# A NEW HYBRID BFGS-CG METHOD UNDER INEXACT LINE SEARCH AND ITS GLOBAL CONVERGENCE

Alireza Hosseini Dehmiry[1] and Maryam Kargarfard[2]

*This paper is an attempt to propose a novel hybrid BFGS-CG method with an inexact line search (ILS) formula for solving unconstrained optimization prob-lems. This ILS technique has recently been used to achieve the convergence of the BFGS method for non-convex functions. We establish the global convergence of our proposed algorithm for general functions and under standard assumptions. Numerical results confirm the effectiveness of our approach, and show that the proposed hybrid method is comparable to other similar techniques, and is even more efficient.*

**Keywords:** Unconstrained optimization, BFGS method, Global convergence, Conjugate gradient method

MSC 2010: 90C53, 90C30.

## 1. Introduction

An unconstrained optimization problem is a problem of the form

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

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is an objective function.

In this paper, the following assumptions are considered.

I) The level set $D := \{x \mid f(x) \leqslant f(x_0)\}$ is bounded for all $x_0 \in dom(f)$.

II) $f$ is a twice continuously differentiable function. Moreover, its gradient is Lipschitz continuous with constant $L > 0$, in the sense that

$$\|\nabla f(x) - \nabla f(y)\| \leqslant L\|x - y\|, \qquad \forall x, y \in D. \tag{2}$$

Problem (1) can be solved using various approaches. Most of these algorithms use the following iterative formula.

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, \ldots, n.$$

Here, $x_k$ is the $k$th iteration point, $d_k$ is the improvement direction, and $\alpha_k$ is the step length. Hereafter, $f(x_k)$ and $g(x_k)$ (that is, the gradient of $f(x)$ at $x_k$) are denoted by $f_k$ and $g_k$, respectively. The step length is calculated via a local optimization of the objective function, known as a line search, where the direction and starting point are given. Nonlinear descent optimization strategies heavily rely on line search techniques.

In this regard, Armijo and Wolfe's conditions are the foundation of a wide range of minimization approaches that use line search tactics. The generalized Wolfe conditions [5]

[1]Department of Mathematics, Vali-e-Asr University of Rafsanjan, Rafsanjan, Iran, e-mail: dehmiry@vru.ac.ir
[2]Department of Mathematics, Vali-e-Asr University of Rafsanjan, Rafsanjan, Iran

are

$$\begin{gathered} f(x_k + \alpha_k d_k) - f(x_k) \leqslant \delta \alpha_k g_k^T d_k, \\ \sigma_1 g_k^T d_k \leqslant g_{k+1}^T d_k \leqslant \sigma_2 g_k^T d_k, \end{gathered} \tag{3}$$

where $0 < \delta < \sigma_1 < 1$ and $\sigma_2 \leqslant 0$. The special cases $\sigma_1 = -\sigma_2$ and $\sigma_2 = 0$ correspond to the strong and standard Wolfe conditions, respectively. The first condition is known as the Armijo condition, which guarantees a sufficient decrease of the value of the objective function, whereas the other condition is called the curvature condition, which guarantees the non-acceptance of short steplengths.

## 2. The Conjugate-Gradient and Quasi-Newton Methods

The conjugate-gradient (CG) method is one of the well-known iterative methods of solving (1). This method was originally suggested by Hestenes and Stiefel for minimizing convex quadratic functions with symmetric and positive definite matrices [11, 12]. Later on, it was extended to nonlinear unconstrained optimization problems by Fletcher and Reeves [10]. Due to the moderate memory requirement, this method is very suitable for solving large-scale unconstrained optimization problems.

In the CG method, the search direction $d_k$ is defined as

$$d_k = \begin{cases} -g_k, & k = 0, \\ -g_k + \beta_k d_{k-1}, & k > 0, \end{cases} \tag{4}$$

where $\beta_k$ is known as the CG update parameter. There are many ways to calculate $\beta_k$. Some well-known formulas used for this purpose are as follows.

$$\begin{aligned} \beta_k^{HS} &= \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}, & \beta_k^{FR} &= \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, & \beta_k^{PRP} &= \frac{g_k^T y_{k-1}}{\|g_{k-1}\|^2}, \\ \beta_k^{CD} &= -\frac{\|g_k\|^2}{d_{k-1}^T g_{k-1}}, & \beta_k^{LS} &= -\frac{g_k^T y_{k-1}}{d_{k-1}^T g_{k-1}}, & \beta_k^{DY} &= \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}}. \end{aligned}$$

Here, $\|.\|$ denotes the $\ell_2$ norm and $y_{k-1} := g_k - g_{k-1}$. The corresponding methods are known as the Hesten and Stiefel (HS) [11], Fletcher and Reeves (FR) [10], Polak, Ribiere and Polyak (PRP) [15, 16], Conjugate-Descent (CD) [9], Liu and Storey (LS) [14] and, Dai and Yuan (DY) [4] algorithms, respectively.

When $f$ is a strong convex quadratic function, and an exact line search (ELS) is chosen, all the six choices of the update parameter are equivalent in theory [3]. In non-quadratic functions, each choice of the update parameter leads us to a method with different numerical performance. By comparing these methods, we can see that the PRP, LS, and HS algorithms have good numerical results, but fail to have good theory convergence. However, the FR, DY, and CD methods have a completely opposite behavior.

The quasi-Newton method is another popular algorithm for solving unconstrained optimization problems. The main difference between the quasi-Newton method and the conjugate-gradient method is in the use of an approximation of the Hessian matrix for finding the improvement direction [2]. Hence, the conjugate-gradient methods are preferred especially when the problem size is large and memory is limited.

In the quasi-Newton methods, the improvement direction $d_k$ is obtained by solving the equation

$$d_k = -H_k g_k, \quad k \geqslant 0, \tag{5}$$

where $H_k$ is an approximation of the inverse Hessian. Usually, $H_k$ is updated by the famous Broyden- Fletcher-Goldfarb-Shanno (BFGS) formula

$$H_{k+1} = H_k + \left(1 + \frac{y_k^T H_k y_k}{y_k^T s_k}\right) \frac{s_k s_k^T}{y_k^T s_k} - \frac{s_k y_k^T H_k + H_k y_k s_k^T}{y_k^T s_k}, \tag{6}$$

where $s_k = x_{k+1} - x_k = \alpha_k d_k$, $y_k = g_{k+1} - g_k$, and $H_0$ is an arbitrarily given $n \times n$ positive definite matrix. Another similar update formula is the Davidon–Fletcher–Powell (DFP) formula, which approximates the inverse Hessian matrix as follows.

$$H_{k+1} = H_k + \frac{s_k s_k^T}{s_k^T y_k} - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k}. \tag{7}$$

The convex combination of (6) and (7) is known as the Broyden class of the quasi-Newton update formula.

