A hybrid FR-DY conjugate gradient algorithm for unconstrained optimization with application in portfolio selection

Auwal Bala Abubakar1,2,3, Poom Kumam1,4,*, Maulana Malik5, Parin Chaipunya1,6 and Abdulkarim Hassan Ibrahim1

1 Center of Excellence in Theoretical and Computational Science (TaCS-CoE), Faculty of Science, King Mongkut’s University of Technology Thonburi (KMUTT), 126 Pracha Uthit Rd., Bang Mod, Thung Khru, Bangkok 10140, Thailand
2 Department of Mathematical Sciences, Faculty of Physical Sciences, Bayero University, Kano, Kano, Nigeria
3 Department of Mathematics and Applied Mathematics, Sefako Makgatho Health Sciences University, Ga-Rankuwa, Pretoria, Medunsa-0204, South Africa
4 Departments of Medical Research, China Medical University Hospital, China Medical University, Taichung 40402, Taiwan
5 Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Indonesia (UI), Depok 16424, Indonesia
6 NCAO Research Center, Fixed Point Theory and Applications Research Group, King Mongkut’s University of Technology Thonburi (KMUTT), 126 Pracha Uthit Rd., Bang Mod, Thung Khru, Bangkok 10140, Thailand

* Correspondence: Email: poom.kum@kmutt.ac.th; Tel: +6624708994; Fax: +6624284025.

Abstract: In this paper, we present a new hybrid conjugate gradient (CG) approach for solving unconstrained optimization problem. The search direction is a hybrid form of the Fletcher-Reeves (FR) and the Dai-Yuan (DY) CG parameters and is close to the direction of the memoryless Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton approach. Independent of the line search, the search direction of the new approach satisfies the descent condition and possess the trust region. We establish the global convergence of the approach for general functions under the Wolfe-type and Armijo-type line search. Using the CUTEr library, numerical results show that the propose approach is more efficient than some existing approaches. Furthermore, we give a practical application of the new approach in optimizing risk in portfolio selection.

Keywords: unconstrained optimization; hybrid three-term conjugate gradient method; global convergence; portfolio selection
1. Introduction

In this paper, we present a hybrid nonlinear conjugate gradient (CG) method for solving unconstrained optimization problems:

$$\min_{x \in \mathbb{R}^n} f(x),$$  \hspace{1cm} (1.1)

where $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and its gradient $g(x) = \nabla f(x)$ is Lipschitz continuous. CG methods are among the most preferred iterative methods for solving large-scale problems because of their simplicity in implementation, Hessian free and less storage requirements [14]. In view of these advantages, an encouraging number of CG methods were proposed (see, for example, [1, 5, 10, 12, 13, 31]).

The conjugate gradient method for solving (1.1) generates a sequence $\{x_k\}$ via the iterative formula

$$x_{k+1} = x_k + s_k, \quad s_k = \alpha_k d_k, \quad k = 0, 1, \ldots, \hspace{1cm} (1.2)$$

where $d_k$ is the search direction defined by

$$d_k := \begin{cases} -g_k, & \text{if } k = 0, \\ -g_k + \beta_k d_{k-1}, & \text{if } k > 0, \end{cases} \hspace{1cm} (1.3)$$

and for descent methods, $d_k$ is usually required to satisfy the sufficient descent property, if there exists a constant $c > 0$ such that for all $k$

$$g_k^T d_k \leq -c\|g_k\|^2. \hspace{1cm} (1.4)$$

The scalar $\alpha_k > 0$ is the step-size determined by a suitable line search rule, and $\beta_k$ is the conjugate gradient parameter that characterizes different type of conjugate gradient methods based on the global convergence properties and numerical performance. Some of the well-known nonlinear conjugate gradient parameters are the Fetcher-Reeves (FR) [9], Polak-Ribiére-Polyak (PRP) [25, 26], Hestenes-Stiefel (HS) [15], conjugate descent (CD) [8], Liu-Storey (LS) [22] and Dai-Yuan (DY) [6]. These parameters are given by the following formulae:

$$\beta_{k}^{FR} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \quad \beta_{k}^{CD} = \frac{\|g_k\|^2}{-d_{k-1}^T g_{k-1}}, \quad \beta_{k}^{DY} = \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}}, \hspace{1cm} (1.5)$$

$$\beta_{k}^{HS} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}, \quad \beta_{k}^{PRP} = \frac{g_k^T y_{k-1}}{\|g_{k-1}\|^2}, \quad \beta_{k}^{LS} = \frac{g_k^T y_{k-1}}{-d_{k-1}^T d_{k-1}}, \hspace{1cm} (1.6)$$

where $g_k = g(x_k)$, $y_{k-1} = g_k - g_{k-1}$ and $\| \cdot \|$ is the Euclidean norm.

In order to have some of the classical CG methods possess a descent direction and trust region as well as improving their numerical efficiency, the three-term CG and hybrid CG methods were introduced for solving (1.1). For instance, Mo et al. [23] proposed two hybrid methods for solving
unconstrained optimization problems. The methods are based on the Touati-Ahmed and Storey [30] and the DY methods. Under the strong Wolfe condition, the global convergence was proved. Andrei in [2] proposed a simple three-term CG method obtained by modifying the Broyden-Fletcher-Goldfarb-Shanno (BFGS) updating formula of the inverse approximation of the Hessian. The search direction satisfies both the descent and conjugacy conditions. Numerical results were given to support the theoretical results. Also in [3], Andrei proposed an elliptic CG method for solving (1.1). The search direction is the sum of the negative gradient and a vector obtained by minimizing the quadratic approximation of the objective function. In addition, the search direction satisfies both Dai-Liao (DL) conjugacy and descent conditions. Eigenvalue analysis was carried out to determine parameter which the search direction depends on. In comparison with the well-known CG_DESCENT method, the proposed method was more efficient. Liu and Li [21] proposed a new hybrid CG with it’s search direction satisfying the DL conjugacy condition and the Newton direction independent of the line search. As usual the global convergence was established under the strong Wolfe line search. In [16], Jian et al. proposed a new hybrid CG method based on previous classical methods. At every iteration, the method produces a descent direction independent of the line search. Global convergence was proved and numerical experiments on medium-scale problems was conducted and the results reported. By applying a mild modification on the HS method, Dong et al. proposed a new approach that generates a descent direction. Also, the approach satisfies an adaptive conjugacy condition and has a self-restarting property. The global convergence was proved for general functions and the efficiency of the approach was shown via numerical experiments on some large-scale problems. Min Li [19] developed a three-term PRP CG method with the search direction close to the direction of the memoryless BFGS quasi-Newton method. The method collapses to the standard PRP method when an exact line search is considered. Independent of any line search, the method satisfies the descent condition. The global convergence of the method was established using an appropriate line search. Numerical results show that the method is efficient for the standard unconstrained problems in the CUTEr library. Again, Li [18] proposed a nonlinear CG method which generates a search direction that is close to that of the memoryless BFGS quasi-Newton method. Moreover, the search direction satisfies the descent condition. Global convergence for strongly convex functions and nonconvex functions was established under the strong Wolfe line search. Furthermore, an efficient spectral CG method that combine the spectral parameter and a three-term PRP method was proposed by Li et al. [20]. The method is based on the quasi-Newton direction and quasi-Newton equation. Numerical experiments reported show the superiority of the method over the three-term PRP method.

Motivated by the works of Li [18,19] which originated from [17,27], we propose a new hybrid CG method for solving (1.1). The hybrid direction is a combination of the FR and DY directions that are both close to the direction of the memoryless BFGS quasi-Newton method. Interestingly, the hybrid direction satisfies the descent condition and is bounded independent of any line search procedure. We prove the global convergence under both the Wolfe-type and Armijo-type line searches. Numerical results are also provided to show the efficiency of the new hybrid method. Finally, application of the method in optimizing risk in portfolio selection is also considered. The remainder of this paper is organized as follows. In the next section, the hybrid method is derived together with it’s convergence. In Section 3, we provide some numerical experimental results.
2. Algorithm and theoretical results

We begin this section by recalling a three-term CG method for solving (1.1). From an initial guess $x_0$, the method computes the search direction as follows:

$$d_0 := -g_0, \; d_k := -g_k + \beta_k d_{k-1} + \gamma_k g_k, \; k \geq 1,$$

(2.1)

where $\beta_k$, $\gamma_k$ are parameters. Distinct choices of the parameters $\beta_k$ and $\gamma_k$ correspond to distinct three-term CG methods. It is clear that, the three-term CG methods collapses to the classical ones when $\gamma_k = 0$.

