4999|                |     |
| DIXMAANI       | 3000| KOWOSB         | 4   | SROSENB        | 5000|                |     |