## 3. Motivation and the Proposed Algorithm

The number of iterations and the number of function evaluations in quasi-Newton methods are better compared to CG methods. Moreover, under the ILS technique, the global convergence of the BFGS algorithm has been established for general functions. These advantages motivated us to combine the BFGS algorithm and the CG method (namely, (5) and (4)) with the DY update parameter as follows.

$$d_k = \begin{cases} -H_k g_k, & k = 0. \\ -H_k g_k + \beta_k^{DY} d_{k-1}, & k > 0. \end{cases} \tag{8}$$

In this paper, we use a new, modified version of the ILS technique that guarantees the global convergence of the BFGS methods for general functions. This ILS formula was recently proposed by Dehmiry [6, 7]. It can be written as

$$\begin{aligned} f_{k+1} &\leqslant f_k + \delta \alpha_k g_k^T d_k - \tfrac{\delta \alpha_k^2}{2M} \|d_k\|^2, \\ |g_{k+1}^T d_k| &\leqslant -\sigma g_k^T d_k + \tfrac{\delta \alpha_k}{M} \|d_k\|^2, \end{aligned} \tag{9}$$

where $\delta \in (0, \frac{1}{2})$, $\sigma \in (\delta, 1)$, and $M$ is a sufficiently large positive number. In what follows, the proposed algorithm is presented for the sake of completeness.

**Algorithm 3.1** : A hybrid BFGS-CG algorithm

Step 0: Choose an initial point $x_0 \in \mathbb{R}^n$, constants $\delta \in (0, \frac{1}{2})$, $\sigma \in (\delta, 1)$ and $\varepsilon \in (0, 1)$, a sufficiently large positive number $M$, and an initial $n \times n$ positive definite matrix $H_0$. Set $k := 0$.

Step 1: If $\|g_k\| \leqslant \varepsilon$, stop.

Step 2: Compute $d_k$ by (8).

Step 3: Find $\alpha_k$ that satisfies the inequalities (9) and

$$\alpha_k < -\frac{M(1-\sigma) g_k^T d_k}{2\delta \|d_k\|^2}. \tag{10}$$

Step 4: Set $x_{k+1} := x_k + \alpha_k d_k$.
Step 5: Update $H_{k+1}$ using the DFP formula (7).
Step 6: Set $k := k + 1$ and go to Step 1.

In Section 4, we prove some facts about Algorithm 3.1. These allow us to conclude that the algorithm is well-defined and globally convergent.

## 4. The Main Results

Consider the following function and its derivative to show reasonability of (9):

$$\varphi(\alpha) := f_{k+1} - f_k - \delta\alpha g_k^T d_k + \frac{\delta\alpha^2}{2M}\|d_k\|^2,$$
$$\varphi'(\alpha) = g_{k+1}^T d_k - \delta g_k^T d_k + \frac{\delta\alpha}{M}\|d_k\|^2.$$

**Theorem 4.1.** *Suppose that $f$ is twice continuously differentiable and bounded from below. If $g_k^T d_k \leqslant 0$, then there exists a constant $0 < \alpha < \infty$ satisfying* (9) *and* (10).

*Proof.* From the assumption it follows that $\varphi(0) = 0$ and if $\alpha \to +\infty$, then $\varphi(\alpha) \to +\infty$. For any sufficiently small $\alpha > 0$, we have

$$\begin{aligned}\varphi(\alpha) &= f_{k+1} - f_k - \delta\alpha g_k^T d_k + \frac{\delta\alpha^2}{2M}\|d_k\|^2 \\ &= (f_k + \alpha g_k^T d_k + o(\alpha)) - f_k - \delta\alpha g_k^T d_k + \frac{\delta\alpha^2}{2M}\|d_k\|^2 \\ &= \alpha((1-\delta)g_k^T d_k + \frac{\delta\alpha}{2M}\|d_k\|^2) + o(\alpha) < 0.\end{aligned}$$

Since $\varphi \in \mathbb{C}^2$, there exists a constant $\rho > 0$ such that $\varphi(\rho) = 0$ and $\varphi(\alpha) < 0$ for all $\alpha \in (0, \rho)$ and there exists an interval $I := (\alpha^*, \alpha')$ such that $\varphi'(\alpha) > 0$ for all $\alpha \in I$, where $\alpha^*$ is a local minimizer of $\varphi$ on $(0, \rho)$ and $\alpha^* < \alpha' < \rho$.
Now, from $\sigma > \delta$ and $g_k^T d_k < 0$, we can write

$$g_{k+1}^T d_k \geqslant \delta g_k^T d_k - \frac{\delta\alpha}{M}\|d_k\|^2 \geqslant \sigma g_k^T d_k - \frac{\delta\alpha}{M}\|d_k\|^2, \quad \forall\alpha \in I.$$

For sufficiently large values of $M$, one ensures that any $\alpha \in I$ satisfies the condition (10) too. □

The following lemma shows that the update formula (7) can preserve the positive definiteness of $\{H_k\}$.

**Lemma 4.1.** *Let $\{H_k\}$ be the sequence generated by Algorithm 3.1. Then, $H_k$ is positive definite for all $k$.*

*Proof.* Since $s_k = H_{k+1}y_k$, it is sufficient to show that $y_k^T s_k > 0$. From (10) we obtain

$$\alpha_k < -\frac{M(1-\sigma)g_k^T d_k}{2\delta\|d_k\|^2} \;\Rightarrow\; (\sigma-1)g_k^T d_k - \frac{2\delta\alpha_k}{M}\|d_k\|^2 > 0. \tag{11}$$

By (9) and (11),

$$\begin{aligned} y_k^T d_k &= (g_{k+1} - g_k)^T d_k \\ &\geqslant \sigma g_k^T d_k - \frac{\delta \alpha_k}{M} \|d_k\|^2 - g_k^T d_k \\ &= -(1-\sigma) g_k^T d_k - \frac{\delta \alpha_k}{M} \|d_k\|^2 \\ &> \frac{2\delta \alpha_k}{M} \|d_k\|^2 - \frac{\delta \alpha_k}{M} \|d_k\|^2 \\ &= \frac{\delta \alpha_k}{M} \|d_k\|^2 > 0. \end{aligned} \tag{12}$$

Hence, $y_k^T s_k = \alpha_k y_k^T d_k > 0$. □

**Lemma 4.2.** *Suppose that $\{x_k\}$ and $\{f_k\}$ are the sequences generated by Algorithm 3.1. Then, $g_k^T d_k < 0$ holds for all $k \geqslant 0$ and therefore, $\{f_k\}$ is a strictly descending sequence.*