Next, we will recall the memoryless BFGS method proposed by Nocedal [17] and Shanno [27], where the search direction can be written as

$$d_k^{BFGS} := -Q_k g_k,$$

$$Q_k = -(I - \frac{s_k^{-1} y_k^T}{s_k^{-1} y_k} - \frac{y_k s_k^{-T}}{s_k^{-1} y_k} + \frac{s_k^{-1} y_k^T y_k s_k^{-T}}{s_k^{-1} y_k} + \frac{s_k^{-1} s_k^{-T}}{s_k^{-1} y_k} ),$$

$s_{k-1} = x_k - x_{k-1} = \alpha_k d_{k-1}$ and $I$ is the identity matrix. After simplification, $d_k^{BFGS}$ can be rewritten as

$$d_k^{BFGS} := -g_k + \left( \beta_k^{HS} - \frac{\|y_k-1\|^2 g_k^T d_{k-1}}{(d_{k-1}^T y_k)^2} \right) d_{k-1} + \frac{g_k^T d_{k-1}}{d_{k-1}^T y_k} (y_k - s_{k-1}).$$

(2.2)

Replacing $\beta_k^{HS}$ with $\beta_k^{FR}$ and $\frac{\|y_k-1\|^2 g_k^T d_{k-1}}{(d_{k-1}^T y_k-1)^2}$ with $\frac{\|g_k\|^2 g_k^T d_{k-1}}{\|g_k\|^2}$ in (2.2), we define a three-term search direction as

$$d_k^{TTFR} := -g_k + \left( \beta_k^{FR} - \frac{\|g_k\|^2 g_k^T d_{k-1}}{\|g_k\|^2} \right) d_{k-1} - t_k \frac{g_k^T d_{k-1}}{\|g_k\|^2} g_k.$$  

(2.3)

Again, replacing $\beta_k^{HS}$ with $\beta_k^{DY}$ and $\frac{\|y_k-1\|^2 g_k^T d_{k-1}}{(d_{k-1}^T y_k-1)^2}$ with $\frac{\|g_k\|^2 g_k^T d_{k-1}}{(d_{k-1}^T y_k)^2}$ in (2.2), we define another three-term search direction as

$$d_k^{TTFR} := -g_k + \left( \beta_k^{DY} - \frac{\|g_k\|^2 g_k^T d_{k-1}}{(d_{k-1}^T y_k)^2} \right) d_{k-1} - t_k \frac{g_k^T d_{k-1}}{d_{k-1}^T y_k} g_k.$$  

(2.4)

To find the parameter $t_k$, we require the solution of the univariate minimal problem

$$\min_{t \in \mathbb{R}} \| (y_k-1 - s_{k-1}) - t g_k \|^2.$$

Let $E_k = (y_k-1 - s_{k-1}) - t g_k$, then

$$E_k E_k^T = [(y_k-1 - s_{k-1}) - t g_k] [(y_k-1 - s_{k-1}) - t g_k]^T$$

$$= [(y_k-1 - s_{k-1}) - t g_k] [(y_k-1 - s_{k-1})^T - t g_k^T]$$

$$= t^2 g_k g_k^T - t [(y_k-1 - s_{k-1}) g_k^T + g_k (y_k-1 - s_{k-1})^T] + (y_k-1 - s_{k-1}) (y_k-1 - s_{k-1})^T.$$

Letting $A_k = y_k-1 - s_{k-1}$, then

$$E_k E_k^T = t^2 g_k g_k^T - t [A_k g_k^T + g_k A_k^T] + A_k A_k^T$$
and
\[ tr(E_k E_k^T) = t^2 \|g_k\|^2 - t \{ tr(A_k g_k^T) + tr(g_k A_k^T) \} + \|A_k\|^2 \]
\[ = t^2 \|g_k\|^2 - 2t g_k^T A_k + \|A_k\|^2. \]

Differentiating the above with respect to \( t \) and equating to zero, we have
\[ 2t \|g_k\|^2 - 2g_k^T A_k = 0, \]
which implies
\[ t = \frac{g_k^T (y_{k-1} - s_{k-1})}{\|g_k\|^2}. \tag{2.5} \]

Hence, we select \( t_k \) as
\[ t_k := \min \{ \bar{t}, \max \left\{ 0, \frac{g_k^T (y_{k-1} - s_{k-1})}{\|g_k\|^2} \right\} \}, \tag{2.6} \]
which implies \( 0 \leq t_k \leq \bar{t} < 1 \).

Motivated by the three-term CG directions defined in (2.3) and (2.4), we propose a hybrid three-term CG based algorithm for solving (1.1), where the search direction is defined as
\[ d_0 := -g_0, \quad d_k := -g_k + \beta_k^{HTT} d_{k-1} + \gamma_k g_k, \quad k \geq 1, \tag{2.7} \]
where
\[ \beta_k^{HTT} := \frac{\|g_k\|^2}{w_k} - \frac{\|g_k\|^2 g_k^T d_{k-1}}{w_k}, \quad \gamma_k := -t_k \frac{g_k^T d_{k-1}}{w_k}, \tag{2.8} \]
and
\[ w_k := \max \{ \lambda \|d_{k-1}\|, \|g_k\|, \|g_k^T d_{k-1}\|, \|g_k^T y_{k-1}\|, \|g_k\|^2 \}, \quad \lambda > 0. \tag{2.9} \]

**Remark 2.1.** Observe that, because of the way \( w_k \) is defined, the parameter \( \beta_k^{HTT} \) is a hybrid of \( \beta_k^{FR} - \frac{\|g_k\|^2 g_k^T d_{k-1}}{\|g_k\|^2} \) and \( \beta_k^{DY} - \frac{\|g_k\|^2 g_k^T d_{k-1}}{\|d_{k-1}\| \|y_{k-1}\|^2} \) in (2.3) and (2.4), respectively. Also, the parameter \( \gamma_k \) defined by (2.8) is a hybrid of \(-t_k \frac{g_k^T d_{k-1}}{\|d_{k-1}\|^2}\) and \(-t_k \frac{g_k^T d_{k-1}}{\|d_{k-1}\| \|y_{k-1}\|^2}\) in (2.3) and (2.4), respectively. Finally, the search direction given by (2.7) is close to that of the memoryless BFGS method when \( t_k = \frac{\|g_k^T (y_{k-1} - s_{k-1})\|}{\|g_k\|^2} \).

**Remark 2.2.** Note that, relation (2.9) is carefully defined such that the search direction possess a trust region (see Lemma 2.7) independent of the line search.

**Lemma 2.3.** The search direction \( d_k \) defined by (2.7) satisfy (1.4) with \( c = \frac{3}{4} \).

**Proof.** Multiplying both sides of (2.7) with \( g_k^T \), we have
\[ g_k^T d_k = -\|g_k\|^2 + \frac{\|g_k\|^2}{w_k} g_k^T d_{k-1} - \frac{\|g_k\|^2}{w_k^2} (g_k^T d_{k-1})^2 - t_k \frac{\|g_k\|^2}{w_k} g_k^T d_{k-1} \]
\[ \leq -\|g_k\|^2 + (1 - t_k) \frac{\|g_k\|^2}{w_k} g_k^T d_{k-1} - \frac{\|g_k\|^2}{w_k^2} (g_k^T d_{k-1})^2 \]
\[ = -\|g_k\|^2 + 2 \left( \frac{1 - t_k}{2} \right) \frac{g_k^T d_{k-1}}{w_k} - \frac{\|g_k\|^2}{w_k^2} (g_k^T d_{k-1})^2 \]
\[ \leq -\|g_k\|^2 + \left( \frac{1 - t_k}{4} \right) \|g_k\|^2 + \left( \frac{\|g_k\|^2}{w_k^2} \right) (g_k^T d_{k-1})^2 - \left( \frac{\|g_k\|^2}{w_k^2} \right) (g_k^T d_{k-1})^2 \]

\[ = -\|g_k\|^2 + \left( \frac{1 - t_k}{4} \right) \|g_k\|^2 \]

\[ = - \left( 1 - \left( \frac{1 - t_k}{4} \right) \right) \|g_k\|^2 \]

\[ \leq - \frac{3}{4} \|g_k\|^2. \]

\[ □ \]

Next, we will turn our attention to establishing the convergence of the proposed scheme by first considering the standard Wolfe line search conditions \[29,\]

\[ f(x_k + \alpha_k d_k) - f(x_k) \leq \vartheta \alpha_k g_k^T d_k, \quad (2.10) \]

\[ g(x_k + \alpha_k d_k)^T d_k \geq \sigma g_k^T d_k \quad (2.11) \]

where \( 0 < \vartheta < \sigma < 1 \). In addition, we will assume that

**Assumption 2.4.** The level set \( \mathcal{H} = \{x : f(x) \leq f(x_0)\} \) is bounded.

**Assumption 2.5.** Suppose \( \mathcal{H} \) is some neighborhood of \( \mathcal{L} \), then \( f \) is continuously differentiable and its gradient Lipschitz continuous on \( \mathcal{H} \). That is, we can find \( L > 0 \) such that for all \( x \)

\[ \|g(x) - g(\bar{x})\| \leq L\|x - \bar{x}\|, \quad \bar{x} \in \mathcal{H}. \quad (2.12) \]

From Assumption 2.4 and 2.5, we can deduce that for all \( x \in \mathcal{L} \) there exist \( b_1, b_2 > 0 \) for which

- \( \|x\| \leq b_1 \).
- \( \|g(x)\| \leq b_2 \).