*Proof.* The proof of the first part is by induction on $k$. By Lemma 4.1 and (8), the proof is straightforward for $k = 0$. Let $k > 0$. By (8),

$$\begin{aligned} g_{k+1}^T d_{k+1} &= g_{k+1}^T(-H_{k+1} g_{k+1} - \beta_{k+1}^{DY} d_k) \\ &= -g_{k+1}^T H_{k+1} g_{k+1} - \beta_{k+1}^{DY} g_{k+1}^T d_k \\ &\leqslant -g_{k+1}^T H_{k+1} g_{k+1} - \beta_{k+1}^{DY}(-\sigma g_k^T d_k + \tfrac{\delta \alpha_k}{M} \|d_k\|^2) \\ &< -g_{k+1}^T H_{k+1} g_{k+1} + (\tfrac{1+\sigma}{2}) \beta_{k+1}^{DY} g_k^T d_k \\ &\leqslant 0, \end{aligned}$$

where the last inequality follows from Lemma 4.1 and the induction hypothesis. To prove the final part of this lemma, we use (9) and (17) to obtain

$$f_{k+1} \leqslant f_k + \delta \alpha_k g_k^T d_k - \frac{\delta \alpha_k^2}{2M} \|d_k\|^2 = f_k + \delta \alpha_k (\overbrace{g_k^T d_k - \frac{\alpha_k}{2M} \|d_k\|^2}^{¡0}) < f_k.$$

□

In the next lemma, we find a lower bound for $-g_k^T d_k$. This bound will be used to prove the global convergence of our proposed algorithm.

**Lemma 4.3.** *Let $\{d_k\}$ and $\{g_k\}$ be the sequences generated by Algorithm 3.1. Then, $\frac{1}{4}(1+\sigma)\|g_k\|^2$ is a lower bound for $-g_k^T d_k$.*

*Proof.* By (9) and (10) we can write

$$\begin{aligned} g_k^T d_{k-1} &\geqslant \sigma g_{k-1}^T d_{k-1} - \frac{\delta \alpha_{k-1}}{M} \|d_{k-1}\|^2 \\ &\geqslant \sigma g_{k-1}^T d_{k-1} - \frac{1}{2}(\sigma - 1) g_{k-1}^T d_{k-1} \\ &= \frac{1}{2}(1+\sigma) g_{k-1}^T d_{k-1}. \end{aligned} \tag{13}$$

On the other hand, using (9) and (10) we obtain

$$\begin{aligned} y_{k-1}^T d_{k-1} &= (g_k - g_{k-1})^T d_{k-1} \\ &\leqslant -(1+\sigma) g_{k-1}^T d_{k-1} + \frac{\delta \alpha_{k-1}}{M} \|d_{k-1}\|^2 \\ &\leqslant -(1+\sigma) g_{k-1}^T d_{k-1} \\ &\leqslant -2 g_{k-1}^T d_{k-1}. \end{aligned}$$

So,

$$\frac{1}{y_{k-1}^T d_{k-1}} \geqslant \frac{1}{-2 g_{k-1}^T d_{k-1}}. \tag{14}$$

Now, by (5), (13) and (14) we conclude that

$$\begin{aligned} -g_k^T d_k &= g_k^T H_k g_k - \beta_k^{DY} g_k^T d_{k-1} \\ &= g_k^T H_k g_k - \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}} g_k^T d_{k-1} \\ &\geqslant -\frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}} g_k^T d_{k-1} \\ &\geqslant -\frac{\frac{1}{2}(1+\sigma) g_{k-1}^T d_{k-1}}{-2 g_{k-1}^T d_{k-1}} \|g_k\|^2 \\ &= \frac{1}{4}(1+\sigma)\|g_k\|^2. \end{aligned} \tag{15}$$

□

**Theorem 4.2.** *Assume that $\{g_k\}$ and $\{d_k\}$ are the sequences generated by Algorithm 3.1. Then,*

$$\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < \infty. \tag{16}$$

*Proof.* By using (9), (2) and the Cauchy-Schwarz inequality we obtain

$$\begin{aligned} L &\geqslant \frac{\|g_{k+1} - g_k\|}{\|x_{k+1} - x_k\|} = \frac{\|g_{k+1} - g_k\|}{\|\alpha_k d_k\|} = \frac{\|g_{k+1} - g_k\| \|d_k\|}{\alpha_k \|d_k\|^2} \\ &\geqslant \frac{|(g_{k+1} - g_k)^T d_k|}{\alpha_k \|d_k\|^2} \\ &\geqslant \frac{(g_{k+1} - g_k)^T d_k}{\alpha_k \|d_k\|^2} \\ &\geqslant \frac{\sigma g_k^T d_k - \frac{\delta \alpha_k}{M}\|d_k\|^2 - g_k^T d_k}{\alpha_k \|d_k\|^2} \\ &> -\frac{(1-\sigma) g_k^T d_k}{2\alpha_k \|d_k\|^2}. \end{aligned}$$

Thus, $\alpha_k > -\frac{(1-\sigma) g_k^T d_k}{2L\|d_k\|^2}$. Now, by assumption (**I**) and (9) we can write

$$f_k - f_{k+1} \geqslant -\delta \alpha_k g_k^T d_k + \frac{\delta \alpha_k^2}{2M}\|d_k\|^2 \geqslant -\delta \alpha_k g_k^T d_k,$$

$$\sum_{k=0}^{\infty} -\delta \alpha_k g_k^T d_k \leqslant \sum_{k=0}^{\infty} (f_k - f_{k+1}) = f_1 - \lim_{k\to\infty} f_k < \infty.$$

Hence,

$$\frac{\delta(1-\sigma)}{2L}\sum_{k=0}^{\infty}\frac{(g_k^T d_k)^2}{\|d_k\|^2} < \infty. \tag{17}$$

□

Combining the previous theorem with Lemma 4.3, we obtain the following immediate corollary.

**Corollary 4.1.** *(Global Convergence) Suppose that $\{g_k\}$ is the sequence generated by Algorithm 3.1. Then,*

$$\lim_{k\to\infty}\|g_k\| = 0. \tag{18}$$

*Proof.* By Theorem 4.2 and Lemma 4.3,

$$\|g_k\|^2 \leqslant -\frac{4}{1+\sigma}g_k^T d_k. \tag{19}$$

Thus

$$\sum_{k=0}^{\infty}\frac{\|g_k\|^4}{\|d_k\|^2} < \sum_{k=0}^{\infty}\frac{16}{(1+\sigma)^2}\frac{(g_k^T d_k)^2}{\|d_k\|^2} < \infty. \tag{20}$$

□

## 5. Numerical results

In this section, we present some numerical experiments of Algorithm 3.1 and the similar algorithms that use the (ILS) formula (namely, the normal BFGS method [5], the modified Yuan-Wei-Lu method [17], and the Hui-Fukushima method (HF) [13]) to evaluate their performance. A set of test problems are taken from [1] and are listed in Table 1 together with the related initial points.

All tests were coded in MATLAB R2020a, and were run on a PC with a 2.70 GHz CPU, and 12.0 GB of memory running the Windows 10 OS.
The parameters of Algorithm 3.1 were chosen as follows.

- **Parameters:** $\varepsilon = 1e-6$, $\delta = \frac{1}{3}$, $\sigma = \frac{2}{3}$, and $M = 10^4$.
- **Dimensions of the variable $x$:** 10, 100, 1000, and 3000.
- **Stop rule:** Since the results of iteration number are stable, we chose the Himmeblau stop rule [18]. This can be described as follows.
  Let
  $$stop1 := \begin{cases} \frac{|f_k - f_{k+1}|}{|f_k|}, & |f_k| > 1e-5, \\ |f_k - f_{k+1}|, & otherwise. \end{cases}$$
  For every problem, if each of the conditions $\|g_k\| \leqslant \varepsilon$ or $stop1 < 1e-5$ is satisfied, the program stops. This program also stops when the number of iterations is greater than 1000.

Table 1: Test problems.

| No | Name function | $x_0$ |
|---|---|---|
| 1 | Rosenbrock Func. | $[0, 0, \ldots, 0]$ |
| 2 | Extended Trigonometric Func. | $[0.2, 0.2, \ldots, 0.2]$ |
| 3 | Extended Rosenbrock Func. | $[0.5, -2, \ldots, 0.5, -2]$ |

*Continued on next page*

Table 1 – *Continued from previous page*

| **No** | **Name function** | $x_0$ |
|---|---|---|
| 4 | Generalized Rosenbrock Func. | $[-1, 2, 1, \ldots, -1, 2, 1]$ |
| 5 | Extended White and Holst Func. | $[-1, 2, 1, \ldots, -1, 2, 1]$ |
| 6 | Extended Beale Func. | $[1, 0.8, \ldots, 1, 0.8]$ |
| 7 | Extended Penalty Func. | $[1, 2, \ldots, n]$ |
| 8 | Perturbed Quadratic Func. | $[0.5, 0.5, \ldots, 0.5]$ |
| 9 | Generalized Tridiagonal 1 Func. | $[2, 2, \ldots, 2]$ |
| 10 | Extended Tridiagonal 1 Func. | $[2, 2, \ldots, 2]$ |
| 11 | Extended Frigonometric 1 Func. | $[2, 2, \ldots, 2]$ |
| 12 | Diagonal 4 Func. | $[1, 1, \ldots, 1]$ |
| 13 | Extended Himmelblau Func. | $[1, 1, \ldots, 1]$ |
| 14 | Generalized PSC1 Func. | $[3, 0.1, \ldots, 3, 0.1]$ |
| 15 | Extended Powell Func. | $[3, -1, 0, 1, \ldots, 3, -1, 0, 1]$ |
| 16 | Full Hessian FH1 Func. | $[0.01, 0.01, \ldots, 0.01]$ |
| 17 | Extended Cliff Func. | $[0, -1, \ldots, 0, -1]$ |
| 18 | Perturbed Quadratic Diagonal Func. | $[0.5, 0.5, \ldots, 0.5]$ |
| 19 | Quadratic QF1 Func. | $[1, 1, \ldots, 1]$ |
| 20 | Extended Quadratic Penalty QP1 Func. | $[1, 1, \ldots, 1]$ |
| 21 | Extended Quadratic Penalty QP2 Func. | $[1, 1, \ldots, 1]$ |
| 22 | Quadratic QF2 Func. | $[0.5, 0.5, \ldots, 0.5]$ |
| 23 | FLETCHCR Func. (CUTE) | $[0, 0, \ldots, 0]$ |
| 24 | TRIDIA Func. (CUTE) | $[1, 1, \ldots, 1]$ |
| 25 | ARWHEAD Func. (CUTE) | $[1, 1, \ldots, 1]$ |
| 26 | NONDIA Func. (CUTE) | $[-1, -1, \ldots, -1]$ |
| 27 | Broyden Tridiagonal Func. | $[-1, -1, \ldots, -1]$ |
| 28 | LIARWHD Func. (CUTE) | $[4, 4, \ldots, 4]$ |
| 29 | POWER Func. (CUTE) | $[1, 1, \ldots, 1]$ |
| 30 | ENGVAL1 Func. (CUTE) | $[2, 2, \ldots, 2]$ |
| 31 | EDENSCH Func. (CUTE) | $[0, 0, \ldots, 0]$ |
| 32 | NONSCOMP Func. (CUTE) | $[3, 3, \ldots, 3]$ |
| 33 | LIARWHD Func. (CUTE) | $[4, 4, \ldots, 4]$ |
| 34 | DIXON3DQ Func. (CUTE) | $[-1, -1, \ldots, -1]$ |
| 35 | SINQUAD Func. (CUTE) | $[0.1, 0.1, \ldots, 0.1]$ |

Table 2: Numerical results for the BFGS-CG method.

| $No$ | $Dim$ | $T_{BFGS-CG}$ | $NI$ | $T_{BFGS}$ | $NI$ | $T_{YWL}$ | $NI$ | $T_{HF}$ | $NI$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 10 | 0.012500 | 15 | 0.015625 | 23 | 0.014063 | 23 | 0.040625 | 23 |
| 1 | 100 | 0.059375 | 12 | 0.043750 | 11 | 0.064062 | 11 | 0.054688 | 11 |
| 1 | 1000 | 2.943750 | 10 | 2.353125 | 8 | 2.278125 | 8 | 2.421875 | 8 |
| 1 | 3000 | 57.34218 | 8 | 49.09218 | 7 | 49.72656 | 7 | 49.09687 | 7 |
| 2 | 10 | 0.015625 | 15 | 0.021875 | 23 | 0.026562 | 23 | 0.029687 | 18 |
| 2 | 100 | 0.185938 | 8 | 0.190625 | 9 | 0.232813 | 9 | 0.240625 | 9 |