Furthermore, the sequence \( \{x_k\} \in \mathcal{L} \) because \( \{f(x_k)\} \) is decreasing. Henceforth, we will suppose that Assumption 2.4–2.5 hold and that the objective function is bounded below. We will now prove the convergence result.

**Theorem 2.6.** Let conditions (2.10) and (2.11) be fulfilled. If

\[ \sum_{k=0}^{\infty} \frac{1}{\|d_k\|^2} = +\infty, \quad (2.13) \]

then

\[ \liminf_{k \to \infty} \|g_k\| = 0. \quad (2.14) \]

**Proof.** Suppose by contradiction that Eq (2.14) does not hold, then there exists a nonnegative scalar \( \epsilon \) such that

\[ \|g_k\| \geq \epsilon \quad \text{for all } k \in \mathbb{N}. \quad (2.15) \]

From Lemma 2.3 and (2.10),

\[ f(x_{k+1}) \leq f(x_k) + \vartheta \alpha_k g_k^T d_k \leq f(x_k) - \vartheta \alpha_k \|g_k\|^2 \leq f(x_k) \leq f(x_{k-1}) \leq \ldots \leq f(x_0). \]
Likewise, by Lemma 2.3, condition (2.11) and Assumption 2.5, we have

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

Combining the above inequality with (2.10), we obtain

\[
\frac{\vartheta(1 - \sigma) (g_k^T d_k)^2}{L \|d_k\|^2} \leq f(x_k) - f(x_{k+1})
\]

and

\[
\frac{\vartheta(1 - \sigma)}{L} \sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} \leq (f(x_0) - f(x_1)) + (f(x_1) - f(x_2)) + \ldots \leq f(x_0) < +\infty,
\]

since \(\{f(x_k)\}\) is bounded. The above implies that

\[
\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < +\infty. \quad (2.16)
\]

Now, inequality (2.15) with (1.4) implies that

\[
g_k^T d_k \leq -\frac{3}{4} \|g_k\|^2 \leq -\frac{3}{4} \epsilon^2. \quad (2.17)
\]

Squaring both sides and dividing by \(\|d_k\|^2 \neq 0\) of (2.17), yields

\[
\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} \geq \frac{9}{16} \sum_{k=0}^{\infty} \frac{\epsilon^4}{\|d_k\|^2} = +\infty \quad (2.18)
\]

This contradicts (2.16). Therefore, the conclusion of the theorem hold. \(\square\)

Next, we will establish the convergence of the proposed method under the Armijo-type backtracking line search procedure. The procedure was first introduced by Grippo and Lucidi [11], where the step size \(\alpha_k\) is determined as follows: \(\rho, \vartheta \in (0, 1), \alpha_k = \rho^i\), where \(i\) is the smallest nonnegative integer for which the relation

\[
f(x_k + \alpha_k d_k) \leq f(x_k) - \vartheta \alpha_k^2 \|d_k\|^2
\]

hold.

From (2.19) and the fact that \(\{f(x_k)\}\) is decreasing, we can deduce that

\[
\sum_{k=0}^{\infty} \alpha_k^2 \|d_k\|^2 < +\infty,
\]

which further implies that

\[
\lim_{k \to \infty} \alpha_k \|d_k\| = 0. \quad (2.20)
\]

**Lemma 2.7.** If \(\{d_k\}\) is defined by (2.7), then there is \(N_1 > 0\) for which \(\|d_k\| \leq N_1\).
Proof. From (2.8),
\[
\|\beta_k^{HTT}\| = \frac{\|g_k\|^2}{w_k} - \frac{\|g_k\|^2 g_k^T d_{k-1} - g_k^T g_k d_{k-1}}{w_k^2}
\leq \frac{\|g_k\|^2}{\lambda \|d_{k-1}\| \|g_k\|} - \frac{\|g_k\|^3 \|d_{k-1}\|}{(\lambda \|d_{k-1}\| \|g_k\|)^2}
= \left(\frac{1}{\lambda} + \frac{1}{\lambda^2}\right) \frac{\|g_k\|}{\|d_{k-1}\|}.
\]
(2.21)

Also,
\[
|\gamma_k| = \left| -t_k \frac{g_k^T d_{k-1}}{w_k} \right|
= t_k \left| \frac{g_k^T d_{k-1}}{w_k} \right|
\leq \tilde{t} \frac{\|g_k\| \|d_{k-1}\|}{w_k}
\leq \tilde{t} \frac{\|g_k\| \|d_{k-1}\|}{\lambda \|d_{k-1}\| \|g_k\|}
= \frac{\tilde{t}}{\lambda}.
\]
(2.22)

Therefore, from (2.7), (2.21) and (2.22), we have
\[
\|d_k\| = \|g_k + \beta_k^{HTT} d_{k-1} + \gamma_k g_k\|
\leq \|g_k\| + \|\beta_k^{HTT}\| \|d_{k-1}\| + |\gamma_k| \|g_k\|
\leq \|g_k\| + \left(\frac{1}{\lambda} + \frac{1}{\lambda^2}\right) \frac{\|g_k\|}{\|d_{k-1}\|} \|d_{k-1}\| + \tilde{t} \|g_k\|
= \left(1 + \frac{1 + \tilde{t}}{\lambda} + \frac{1}{\lambda^2}\right) \|g_k\|
\leq \left(1 + \frac{1 + \tilde{t}}{\lambda} + \frac{1}{\lambda^2}\right) b_2.
\]
(2.23)

(2.24)

Letting \(N_1 = \left(1 + \frac{1 + \tilde{t}}{\lambda} + \frac{1}{\lambda^2}\right) b_2\), we have
\[
\|d_k\| \leq N_1.
\]
(2.25)

\[\square\]

Theorem 2.8. If the step size \(\alpha_k\) is obtained via relation (2.19), then
\[
\liminf_{k \to \infty} \|g_k\| = 0.
\]
(2.26)

Proof. Suppose by contradiction equation (2.26) is not true. Then for all \(k\), we can find an \(r > 0\) for which
\[
\|g_k\| \geq r.
\]
(2.27)
Let $\alpha_k = \rho^{-1} \alpha_k$, then $\alpha_k$ does not satisfy (2.19). That is
\[
f(x_k + \rho^{-1} \alpha_k d_k) > f(x_k) - \vartheta \rho^{-2} \alpha_k^2 ||d_k||^2.
\] (2.28)
Applying the mean value theorem together with Lemma 2.7, (1.4) and (2.12), there exist an $l_k \in (0, 1)$ such that
\[
f(x_k + \rho^{-1} \alpha_k d_k) - f(x_k) = \rho^{-1} \alpha_k g(x_k + l_k \rho^{-1} \alpha_k d_k) - f(x_k)
= \rho^{-1} \alpha_k g^T_k d_k + \rho^{-1} \alpha_k (g(x_k + l_k \rho^{-1} \alpha_k d_k) - g_k)^T d_k
\leq \rho^{-1} \alpha_k g^T_k d_k + L \rho^{-2} \alpha_k^2 ||d_k||^2.
\]
Inserting the above relation in (2.28), together with (2.25) and (2.27) we have
\[
\alpha_k \geq \frac{\rho ||g_k||^2}{(L + \vartheta) ||d_k||^2} \geq \frac{\rho r^2}{(L + \vartheta) N_i^2} > 0.
\]
This and (2.20) gives
\[
\lim_{k \to \infty} ||d_k|| = 0. \quad (2.29)
\]
On the other hand, applying backward Cauchy-Schwartz inequality on (1.4) gives
\[
||d_k|| \geq \frac{3}{4} ||g_k||.
\]
Thus, we have $\lim_{k \to \infty} ||g_k|| = 0$. This is a contradiction and therefore (2.26) holds.