*Continued on next page*

Table 2 – *Continued from previous page*

| $No$ | $Dim$ | $T_{BFGS-CG}$ | $NI$ | $T_{BFGS}$ | $NI$ | $T_{YWL}$ | $NI$ | $T_{HF}$ | $NI$ |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 1000 | 9.878125 | 6 | 6.571875 | 5 | 9.535937 | 5 | 6.342187 | 5 |
| 2 | 3000 | 39.43593 | 4 | 35.76406 | 4 | 38.08437 | 4 | 35.30625 | 4 |
| 3 | 10 | 0.003125 | 6 | 0.003125 | 6 | 0.012500 | 6 | 0.009375 | 6 |
| 3 | 100 | 0.057813 | 6 | 0.060937 | 6 | 0.073438 | 6 | 0.087500 | 6 |
| 3 | 1000 | 1.510938 | 5 | 1.539063 | 5 | 1.756250 | 5 | 1.560938 | 5 |
| 3 | 3000 | 34.29375 | 5 | 34.34687 | 5 | 35.68125 | 5 | 34.40312 | 5 |
| 4 | 10 | 0.006250 | 10 | 0.023438 | 10 | 0.007813 | 10 | 0.015625 | 10 |
| 4 | 100 | 0.060937 | 7 | 0.068750 | 6 | 0.087500 | 6 | 0.090625 | 6 |
| 4 | 1000 | 1.528125 | 5 | 1.573438 | 5 | 1.857813 | 5 | 1.678125 | 5 |
| 4 | 3000 | 35.42812 | 5 | 35.42187 | 5 | 37.69218 | 5 | 35.25468 | 5 |
| 5 | 10 | 0.003125 | 5 | 0.004687 | 5 | 0.009375 | 5 | 0.009375 | 5 |
| 5 | 100 | 0.075000 | 4 | 0.076563 | 4 | 0.090625 | 4 | 0.100000 | 4 |
| 5 | 1000 | 4.078125 | 4 | 4.131250 | 4 | 6.290625 | 4 | 4.062500 | 4 |
| 5 | 3000 | 51.65156 | 4 | 51.66093 | 4 | 71.49843 | 4 | 51.14687 | 4 |
| 6 | 10 | 0.004687 | 6 | 0.012500 | 9 | 0.009375 | 9 | 0.010937 | 9 |
| 6 | 100 | 0.120313 | 6 | 0.120313 | 6 | 0.145313 | 6 | 0.114062 | 6 |
| 6 | 1000 | 5.237500 | 5 | 5.403125 | 5 | 8.307813 | 5 | 5.312500 | 5 |
| 6 | 3000 | 68.05937 | 5 | 67.66093 | 5 | 93.85000 | 5 | 67.38281 | 5 |
| 7 | 10 | 0.004687 | 6 | 0.009375 | 7 | 0.012500 | 7 | 0.012500 | 7 |
| 7 | 100 | 0.025000 | 3 | 0.029687 | 3 | 0.026562 | 3 | 0.026562 | 3 |
| 7 | 1000 | 0.389062 | 2 | 0.406250 | 2 | 0.468750 | 2 | 0.453125 | 2 |
| 7 | 3000 | 8.807813 | 2 | 8.809375 | 2 | 9.195313 | 2 | 8.795312 | 2 |
| 8 | 10 | 0.007813 | 17 | 0.015625 | 27 | 0.026562 | 27 | 0.054688 | 27 |
| 8 | 100 | 0.064062 | 7 | 0.060937 | 6 | 0.082812 | 6 | 0.065625 | 6 |
| 8 | 1000 | 1.585938 | 5 | 1.595312 | 5 | 1.823438 | 5 | 1.646875 | 5 |
| 8 | 3000 | 26.12031 | 4 | 26.13593 | 4 | 28.73125 | 4 | 26.22968 | 4 |
| 9 | 10 | 0.042188 | 37 | 0.045312 | 41 | 0.046875 | 44 | 0.050000 | 44 |
| 9 | 100 | 0.146875 | 9 | 0.146875 | 9 | 0.201563 | 9 | 0.148438 | 9 |
| 9 | 1000 | 2.339062 | 5 | 2.360937 | 5 | 3.428125 | 5 | 2.564062 | 5 |
| 9 | 3000 | 41.83125 | 5 | 41.88281 | 5 | 48.61406 | 5 | 41.27812 | 5 |
| 10 | 10 | 0.004687 | 10 | 0.025000 | 31 | 0.004687 | 31 | 0.045312 | 31 |
| 10 | 100 | 0.170313 | 9 | 0.178125 | 9 | 0.170313 | 9 | 0.185938 | 9 |
| 10 | 1000 | 5.706250 | 6 | 4.367188 | 5 | 6.734375 | 5 | 4.593750 | 5 |
| 10 | 3000 | 60.64687 | 5 | 60.48750 | 5 | 60.48750 | 5 | 60.46406 | 5 |
| 11 | 10 | 0.004687 | 5 | 0.009375 | 5 | 0.012500 | 5 | 0.004687 | 5 |
| 11 | 100 | 0.045312 | 5 | 0.048438 | 5 | 0.070313 | 5 | 0.045312 | 5 |
| 11 | 1000 | 1.121875 | 4 | 1.121875 | 4 | 1.335938 | 4 | 1.207812 | 4 |
| 11 | 3000 | 25.91406 | 4 | 26.00000 | 4 | 27.18437 | 4 | 25.82343 | 4 |
| 12 | 10 | 0.014063 | 15 | 0.025000 | 31 | 0.028125 | 31 | 0.045312 | 31 |
| 12 | 100 | 0.064062 | 7 | 0.062500 | 6 | 0.068750 | 6 | 0.048438 | 6 |
| 12 | 1000 | 1.092188 | 4 | 1.521875 | 5 | 1.629687 | 5 | 1.489062 | 5 |
| 12 | 3000 | 25.46093 | 4 | 25.65781 | 4 | 26.36406 | 4 | 25.77343 | 4 |
| 13 | 10 | 0.006250 | 15 | 0.017188 | 13 | 0.017188 | 13 | 0.009375 | 13 |