3. Numerical experiments

This section discusses the computational efficiency of the proposed method, namely HTT method. One way to measure the efficiency of a method is to use the test function. An important test function is used to validate and compare among optimization methods, especially newly developed methods. We selected 34 test functions and initial points as in Table 1 mostly considered by Andrei [4]. For each function we have taken five numerical experiments with a dimension of which is among $n = 10, 50, 80, 100, 200, 400, 300, 500, 600, 1000, 3000, 5000, 10,000, 15,000, 20,000$. However, we often use dimensions that are greater than 1000. The executed methods are coded in Matlab 2019a and compiled using personal laptop; Intel Core i7 processor, 16 GB RAM, 64 bit Windows 10 Pro operating system.

As a good comparison, for NHS+ and NPRP+ all parameters are maintained according to their articles in [18] and [19], i.e, $\vartheta = 0.1$, $\sigma = 0.9$. Specially, for HTT, we set the value of parameters $\vartheta = 0.0001$, $\sigma = 0.009$. For all methods, we use the parameter value $\bar{t} = 0.3$, $\lambda = 0.01$ and the step-size $\alpha_k$ is calculated by standard Wolfe line search. The numerical results are compared based on number of iterations (NOI), number of function evaluations (NOF), and CPU time in seconds (CPU). In this experiment, we consider a stopping condition that many researchers suggest (see [16, 18, 21, 23]), namely that the algorithm will stop when $||g_k|| \leq 10^{-6}$.
In Table 5 we list the numerical results obtained by compiling each method for completing each test function with different dimension sizes. If the number of iterations of the method exceeds 10,000 or it never reaches the optimal value, the algorithm stops and we write it as ‘FAIL’.

To compare the performance between methods, we use the performance profiles of Dolan and Moré [7] with rule as follows. Let $S$ is the set of methods and $P$ is set of the test problems with $n_p, n_s$ are the number of test problems and the number of methods, respectively.

**Table 1.** List of test functions and initial points.

| Number | Functions                          | Initial Points          |
|--------|-----------------------------------|-------------------------|
| F1     | Extended White & Holst            | (-1.2, 1, ..., -1.2, 1) |
| F2     | Extended Rosenbrock               | (-1.2, 1, ..., -1.2, 1) |
| F3     | Extended Freudenstein & Roth      | (0.5, -2, ..., 0.5, -2) |
| F4     | Extended Beale                    | (1, 0.8, ..., 1, 0.8)   |
| F5     | Raydan 1                          | (1, 1, ..., 1)          |
| F6     | Extended Tridiagonal 1            | (2, 2, ..., 2)          |
| F7     | Diagonal 4                        | (1, 1, ..., 1)          |
| F8     | Extended Himmelblau              | (1, 1, ..., 1)          |
| F9     | FLETCHCR                          | (0.5, 0.5, ..., 0.5)    |
| F10    | Extended Powel                    | (1, 1, ..., 1)          |
| F11    | NONSCOMP                          | (3, 3, ..., 3)          |
| F12    | DENSCHNB                         | (10, 10, ..., 10)       |
| F13    | Extended Penalty Function U52    | (1/100, 2/100, ..., n/100) |
| F14    | Hager                            | (1, 1, ..., 1)          |
| F15    | Shallow                          | (2, 2, ..., 2)          |
| F16    | Quadratic QF2                    | (0.5, 0.5, ..., 0.5)    |
| F17    | Generalized Tridiagonal 1        | (2, 2, ..., 2)          |
| F18    | Generalized Tridiagonal 2        | (1, 1, ..., 1)          |
| F19    | POWER                             | (1, 1, ..., 1)          |
| F20    | Quadratic QF1                    | (1, 1, ..., 1)          |
| F21    | QP2 Extended Quadratic Penalty   | (2, 2, ..., 2)          |
| F22    | QP1 Extended Quadratic Penalty   | (1, 1, ..., 1)          |
| F23    | Sphere                            | (1, 1, ..., 1)          |
| F24    | Sum Squares                       | (0.1, 0.1, ..., 0.1)    |
| F25    | DENSCHNA                          | (7, 7, ..., 7)          |
| F26    | DENSCHNC                          | (100, -1, -1, ..., -1, -1) |
| F27    | DENSCHNF                          | (100, -100, ..., 100, -100) |
| F28    | Extended Block-Diagonal BD1      | (1, 1, ..., 1)          |
| F29    | HIMMELBH                          | (0.8, 0.8, ..., 0.8)    |
| F30    | Extended Hiebert                 | (5.001, 5.001, ..., 5.001) |
| F31    | ENGVAL1                           | (2, 2, ..., 2)          |
| F32    | ENGVAL8                           | (0, 0, ..., 0)          |
| F33    | Linear Perturbed                  | (0, 0, ..., 0)          |
| F34    | QUARTICCM                         | (2, 2, ..., 2)          |
The performance profile $\omega : \mathbb{R} \to [0, 1]$ is for each $s \in S$ and $p \in P$ defined that $a_{p,s} > 0$ is NOI or NOF or CPU time required to solve problems $p$ by method $s$. Furthermore, the performance profile is obtained by:

$$\omega_s(\tau) = \frac{1}{n_p} \text{size}\{p \in P : \log_2 r_{p,s} \leq \tau\},$$

where $\tau > 0$, $\text{size}\ B$ is the number of the elements in the set $B$, and $r_{p,s}$ is performance ratio defined as:

$$r_{p,s} = \frac{a_{p,s}}{\min\{a_{p,s} : p \in P \text{ and } s \in S\}}.$$

In general, the method whose performance profile plot is on the top right will win the rest of the methods or represents the best method.

From Table 5, all the methods failed to solve the Raydan 1 and NONSCOMP with $n = 1,000, 5,000, 10,000, 15,000, 20,000$, the NHS+ and NPRP+ methods failed to solve Extended Powel, Hager, Generalized Tridiagonal 1, Generalized Tridiagonal 2, QP2 Extended Quadratic Penalty, DENSCHNA, Extended Block-Diagonal BD1, ENGVAL1, and ENGVAL8 for all of dimension given, and NPRP+ method has more failures for the given problem compared to other methods. So, based on the numerical results in Table 5, we can say that the HTT method has the best performance. Meanwhile, from the result in performance profile in Figures 1–3 show that the HTT method profile is always on the top right curve, whether it’s based on NOI, NOF or CPU time. The final conclusion is that the HTT method has the best performance compared the NHS+ and NPRP+ methods under the test functions given.
4. Application in portfolio selection

Investment is a commitment to invest several funds carried out at this time to obtain many benefits in the future. One investment that is quite attractive is a stock investment. An investor can invest in more than one stock. Of course, the thing to consider is whether the investment can be profitable or not. One theory that discusses investment in several assets is portfolio theory. A stock portfolio is a collection of stocks owned by an investor [24].

In this section, we present the problem of risk management in a portfolio of stocks. The main issue
is how to balance a portfolio, that is, how to choose the percentage of each stock in the portfolio to minimize the risk [28]. In investing, investors expect to get a large return by choosing the smallest possible risk. Return is the income received by an investor from stocks traded on the capital market. There are several ways to calculate returns, one of which is to use arithmetic returns which can be calculated as follows:

\[ R_t = \frac{P_t - P_{t-1}}{P_{t-1}}, \]

where \( P_t \) is the stock prices at time \( t \) and \( P_{t-1} \) is the stock prices at time \( t - 1 \). Furthermore, we may consider the mean of return, the expected value, and the variance of the return. The mean of return of a stock \( i \) is denoted by

\[ \bar{r}_i = \frac{1}{n} \sum_{t=1}^{n} r_{it}, \]

where \( n \) is number of returns on a stock and \( r_{it} \) is return at time \( t \) on stock \( i \). The expected return of stock \( i \)

\[ \mu_i = E(R_i). \]

The variance of return of stock \( i \)

\[ \sigma_i^2 = Var(R_i), \]

is called the risk of stock \( i \) [28]. One thing that needs to be considered is the relationship between stocks, so we must also pay attention to how stocks interact, i.e., by using the covariance of individual risk. Covariance measures the directional relationship between the returns on two stocks. If positive covariance then that stock returns move together while a negative covariance means they move inversely. Usually the covariance of return between two stocks \( R_i, R_j \) is denoted as \( \text{Cov}(R_i, R_j) \) [28].