*Continued on next page*

Table 2 – *Continued from previous page*

| $No$ | $Dim$ | $T_{BFGS-CG}$ | $NI$ | $T_{BFGS}$ | $NI$ | $T_{YWL}$ | $NI$ | $T_{HF}$ | $NI$ |
|---|---|---|---|---|---|---|---|---|---|
| 13 | 100 | 0.059375 | 7 | 0.067187 | 7 | 0.089063 | 7 | 0.075000 | 7 |
| 13 | 1000 | 1.475000 | 5 | 1.481250 | 5 | 1.706250 | 5 | 1.523438 | 5 |
| 13 | 3000 | 34.08750 | 5 | 33.97187 | 5 | 35.30468 | 5 | 34.22812 | 5 |
| 14 | 10 | 0.007813 | 12 | 0.012500 | 13 | 0.014063 | 13 | 0.021875 | 13 |
| 14 | 100 | 0.140625 | 7 | 0.140625 | 7 | 0.193750 | 7 | 0.190625 | 7 |
| 14 | 1000 | 6.564063 | 5 | 6.626563 | 5 | 10.70312 | 10 | 6.676563 | 5 |
| 14 | 3000 | 43.37656 | 4 | 77.94218 | 5 | 106.3359 | 5 | 77.98906 | 5 |
| 15 | 10 | 0.010937 | 6 | 0.018750 | 12 | 0.025000 | 12 | 0.020313 | 12 |
| 15 | 100 | 0.223438 | 12 | 0.173437 | 9 | 0.240625 | 9 | 0.179688 | 9 |
| 15 | 1000 | 9.826562 | 9 | 6.178125 | 6 | 9.362500 | 6 | 5.600000 | 6 |
| 15 | 3000 | 96.36562 | 7 | 79.18906 | 6 | 109.5046 | 6 | 75.29218 | 6 |
| 16 | 10 | 0.006250 | 9 | 0.012500 | 9 | 0.007813 | 9 | 0.012500 | 9 |
| 16 | 100 | 0.103125 | 5 | 0.148438 | 5 | 0.132813 | 5 | 0.104688 | 5 |
| 16 | 1000 | 13.11406 | 2 | 13.16093 | 2 | 20.94375 | 2 | 11.98125 | 2 |
| 16 | 3000 | 297.6812 | 2 | 297.0953 | 2 | 492.7656 | 2 | 341.1718 | 2 |
| 17 | 10 | 0.000625 | 2 | 0.001250 | 2 | 0.001250 | 2 | 0.001563 | 2 |
| 17 | 100 | 0.014063 | 2 | 0.014063 | 2 | 0.023438 | 2 | 0.006250 | 2 |
| 17 | 1000 | 0.515625 | 2 | 0.545312 | 2 | 0.704688 | 2 | 0.543750 | 2 |
| 17 | 3000 | 10.33125 | 2 | 10.68593 | 2 | 12.11875 | 2 | 9.962500 | 2 |
| 18 | 10 | 0.031250 | 17 | 0.051562 | 25 | 0.071875 | 25 | 0.170313 | 25 |
| 18 | 100 | 0.198437 | 7 | 0.182812 | 6 | 0.264062 | 6 | 0.201563 | 6 |
| 18 | 1000 | 6.551563 | 4 | 4.146875 | 3 | 5.696875 | 3 | 4.062500 | 3 |
| 18 | 3000 | 66.52187 | 3 | 66.12343 | 3 | 78.35781 | 3 | 65.69843 | 3 |
| 19 | 10 | 0.014063 | 17 | 0.023438 | 27 | 0.021875 | 27 | 0.029687 | 27 |
| 19 | 100 | 0.051562 | 7 | 0.059375 | 7 | 0.089063 | 7 | 0.068750 | 7 |
| 19 | 1000 | 1.385938 | 5 | 1.437500 | 5 | 1.671875 | 5 | 1.545313 | 5 |
| 19 | 3000 | 25.43750 | 4 | 25.53750 | 4 | 26.32968 | 4 | 25.84375 | 4 |
| 20 | 10 | 0.004687 | 11 | 0.018750 | 16 | 0.018750 | 16 | 0.015625 | 16 |
| 20 | 100 | 0.053125 | 7 | 0.062500 | 6 | 0.068750 | 6 | 0.081250 | 6 |
| 20 | 1000 | 1.225000 | 4 | 1.651563 | 5 | 1.954687 | 5 | 1.753125 | 5 |
| 20 | 3000 | 26.85937 | 4 | 26.84375 | 4 | 28.93125 | 4 | 26.92968 | 4 |
| 21 | 10 | 0.003125 | 8 | 0.009375 | 8 | 0.007813 | 8 | 0.009375 | 8 |
| 21 | 100 | 0.076563 | 5 | 0.076563 | 5 | 0.101563 | 5 | 0.076563 | 5 |
| 21 | 1000 | 3.303125 | 4 | 3.443750 | 4 | 4.620312 | 4 | 3.376562 | 4 |
| 21 | 3000 | 44.97031 | 4 | 45.49062 | 4 | 55.20156 | 4 | 44.93750 | 4 |
| 22 | 10 | 0.009375 | 17 | 0.018750 | 32 | 0.021875 | 32 | 0.053125 | 32 |
| 22 | 100 | 0.065625 | 7 | 0.065625 | 7 | 0.081250 | 7 | 0.073438 | 7 |
| 22 | 1000 | 1.575000 | 5 | 1.614062 | 5 | 1.857813 | 5 | 1.650000 | 5 |
| 22 | 3000 | 26.46875 | 4 | 26.81250 | 4 | 27.50625 | 4 | 26.27656 | 4 |
| 23 | 10 | 0.007813 | 12 | 0.015625 | 11 | 0.017188 | 11 | 0.017188 | 11 |
| 23 | 100 | 0.084375 | 8 | 0.081250 | 8 | 0.104688 | 8 | 0.084375 | 8 |
| 23 | 1000 | 2.020313 | 6 | 2.428125 | 6 | 2.428125 | 6 | 2.176563 | 6 |
| 23 | 3000 | 44.84062 | 6 | 45.13593 | 6 | 47.39375 | 6 | 44.66406 | 6 |