For our cases, we consider the seven blue chip stocks in Indonesia, namely, PT Bank Rakyat Indonesia (Persero) Tbk (BBRI), PT Unilever Indonesia Tbk (UNVR), PT Telekomunikasi Indonesia Tbk (TLKM), PT Indofood CBP Sukses Makmur Tbk (ICBP), PT Bank Mandiri (Persero) Tbk (BMRI), PT Perusahaan Gas Negara Tbk (PGAS), and PT Astra International Tbk (ASII). The stock price used is the weekly closing price which data history is taken from the database http://finance.yahoo.com, over a period of 3 years (Jan 1, 2018 – Dec 31, 2020). Based on this data, we may see the movement of the closing price of each stock in Figure 4.

![Figure 4. Weekly closing price of all stocks in IDR (Jan 1, 2018 – Dec 31, 2020).](image)
So that the portfolio returns for the seven stocks are defined as the weighted amount of returns for each stocks

\[ R = \sum_{i=1}^{7} w_i R_i, \]

where \( w_i \) is the percentage of the value of the stock contained in the portfolio. Thus, we may define the expected return and risk on our portfolio (see [28]). The expected return \( \mu \) on our portfolio is the expected value of the portfolio’s return as follows:

\[ \mu = E\left( \sum_{i=1}^{7} w_i R_i \right) = \sum_{i=1}^{7} w_i \mu_i, \quad (4.1) \]

and the risk of our portfolio is the variance of the portfolio’s return as follows,

\[ \sigma^2 = Var\left( \sum_{i=1}^{7} w_i R_i \right) = \sum_{i=1}^{7} \sum_{j=1}^{7} w_i w_j Cov(R_i, R_j). \quad (4.2) \]

Since our main objective is to determine the optimal portfolio by minimizing the risk of returns, then our problem model is

\[
\begin{align*}
\text{minimize} : \quad & \sigma^2 = Var\left( \sum_{i=1}^{7} w_i R_i \right) = \sum_{i=1}^{7} \sum_{j=1}^{7} w_i w_j Cov(R_i, R_j), \\
\text{subject to} : \quad & \sum_{j=1}^{7} w_j = 1.
\end{align*}
\]

(4.3)

The next step changes the problem (4.3) to unconstrained optimization model, i.e, considering \( w_7 = 1 - w_1 - w_2 - w_3 - w_4 - w_5 - w_6 \), then the problem (4.3) changes into an unconstrained optimization model as follows:

\[ \min_{w \in \mathbb{R}^7} \sum_{i=1}^{7} \sum_{j=1}^{7} w_i w_j Cov(R_i, R_j), \quad (4.4) \]

where \( w = (w_1, w_2, w_3, w_4, w_5, w_6, 1 - w_1 - w_2 - w_3 - w_4 - w_5 - w_6) \).

The following table shows the mean, variance, and covariance values of the returns of UNVR, BBRI, TLKM, ICBP, BMRI, PGAS, and ASII stocks.

| Stocks | Mean  | Variance |
|--------|-------|----------|
| UNVR   | 0.00311 | 0.00127 |
| BBRI   | 0.00033 | 0.00273 |
| TLKM   | 0.00247 | 0.00166 |
| ICBP   | 0.00047 | 0.00142 |
| BMRI   | 0.00277 | 0.00309 |
| PGAS   | 0.00359 | 0.00667 |
| ASII   | 0.00321 | 0.00238 |

Table 2. Mean and variance of return for Seven Stocks.
Table 3. Covariance of return for seven stocks.

| Stocks | UNVR  | BBRI  | TLKM  | ICBP  | BMRI  | PGAS  | ASII  |
|--------|-------|-------|-------|-------|-------|-------|-------|
| UNVR   | 0.00127 | 0.00058 | 0.00053 | 0.00062 | 0.000906 | 0.00105 | 0.000744 |
| BBRI   | 0.00058 | 0.00273 | 0.00091 | 0.00059 | 0.00235 | 0.002341 | 0.001844 |
| TLKM   | 0.00053 | 0.00091 | 0.00166 | 0.00048 | 0.001101 | 0.001579 | 0.00089 |
| ICBP   | 0.00062 | 0.00059 | 0.00048 | 0.00142 | 0.000807 | 0.000858 | 0.000538 |
| BMRI   | 0.00091 | 0.00235 | 0.00110 | 0.00081 | 0.00309 | 0.002771 | 0.001888 |
| PGAS   | 0.00105 | 0.00234 | 0.00158 | 0.00086 | 0.002771 | 0.00667 | 0.002288 |
| ASII   | 0.00074 | 0.00184 | 0.00089 | 0.00189 | 0.001888 | 0.002288 | 0.00238 |

Table 4. Test result of NHS+, NPRP+, and HTT methods for solving problem (4.4).

| Initial Points | NHS+ | NPRP+ | HTT |
|---------------|------|-------|-----|
|               | NOI  | NOF   | CPU | NOI  | NOF | CPU | NOI  | NOF   | CPU |
| (0.1, 0.2, ..., 0.6) | 62   | 496   | 0.00680 | 61 | 497 | 0.00610 | 7 | 83 | 0.00110 |
| (0.1, ..., 0.1) | 62 | 491 | 0.00690 | 60 | 490 | 0.00610 | 12 | 126 | 0.00150 |
| (0.6, 0.5, ..., 0.1) | 57 | 448 | 0.00420 | 57 | 455 | 0.00590 | 11 | 123 | 0.00130 |
| (0.3, ..., 0.3) | 58 | 463 | 0.00680 | 58 | 472 | 0.00610 | 6 | 71 | 0.00097 |
| (1, ..., 1) | 61 | 476 | 0.00700 | 62 | 490 | 0.00430 | 12 | 127 | 0.00140 |
| (-0.1, ..., -0.1) | 64 | 502 | 0.00630 | 64 | 515 | 0.00690 | 14 | 144 | 0.00200 |
| (1.2, 1, ..., 1.2, 1) | 63 | 490 | 0.00670 | 64 | 506 | 0.00650 | 6 | 71 | 0.00140 |
| (1.001, ..., 1.001) | 61 | 476 | 0.01100 | 62 | 490 | 0.00470 | 12 | 127 | 0.00094 |
| (0.5, ..., 0.5) | 59 | 464 | 0.00310 | 60 | 481 | 0.00570 | 11 | 118 | 0.00210 |
| (7, ..., 7) | 80 | 642 | 0.00510 | 80 | 649 | 0.00870 | 13 | 143 | 0.00230 |

According to Tables 2 and 3, we may execute the problem (4.4) by choosing some random initial points and still maintaining the same parameter values according to each method. The results obtained are stated in Table 4.

From Table 4, it is clear that the HTT method more efficient than NHS+ and NPRP+ methods based on NOI, NOF, and CPU time for solving the problem (4.4). Meanwhile, the algorithm executed from each method give the same result for the value of $w$, i.e, $w_1 = 0.3877$, $w_2 = 0.3220$, $w_3 = 0.2878$, $w_4 = 0.4179$, $w_5 = -0.1642$, $w_6 = -0.0465$, and $w_7 = -0.2047$. By using the value of $w$, (4.1), and (4.2), we obtain $\mu = 0.00094$ and $\sigma^2 = 0.00074$. Finally, the selection of stock portfolios for our case with a minimum risk can be done by allocating each stock in the following proportions, i.e, UNVR 38.77%, BBRI 32.22%, TLKM 28.78%, ICBP 41.79%, BMRI −16.42%, PGAS −4.65%, and ASII −20.47% with the expected return is 0.00094 and the portfolio risk value is 0.00074. A negative sign in the proportion indicates that investor is short selling.
5. Conclusions

In this work, we presented a new hybrid CG method that guarantees sufficient descent direction and boundedness of the direction independent of the line search. In addition, the global convergence result is established under both the Wolfe and Armijo line searches. Based on the numerical results, it can be observed that the new hybrid method is more efficient and robust than other methods, providing faster and more stable convergence in most of the problems considered. These can be seen more clearly from Figures 1–3. Finally, the practical applicability of the hybrid method is also explored in risk optimization. It’s efficiency in solving portfolio selection problem was outstanding as it solves the problem with less iteration, function evaluations and CPU time compared with others.

Acknowledgments

The authors acknowledge the financial support provided by the Center of Excellence in Theoretical and Computational Science (TaCS-CoE), KMUTT. Also, the (first) author, (Dr. Auwal Bala Abubakar) would like to thank the Postdoctoral Fellowship from King Mongkut’s University of Technology Thonburi (KMUTT), Thailand. Moreover, this research project is supported by Thailand Science Research and Innovation (TSRI) Basic Research Fund: Fiscal year 2021 under project number 64A306000005. The first author acknowledge with thanks, the Department of Mathematics and Applied Mathematics at the Sefako Makgatho Health Sciences University.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. M. Al-Baali, Descent property and global convergence of the Fletcher-Reeves method with inexact line search, *IMA J. Numer. Anal.*, 5 (1985), 121–124.
2. N. Andrei, A simple three-term conjugate gradient algorithm for unconstrained optimization, *J. Comput. Appl. Math.*, 241 (2013), 19–29.
3. N. Andrei, An adaptive conjugate gradient algorithm for large-scale unconstrained optimization, *J. Comput. Appl. Math.*, 292 (2016), 83–91.
4. N. Andrei, *Nonlinear Conjugate Gradient Methods for Unconstrained Optimization*, Springer, 2020.
5. Y. H. Dai, J. Y. Han, G. H. Liu, D. F. Sun, H. X. Yin, Y. X. Yuan, Convergence properties of nonlinear conjugate gradient methods, *SIAM J. Optim.*, 10 (2000), 345–358.
6. Y. H. Dai, Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property, *SIAM J. Optim.*, 10 (1999), 177–182.
7. E. D. Dolan, J. J. Moré, Benchmarking optimization software with performance profiles, *Math. Program.*, 91 (2002), 201–213.
8. R. Fletcher, *Practical methods of optimization*, John Wiley & Sons, 2013.
9. R. Fletcher, C. M. Reeves, Function minimization by conjugate gradients, *Comput. J.*, 7 (1964), 149–154.

10. J. C. Gilbert, J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, *SIAM J. Optim.*, 2 (1992), 21–42.

11. L. Grippo, S. Lucidi, A globally convergent version of the polak-ribière conjugate gradient method, *Math. Program.*, 78 (1997), 375–391.

12. W. W. Hager, H. C. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, *SIAM J. Optim.*, 16 (2005), 170–192.

13. W. H. Hager, H. C. Zhang, Algorithm 851: CG-DESCENT, a conjugate gradient method with guaranteed descent, *ACM T. Math. Software*, 32 (2006), 113–137.

14. W. Hager, H. C. Zhang, A survey of nonlinear conjugate gradient methods, *Pac. J. Optim.*, 2 (2006), 35–58.

15. M. R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. Natl. Bur. Stand.*, 49 (1952), 409–436.

16. J. B. Jian, L. Han, X. Z. Jiang, A hybrid conjugate gradient method with descent property for unconstrained optimization, *Appl. Math. Model.*, 39 (2015), 1281–1290.

17. N. Jorge, Updating quasi-newton matrices with limited storage, *Math. Comput.*, 35 (1980), 773–782.

18. M. Li, A modified Hestense–Stiefel conjugate gradient method close to the memoryless bfgs quasi-newton method, *Optim. Method. Sofw.*, 33 (2018), 336–353.

19. M. Li, A three term polak-ribière-polyak conjugate gradient method close to the memoryless bfgs quasi-newton method, *J. Ind. Manage. Optim.*, 16 (2020), 245–260.

20. X. L. Li, J. J. Shi, X. L. Dong, J. L. Yu, A new conjugate gradient method based on quasi-newton equation for unconstrained optimization, *J. Comput. Appl. Math.*, 350 (2019), 372–379.

21. J. K. Liu, S. J. Li, New hybrid conjugate gradient method for unconstrained optimization, *Appl. Math. Comput.*, 245 (2014), 36–43.

22. Y. Liu, C. Storey, Efficient generalized conjugate gradient algorithms, part 1: Theory, *J. Optimiz. Theory. App.*, 69 (1991), 129–137.

23. J. T. Mo, N. Z. Gu, Z. X. Wei, Hybrid conjugate gradient methods for unconstrained optimization, *Optimi. Method. Sofw.*, 22 (2007), 297–307.

24. R. Pike, B. Neale, P. Linsley, *Corporate finance and investment-decisions and strategies*, Pearson Education Limited, 2006.

25. E. Polak, G. Ribiere, Note sur la convergence de méthodes de directions conjuguées, *ESAIM-Math. Model. Num.*, 3 (1969), 35–43.

26. B. T. Polyak, The conjugate gradient method in extremal problems, *USSR Comput. Math. Math. Phys.*, 9 (1969), 94–112.

27. D. F. Shanno, Conjugate gradient methods with inexact searches, *Math. Oper. Res.*, 3 (1978), 244–256.
28. R. Steven, *Introduction to the mathematics of finance: from risk management to options pricing*, Springer, 2004.

29. W. Sun, Y. X. Yuan, *Optimization theory and methods: Nonlinear programming*, Springer, 1992.

30. D. Touati-Ahmed, C. Storey, Efficient hybrid conjugate gradient techniques, *J. Optimiz. Theory App.*, 64 (1990), 379–397.

31. L. Zhang, W. J. Zhou, D. H. Li, A descent modified Polak–Ribiére–Polyak conjugate gradient method and its global convergence, *IMA J. Numer. Anal.*, 26 (2006), 629–640.

**Appendix**

Table 5. The numerical results of all methods using standard wolfe line search.

| Function | Dim | NHS+ | CPU | NPRP+ | CPU | HTT |
|----------|-----|------|-----|-------|-----|-----|
|           | NOI | NOF  |  | NOI | NOF  |  |  |
| F1       | 1000 | FAIL | FAIL | FAIL | FAIL | 48 | 211 | 0.1035 |
|          | 5000 | 29   | 144 | 0.3053 | FAIL | FAIL | 45 | 205 | 0.4427 |
|          | 10,000 | FAIL | FAIL | 28 | 133 | 0.5752 | 49 | 214 | 0.8004 |
|          | 15,000 | FAIL | FAIL | FAIL | FAIL | 45 | 205 | 1.0857 |
|          | 20,000 | 29 | 144 | 1.0183 | 31 | 143 | 1.0605 | 45 | 205 | 1.4329 |
| F2       | 1000 | 32   | 168 | 0.049 | FAIL | FAIL | 82 | 613 | 0.1332 |
|          | 5000 | 32   | 168 | 0.1757 | FAIL | FAIL | 59 | 496 | 0.3751 |
|          | 10,000 | 35 | 176 | 0.354 | 39 | 185 | 0.3593 | 87 | 628 | 1.0267 |
|          | 15,000 | FAIL | FAIL | 45 | 201 | 0.6693 | 59 | 496 | 1.1967 |
|          | 20,000 | 32 | 168 | 0.56 | 38 | 182 | 0.6532 | 59 | 496 | 1.4625 |
| F3       | 1000 | 13   | 67  | 0.0401 | FAIL | FAIL | 27 | 96  | 0.0456 |
|          | 5000 | FAIL | FAIL | FAIL | FAIL | FAIL | 27 | 96  | 0.1473 |
|          | 10,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 27 | 96  | 0.5049 |
|          | 15,000 | 13 | 67  | 0.2733 | FAIL | FAIL | 28 | 99  | 0.6252 |
|          | 20,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 28 | 99  | 0.6011 |
| F4       | 1000 | 20   | 66  | 0.3053 | FAIL | FAIL | 80 | 257 | 1.0738 |
|          | 5000 | 20   | 66  | 0.1586 | FAIL | FAIL | 80 | 257 | 0.5459 |
|          | 10,000 | 20 | 66  | 0.3257 | FAIL | FAIL | 80 | 257 | 1.1544 |
|          | 15,000 | 20 | 66  | 0.4333 | FAIL | FAIL | 81 | 260 | 1.5616 |
|          | 20,000 | 20 | 66  | 0.5468 | 36 | 113 | 0.9527 | 85 | 272 | 2.1239 |
| F5       | 1000 | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL |
|          | 5000 | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL |
|          | 10,000 | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL |
|          | 15,000 | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL |
|          | 20,000 | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL |
| F6       | 1000 | 7    | 30  | 0.0239 | FAIL | FAIL | 61 | 206 | 0.0917 |
|          | 5000 | FAIL | FAIL | FAIL | FAIL | FAIL | 74 | 255 | 0.5254 |
|          | 10,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 62 | 224 | 0.9381 |