*Continued on next page*

Table 2 – *Continued from previous page*

| $No$ | $Dim$ | $T_{BFGS-CG}$ | $NI$ | $T_{BFGS}$ | $NI$ | $T_{YWL}$ | $NI$ | $T_{HF}$ | $NI$ |
|---|---|---|---|---|---|---|---|---|---|
| 24 | 10 | 0.006250 | 19 | 0.017188 | 33 | 0.025000 | 33 | 0.057813 | 33 |
| 24 | 100 | 0.070313 | 7 | 0.071875 | 7 | 0.082812 | 7 | 0.065625 | 7 |
| 24 | 1000 | 1.590625 | 5 | 1.629687 | 5 | 1.806250 | 5 | 1.635938 | 5 |
| 24 | 3000 | 26.26093 | 4 | 26.41562 | 4 | 27.82031 | 4 | 26.63437 | 4 |
| 25 | 10 | 0.004687 | 13 | 0.021875 | 15 | 0.020313 | 15 | 0.023438 | 15 |
| 25 | 100 | 0.081250 | 8 | 0.087500 | 8 | 0.092188 | 8 | 0.079687 | 8 |
| 25 | 1000 | 1.962500 | 6 | 1.959375 | 6 | 2.289063 | 6 | 2.070313 | 6 |
| 25 | 3000 | 35.03906 | 5 | 34.70000 | 5 | 36.84531 | 5 | 34.94218 | 5 |
| 26 | 10 | 0.009375 | 8 | 0.014063 | 9 | 0.014063 | 9 | 0.010937 | 9 |
| 26 | 100 | 0.056250 | 6 | 0.040625 | 5 | 0.073438 | 5 | 0.042188 | 5 |
| 26 | 1000 | 1.529688 | 5 | 1.509375 | 5 | 1.789063 | 5 | 1.573438 | 5 |
| 26 | 3000 | 25.81406 | 4 | 25.72343 | 4 | 27.30156 | 4 | 26.14218 | 4 |
| 27 | 10 | 0.012500 | 15 | 0.015625 | 24 | 0.017188 | 24 | 0.051562 | 24 |
| 27 | 100 | 0.087500 | 8 | 0.093750 | 8 | 0.101563 | 8 | 0.107813 | 8 |
| 27 | 1000 | 2.178125 | 6 | 2.203125 | 6 | 2.603125 | 6 | 2.189062 | 6 |
| 27 | 3000 | 36.02343 | 5 | 36.10156 | 5 | 38.96562 | 5 | 36.15781 | 5 |
| 28 | 10 | 0.003125 | 9 | 0.015625 | 13 | 0.020313 | 13 | 0.015625 | 13 |
| 28 | 100 | 0.062500 | 6 | 0.067187 | 6 | 0.084375 | 6 | 0.065625 | 6 |
| 28 | 1000 | 1.628125 | 5 | 1.745313 | 5 | 1.996875 | 5 | 1.729688 | 5 |
| 28 | 3000 | 35.04281 | 5 | 35.05000 | 5 | 37.72500 | 5 | 36.05937 | 5 |
| 29 | 10 | 0.017188 | 14 | 0.026562 | 27 | 0.026562 | 27 | 0.034375 | 27 |
| 29 | 100 | 0.040625 | 5 | 0.056250 | 6 | 0.082812 | 6 | 0.057813 | 6 |
| 29 | 1000 | 1.104688 | 4 | 0.728125 | 3 | 0.865625 | 3 | 0.756250 | 3 |
| 29 | 3000 | 17.02812 | 3 | 16.95468 | 3 | 17.52187 | 3 | 17.20781 | 3 |
| 30 | 10 | 0.004687 | 14 | 0.025000 | 16 | 0.017188 | 16 | 0.018750 | 16 |
| 30 | 100 | 0.062500 | 7 | 0.068750 | 7 | 0.109375 | 7 | 0.087500 | 7 |
| 30 | 1000 | 1.696875 | 5 | 1.723437 | 5 | 2.003125 | 5 | 1.687500 | 5 |
| 30 | 3000 | 35.91781 | 5 | 35.92031 | 5 | 38.39375 | 5 | 35.76406 | 5 |
| 31 | 10 | 0.012500 | 15 | 0.014063 | 15 | 0.017188 | 15 | 0.029687 | 20 |
| 31 | 100 | 0.081250 | 9 | 0.084375 | 9 | 0.121875 | 9 | 0.259375 | 8 |
| 31 | 1000 | 1.834375 | 6 | 1.914063 | 6 | 2.142187 | 6 | 0.042188 | 5 |
| 31 | 3000 | 42.70937 | 6 | 42.94687 | 6 | 44.51406 | 6 | 101.9750 | 5 |
| 32 | 10 | 0.006250 | 12 | 0.017188 | 11 | 0.018750 | 11 | 0.046875 | 11 |
| 32 | 100 | 0.065625 | 7 | 0.067187 | 7 | 0.067187 | 7 | 0.075000 | 7 |
| 32 | 1000 | 1.646875 | 5 | 1.720312 | 5 | 1.960938 | 5 | 1.621875 | 5 |
| 32 | 3000 | 34.97656 | 5 | 26.41406 | 4 | 27.87968 | 4 | 25.76093 | 4 |
| 33 | 10 | 0.012500 | 10 | 0.018750 | 14 | 0.023438 | 14 | 0.025000 | 14 |
| 33 | 100 | 0.064062 | 7 | 0.071875 | 7 | 0.107813 | 7 | 0.092188 | 7 |
| 33 | 1000 | 1.659375 | 5 | 1.737500 | 5 | 2.059375 | 5 | 1.682813 | 5 |
| 33 | 3000 | 35.60468 | 5 | 35.24531 | 5 | 37.45781 | 5 | 35.50937 | 5 |
| 34 | 10 | 0.014063 | 18 | 0.017188 | 27 | 0.026562 | 27 | 0.039063 | 27 |
| 34 | 100 | 0.078125 | 8 | 0.079687 | 8 | 0.098437 | 8 | 0.085938 | 8 |
| 34 | 1000 | 1.960938 | 6 | 1.562500 | 5 | 1.798437 | 5 | 1.489062 | 5 |

*Continued on next page*

Table 2 – *Continued from previous page*

| $No$ | $Dim$ | $T_{BFGS-CG}$ | $NI$ | $T_{BFGS}$ | $NI$ | $T_{YWL}$ | $NI$ | $T_{HF}$ | $NI$ |
|---|---|---|---|---|---|---|---|---|---|
| 34 | 3000 | 34.11562 | 5 | 34.54843 | 5 | 35.77031 | 5 | 34.52343 | 5 |
| 35 | 10 | 0.009375 | 14 | 0.015625 | 30 | 0.025000 | 30 | 0.059375 | 30 |
| 35 | 100 | 0.170313 | 12 | 0.131250 | 9 | 0.171875 | 9 | 0.139063 | 9 |
| 35 | 1000 | 6.531250 | 9 | 4.475000 | 7 | 6.223438 | 7 | 4.621875 | 7 |
| 35 | 3000 | 64.36406 | 7 | 53.50468 | 6 | 61.51093 | 6 | 53.35625 | 6 |

The numerical results are listed in Table 2. Moreover, the results are depicted in Figures 1 and 2, in which a performance measure introduced by Dolan and More [8] is also employed.

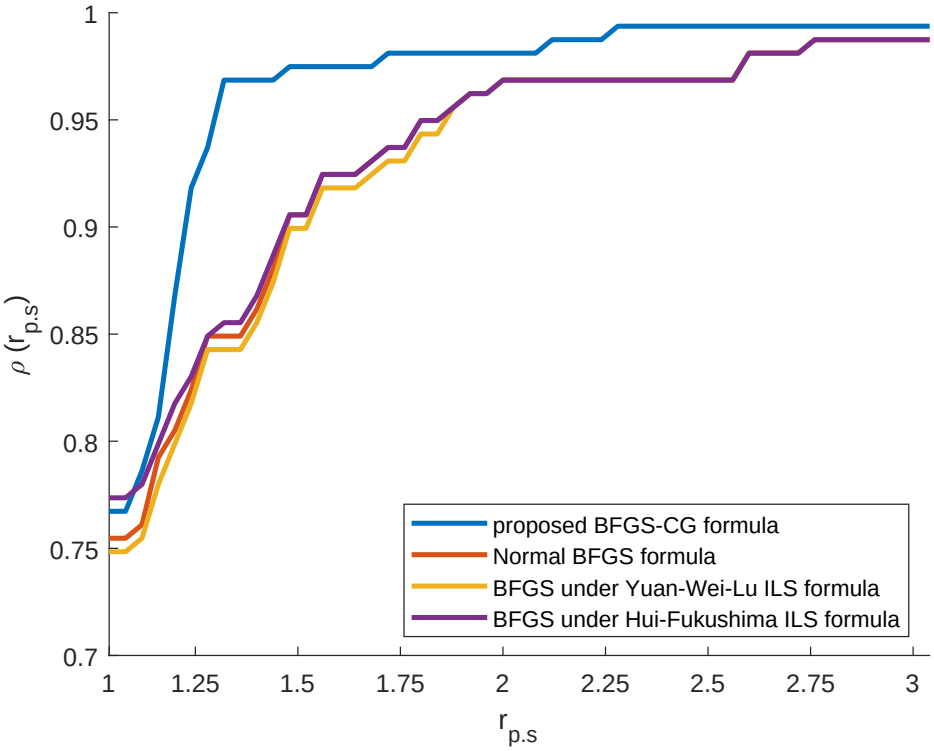


FIGURE 1. Performance profiles of hybrid BFGS-CG method (The number of iterations).

From Figures 1 and 2 we observe that for most problems, the proposed method performs much better than the other three famous methods. It follows that Algorithm 3.1 is efficient and can compete with other algorithms.

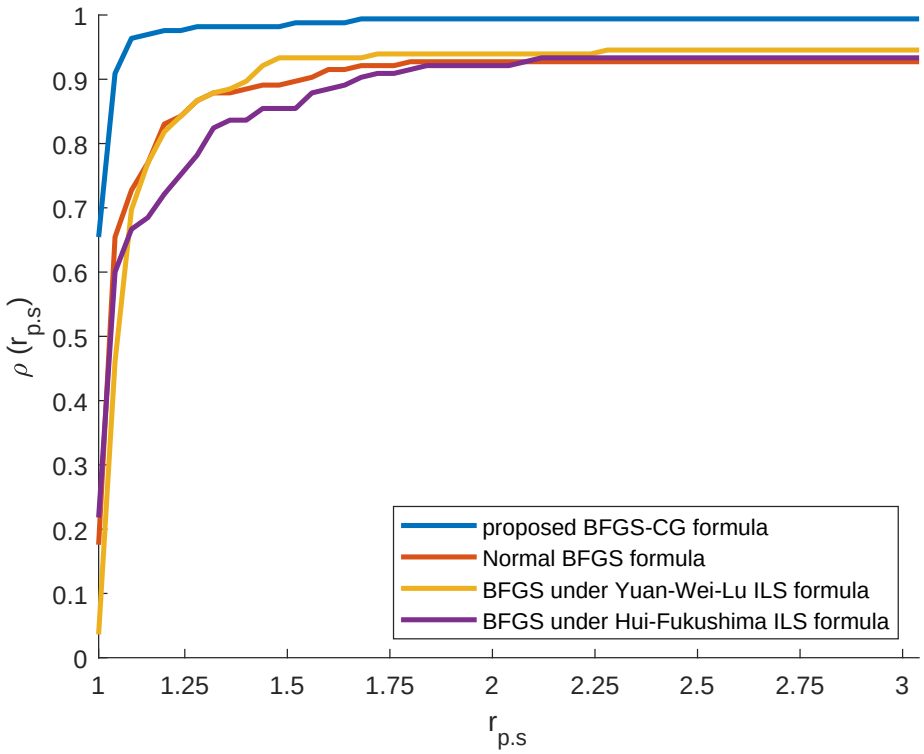


FIGURE 2. Performance profiles of hybrid BFGS-CG method (CPU time).

## 6. Conclusion

This study proposed a new hybrid BFGS-CG method, one that used a novel ILS technique, and proved its global convergence for general functions. We compared it with three well-known algorithms according to the CPU time and the number of iterations. TThe numerical results were depicted in Figures 1 and 2, and indicated the faster convergence of the new algorithm to the answer in most test problems. According to these results, we found that the proposed method was acceptably effi-cient and promising. Modification of this formula for achieving a higher convergence rate may be a good idea for future research.

## REFERENCES

[1] Andrei, N.: An unconstrained optimization test functions collection. Advanced Modeling and Optimization **10**(1), 147–161 (2008)
[2] Andrei, N.: Nonlinear conjugate gradient methods for unconstrained optimization. Springer (2020)
[3] Dai, Y.H.: Nonlinear conjugate gradient methods. Wiley Encyclopedia of Operations Research and Management Science (2010)
[4] Dai, Y.H., Yuan, Y.: A nonlinear conjugate gradient method with a strong global convergence property. SIAM Journal on optimization **10**(1), 177–182 (1999)
[5] Dai, Y.H., YUAN, Y.X.: Convergence properties of the Fletcher-Reeves method. IMA Journal of Numerical Analysis **16**(2), 155–164 (1996)
[6] Dehmiry, A.H.: The global convergence of the BFGS method under a modified Yuan-Wei-Lu line search technique. Numerical Algorithms **84**(2), 781–793 (2019)
[7] Dehmiry, A.H., Hamzehnejad, M.: Modify the linear search formula in the BFGS method to achieve global convergence. Journal of New Researches in Mathematics **5**(21), 37–46 (2019)
[8] Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Mathematical programming **91**(2), 201–213 (2002)
[9] Fletcher, R.: Practical methods of optimization. John Wiley & Sons (2013)
[10] Fletcher, R., Reeves, C.M.: Function minimization by conjugate gradients. The computer journal **7**(2), 149–154 (1964)
[11] Hestenes, M.R., Stiefel, E.: Methods of conjugate gradients for solving linear systems. Journal of Research of the National Bureau of Standards **49**(6), 409–435 (1952)
[12] Hestenes M.R.: Conjugate Direction Methods in Optimization. USA: Springer, (1980)
[13] Li, D.H., Fukushima, M.: On the global convergence of the BFGS method for nonconvex unconstrained optimization problems. SIAM Journal on Optimization **11**(4), 1054–1064 (2001)
[14] Liu, Y., Storey, C.: Efficient generalized conjugate gradient algorithms, part 1: theory. Journal of optimization theory and applications **69**(1), 129–137 (1991)
[15] Polak, E., Ribiere, G.: Note sur la convergence de méthodes de directions conjuguées. ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique **16**(3), 35–43 (1969)
[16] Polyak, B.T.: The conjugate gradient method in extremal problems. USSR Computational Mathematics and Mathematical Physics **9**(4), 94–112 (1969)
[17] Yuan, G., Lu, J., Wang, Z.: The PRP conjugate gradient algorithm with a modified WWP line search and its application in the image restoration problems. Applied Numerical Mathematics **152**, 1–11 (2020)
[18] Yuan, Y.X., Sun, W., et al.: Theory and methods of optimization. Science Press of China, Beijing (1999)