*Continued on next page*
### Table 5. The numerical results of all methods using standard wolfe line search.

| Function | Dim | NHS+ NOI | NOF | CPU | NPRP+ NOI | NOF | CPU | HTT NOI | NOF | CPU |
|----------|-----|----------|-----|-----|----------|-----|-----|---------|-----|-----|
| F7       | 1000| 6        | 16  | 0.0081 | 6       | 16  | 0.0059 | 93     | 327 | 1.9179 |
| F7       | 5000| 6        | 16  | 0.0447 | 6       | 16  | 0.0284 | 14     | 39  | 0.0141 |
| F7       | 10,000| 6       | 16  | 0.0508 | 6      | 16  | 0.059  | 14     | 39  | 0.1035 |
| F7       | 15,000| 8       | 22  | 0.0846 | 8      | 22  | 0.0859 | 14     | 39  | 0.1331 |
| F7       | 20,000| 9       | 25  | 0.1302 | 8      | 22  | 0.1107 | 14     | 39  | 0.1755 |
| F8       | 1000| 8        | 28  | 0.0268 | 8      | 28  | 0.0132 | 15     | 54  | 0.0125 |
| F8       | 5000| 9        | 31  | 0.0587 | 8      | 28  | 0.0347 | 16     | 57  | 0.0585 |
| F8       | 10,000| 9       | 31  | 0.0915 | 8      | 28  | 0.0769 | 16     | 57  | 0.1157 |
| F8       | 15,000| 9       | 31  | 0.1094 | 8      | 28  | 0.0962 | 16     | 57  | 0.1837 |
| F8       | 20,000| 9       | 31  | 0.135  | 8      | 28  | 0.1227 | 16     | 57  | 0.21  |
| F9       | 1000| 23       | 101 | 0.0668 | 24     | 104 | 0.0324 | 22     | 99  | 0.0418 |
| F9       | 5000| 30       | 147 | 0.1632 | 28     | 137 | 0.134  | 21     | 96  | 0.0972 |
| F9       | 10,000| 30      | 147 | 0.1632 | 28     | 137 | 0.134  | 21     | 96  | 0.2199 |
| F9       | 15,000| 30      | 147 | 0.1632 | 28     | 137 | 0.134  | 21     | 97  | 0.278  |
| F9       | 20,000| 30      | 147 | 0.1632 | 28     | 137 | 0.134  | 21     | 97  | 0.3312 |
| F10      | 1000| FAIL     | FAIL | FAIL   | FAIL   | FAIL | FAIL   | 6566   | 19749 | 10.365 |
| F10      | 5000| FAIL     | FAIL | FAIL   | FAIL   | FAIL | FAIL   | 9027   | 27132 | 59.7488 |
| F10      | 10,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | 9027   | 27132 | 59.7488 |
| F10      | 15,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | 9027   | 27132 | 59.7488 |
| F10      | 20,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | 9027   | 27132 | 59.7488 |
| F11      | 1000| FAIL     | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F11      | 5000| FAIL     | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F11      | 10,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F11      | 15,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F11      | 20,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F12      | 1000| 10       | 42  | 0.0375 | 11     | 45  | 0.0198 | 19     | 71  | 0.0153 |
| F12      | 5000| 10       | 42  | 0.0717 | 11     | 45  | 0.0536 | 20     | 74  | 0.0798 |
| F12      | 10,000| 10     | 42  | 0.0953 | 11     | 45  | 0.1071 | 20     | 74  | 0.1711 |
| F12      | 15,000| 10     | 42  | 0.1423 | 11     | 45  | 0.1519 | 20     | 74  | 0.2083 |
| F12      | 20,000| 10     | 42  | 0.1598 | 11     | 45  | 0.1659 | 21     | 77  | 0.2929 |
| F13      | 1000| FAIL     | FAIL | FAIL   | 32     | 159 | 0.0477 | 59     | 1847 | 0.3214 |
| F13      | 5000| FAIL     | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F13      | 10,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F13      | 15,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F13      | 20,000| FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F14      | 50    | FAIL    | FAIL | FAIL   | FAIL   | FAIL | FAIL   | FAIL   | FAIL  | FAIL  |
| F14      | 100    | FAIL   | FAIL | FAIL   | FAIL   | FAIL | FAIL   | 24     | 110  | 0.013  |

Continued on next page
Table 5. The numerical results of all methods using standard wolfe line search.

| Function | Dim | NHS+ | CPU | NRP+ | CPU | HTT | CPU |
|----------|-----|------|-----|------|-----|-----|-----|
|          | NOI | NOF  |     | NOI  | NOF  |     |     |
| 200      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 31  | 171 | 0.031 |
| 300      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 26  | 80  | 0.0886 |
| 500      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 27  | 83  | 0.2039 |
| F15      | 1000 | 23   | 64  | 0.0224 | FAIL | FAIL | FAIL | 24  | 75  | 0.0274 |
| 5000     | 27   | 74   | 0.0825 | FAIL | FAIL | FAIL | FAIL | 26  | 80  | 0.0886 |
| 10,000   | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 27  | 83  | 0.2039 |
| 15,000   | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 27  | 83  | 0.2489 |
| 20,000   | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 29  | 88  | 0.3151 |
| F16      | 50   | 65   | 214 | 0.0109 | 62   | 251  | 0.011 | 112 | 382 | 0.014 |
| 100      | 94   | 361  | 0.0339 | FAIL | FAIL | FAIL | 155 | 534 | 0.0319 |
| 200      | 141  | 518  | 0.0552 | 141  | 519  | 0.0489 | 222 | 777 | 0.074 |
| 300      | FAIL | FAIL | 178  | 600  | 0.0793 | 260  | 924  | 0.1116 |
| 500      | 228  | 818  | 0.1383 | 243  | 812  | 0.1226 | 317 | 1151 | 0.1591 |
| F17      | 50   | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 26  | 84  | 0.0086 |
| 100      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 26  | 84  | 0.009 |
| 200      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 26  | 84  | 0.0244 |
| 300      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 26  | 87  | 0.0216 |
| 500      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 2443 | 125676 | 15.9497 |
| F18      | 50   | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 31  | 93  | 0.0206 |
| 100      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 32  | 112 | 0.0126 |
| 200      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 32  | 117 | 0.0197 |
| 300      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 39  | 499 | 0.0556 |
| 500      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 32  | 97  | 0.0349 |
| F19      | 50   | 66   | 198  | 0.0098 | 65   | 195  | 8.40E-03 | 66  | 198  | 7.30E-03 |
| 100      | 141  | 423  | 0.035 | 140  | 420  | 2.98E-02 | 141 | 423  | 3.58E-02 |
| 200      | 297  | 891  | 0.0839 | 296  | 888  | 7.57E-02 | 297 | 891  | 7.60E-02 |
| 300      | 453  | 1359 | 0.1582 | 454  | 1362 | 0.1759 | 455 | 1365 | 0.1431 |
| 500      | 768  | 2304 | 0.3216 | 770  | 2310 | 0.2792 | 3022 | 9066 | 1.0477 |
| F20      | 50   | 38   | 114  | 0.0068 | 38   | 114  | 0.0108 | 38  | 114  | 0.0307 |
| 100      | 56   | 168  | 0.0174 | 56   | 168  | 0.0223 | 56  | 168  | 0.0247 |
| 200      | 81   | 243  | 0.0373 | 81   | 243  | 0.0368 | 81  | 243  | 0.0373 |
| 300      | 101  | 303  | 0.0464 | 101  | 303  | 0.0435 | 101 | 303  | 0.0523 |
| 500      | 131  | 393  | 0.0706 | 131  | 393  | 0.0693 | 131 | 393  | 0.0693 |
| F21      | 100  | 27   | 243  | 0.0411 | 47   | 317  | 0.028 | 208 | 2671 | 0.1393 |
| 500      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 431 | 4682 | 0.6918 |
| 1000     | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 742 | 5992 | 1.5017 |
| 3000     | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 2402 | 12075 | 7.9879 |
| 5000     | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 4345 | 21114 | 20.7249 |
| F22      | 10   | 11   | 41   | 0.2096 | 10   | 38   | 0.0237 | 18  | 64  | 0.0022 |

Continued on next page
Table 5. The numerical results of all methods using standard wolfe line search.

| Function | Dim | NHS+ | NPRP+ | HTT |
|----------|-----|------|------|-----|
|          | NOI | NOF | CPU  | NOI | NOF | CPU  | NOI | NOF | CPU  |
| 50       | 12  | 49  | 0.0115 | 11  | 45  | 0.009 | 35  | 383  | 0.2059 |
| 80       | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 18  | 72  | 0.0054 |
| 100      | 11  | 45  | 0.004 | FAIL | FAIL | FAIL | 57  | 1635 | 0.0704 |
| 300      | FAIL | FAIL | FAIL | FAIL | FAIL | FAIL | 35  | 485  | 0.2809 |
| F23      | 1000 | 1   | 3    | 0.0127 | 1    | 3    | 0.0027 | 1    | 3    | 1.60E-03 |
|          | 5000 | 1   | 3    | 0.0168 | 1    | 3    | 0.0089 | 1    | 3    | 0.0061 |
|          | 10,000 | 1   | 3    | 0.0165 | 1    | 3    | 0.0134 | 1    | 3    | 0.0129 |
|          | 15,000 | 1   | 3    | 0.0217 | 1    | 3    | 0.0198 | 1    | 3    | 0.0183 |
|          | 20,000 | 1   | 3    | 0.028  | 1    | 3    | 0.0296 | 1    | 3    | 0.0219 |
| F24      | 1000 | 136 | 408  | 0.0978 | 136   | 408   | 0.0969 | 136   | 408   | 0.0976 |
|          | 5000 | 311 | 933  | 0.7694 | 311   | 933   | 0.7563 | 406   | 1218  | 0.975 |
|          | 10,000 | 443 | 1329 | 2.4757 | 443   | 1329   | 2.3572 | 443   | 1329   | 2.3631 |
|          | 15,000 | 544 | 1632 | 4.6718 | 544   | 1632   | 4.4034 | 544   | 1632   | 4.264 |
|          | 20,000 | 630 | 1890 | 6.8562 | 630   | 1890   | 8.618  | 630   | 1890   | 6.5118 |
| F25      | 1000 | FAIL | FAIL | FAIL | FAIL | FAIL | 21   | 77   | 0.0481 |
|          | 5000 | FAIL | FAIL | FAIL | FAIL | FAIL | 21   | 77   | 0.1557 |
|          | 10,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 23   | 82   | 0.3301 |
|          | 15,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 23   | 82   | 0.4332 |
|          | 20,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 23   | 82   | 0.5819 |
| F26      | 1000 | 3   | 155  | 0.0804 | FAIL  | FAIL | FAIL | FAIL | FAIL | FAIL |
|          | 5000 | 3   | 155  | 0.3545 | FAIL  | FAIL | FAIL | FAIL | FAIL | FAIL |
|          | 10,000 | 3   | 155  | 0.6368 | FAIL  | FAIL | FAIL | FAIL | FAIL | FAIL |
|          | 15,000 | 3   | 155  | 0.8296 | FAIL  | FAIL | FAIL | FAIL | FAIL | FAIL |
|          | 20,000 | 3   | 155  | 1.0828 | FAIL  | FAIL | FAIL | FAIL | FAIL | FAIL |
| F27      | 1000 | 14  | 91   | 0.0472 | 14    | 94    | 0.0324 | 12    | 87    | 0.0278 |
|          | 5000 | 16  | 97   | 0.3356 | 14    | 94    | 0.0844 | 13    | 90    | 0.0859 |
|          | 10,000 | 14  | 91   | 0.2063 | 14    | 94    | 0.205  | 13    | 90    | 0.1935 |
|          | 15,000 | FAIL | FAIL | FAIL | 14    | 94    | 0.2503 | 13    | 90    | 0.2522 |
|          | 20,000 | FAIL | FAIL | FAIL | 14    | 94    | 0.3559 | 13    | 90    | 0.2913 |
| F28      | 1000 | FAIL | FAIL | FAIL | FAIL | FAIL | 11   | 163  | 0.05  |
|          | 5000 | FAIL | FAIL | FAIL | FAIL | FAIL | 11   | 164  | 0.1523 |
|          | 10,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 12   | 173  | 0.3325 |
|          | 15,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 12   | 166  | 0.4909 |
|          | 20,000 | FAIL | FAIL | FAIL | FAIL | FAIL | 14   | 228  | 0.7552 |
| F29      | 200  | FAIL | FAIL | FAIL | FAIL | FAIL | 7    | 31   | 0.2276 |
|          | 400  | FAIL | FAIL | FAIL | FAIL | FAIL | 7    | 23   | 0.0132 |
|          | 600  | FAIL | FAIL | FAIL | FAIL | FAIL | 7    | 21   | 0.008 |
|          | 800  | 5    | 15   | 0.0083 | 5    | 15    | 0.0272 | 7    | 26    | 0.0115 |
|          | 1000 | FAIL | FAIL | FAIL | FAIL | FAIL | 7    | 36    | 0.2199 |

Continued on next page
### Table 5. The numerical results of all methods using standard Wolfe line search.

| Function | Dim  | NHS+ | NPRP+ | HTT   |
|----------|------|------|-------|-------|
|          | NOI  | NOF  | CPU   | NOI   | NOF  | CPU   | NOI  | NOF  | CPU   |
| F30      | 1000 | 35   | 199   | 0.2817| FAIL | FAIL | 78   | 376  | 0.1087|
|          | 5000 | 35   | 199   | 0.1804| FAIL | FAIL | 85   | 397  | 0.404 |
|          | 10,000| 37  | 205   | 0.3754| FAIL | FAIL | 85   | 397  | 0.6769|
|          | 15,000| FAIL| FAIL  | FAIL  | FAIL | FAIL | 85   | 397  | 1.0886|
|          | 20,000| FAIL| FAIL  | FAIL  | FAIL | FAIL | 85   | 397  | 1.3642|
| F31      | 50   | FAIL | FAIL  | FAIL  | FAIL | FAIL | 25   | 404  | 0.0131|
|          | 100  | FAIL | FAIL  | FAIL  | FAIL | FAIL | 24   | 408  | 0.0248|
|          | 200  | FAIL | FAIL  | FAIL  | FAIL | FAIL | 776  | 39378| 2.1743|
|          | 300  | FAIL | FAIL  | FAIL  | FAIL | FAIL | 56   | 1903 | 0.156 |
|          | 500  | FAIL | FAIL  | FAIL  | FAIL | FAIL | 1236 | 63303| 6.2995|
| F32      | 50   | FAIL | FAIL  | FAIL  | FAIL | FAIL | 14   | 49   | 0.0224|
|          | 100  | FAIL | FAIL  | FAIL  | FAIL | FAIL | 14   | 69   | 0.0112|
|          | 200  | FAIL | FAIL  | FAIL  | FAIL | FAIL | 19   | 277  | 0.027 |
|          | 300  | FAIL | FAIL  | FAIL  | FAIL | FAIL | 168  | 7913 | 1.0719|
| F33      | 1000 | 140  | 420   | 0.1027| 140  | 420  | 0.0965| 140  | 420  | 0.0993|
|          | 5000 | 320  | 960   | 0.8334| 320  | 960  | 0.7689| 320  | 960  | 0.8576|
|          | 10,000| 456 | 1368  | 2.5879| 456  | 1358 | 2.6651| 456  | 1358 | 2.5723|
|          | 15,000| 562 | 1686  | 4.4691| 562  | 1686 | 6.0655| 561  | 1683 | 4.3236|
|          | 20,000| 651 | 1953  | 6.6436| 651  | 1953 | 13.6986| 650  | 1950 | 7.2173|
| F34      | 1000 | 76   | 625   | 0.2802| 76   | 625  | 0.2711| 3    | 27   | 0.0206|
|          | 5000 | 81   | 695   | 1.645 | 81   | 695  | 1.5016| 3    | 27   | 0.0844|
|          | 10,000| 83  | 724   | 2.8429| 83   | 724  | 2.9767| 3    | 27   | 0.1274|
|          | 15,000| 84  | 739   | 4.2936| 84   | 739  | 5.8732| 3    | 27   | 0.2009|
|          | 20,000| 85  | 754   | 6.1778| 85   | 754  | 10.2159| 3    | 27   | 0.2372|