Shahid Chamran University of Ahvaz

Journal of Advanced Mathematical Modeling
ISSN (print): 2251-8088, ISSN (on-line): 2645-6141
Vol. 16 No. 2 (2026), pp. 5-22.
DOI: 10.22055/jamm.2026.48572.2336

A NEW THREE-TERM CONJUGATE GRADIENT METHOD AND ITS APPLICATION TO TRAINING NEURAL NETWORKS

FAEZE SADAT HOSSEINI-MEIGHAN, ALI ASHRAFI*

Communicated by: Sohrab Effati

Abstract. Jiang et al. introduced a new family of three-term conjugate gradient techniques that are theoretically effective and numerically efficient. based on their study, we propose a new three-term conjugate gradient algorithm as a modified version of the Hestenes-Stiefel method. Numerical experiments were performed on a set of CUTEr test functions as well as on a neural network. They show that the proposed method has promising performance from a computational perspective.

1. Introduction

Conjugate Gradient (CG) algorithms are widely recognized as powerful and efficient tools for solving large-scale unconstrained optimization problems of the form $\min_{x\in\mathbb{R}^n} f(x)$, where f is a continuously differentiable objective function. Their popularity stems from several key advantages compared to other optimization methods like steepest descent: they require significantly less memory storage, making them suitable for high-dimensional problems, and often exhibit faster convergence rates due to their ability to incorporate second-order information implicitly. The iterative process of a CG method typically starts with an initial point $x_0 \in \mathbb{R}^n$, and subsequent iterates are computed by $x_{k+1} = x_k +$

MSC(2020): 49M37, 90C06, 90C30.

Keywords: Nonlinear programming, unconstrained optimization, Three term conjugate gradient method, Neural network.

Received: 15 December 2024, Accepted: 9 June 2026.

*Corresponding author.

$\alpha_k d_k,\ k \geq 0$. The search direction d_k is defined by $d_0 = -g_0$ and $d_{k+1} = -g_{k+1} + \beta_k d_k$, for $k = 0, 1, ...$, where $g_k = \nabla f(x_k)$ is the gradient at x_k, and β_k is a scalar known as the CG parameter.

The step-size $\alpha_k > 0$ along the direction d_k is determined by standard line search procedures, most commonly the Wolfe conditions. These conditions are crucial for ensuring sufficient decrease in the objective function and a suitable curvature condition for stable convergence. The weak Wolfe conditions are defined as:

$$\begin{cases} f(x_k + \alpha_k d_k) - f(x_k) & \leq \delta \alpha_k g_k^T d_k, \\ \nabla f(x_k + \alpha_k d_k)^T d_k & \geq \sigma g_k^T d_k, \end{cases} \tag{1.1}$$

while the strong Wolfe line search imposes a tighter constraint on the curvature condition:

$$|\nabla f(x_k + \alpha_k d_k)^T d_k| \leq \sigma |g_k^T d_k|, \tag{1.2}$$

with $0 < \delta < \sigma < 1$ [8].

Among the numerous classical CG algorithms, the Hestenes-Stiefel (HS) method, defined by $\beta_k^{\mathrm{HS}} = \dfrac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}$ [8] (where $y_k = g_{k+1} - g_k$ and $\|.\|$ denotes the Euclidean norm), is particularly noted for its computational efficiency and an automatic restart feature that effectively prevents jamming. However, a known limitation of the standard HS method is its inability to inherently guarantee the sufficient descent condition, given by

$$g_k^T d_k \leq -\rho \|g_k\|^2,\ k \geq 0, \tag{1.3}$$

for some positive constant ρ. This inequality is fundamental for establishing global convergence of CG algorithms [8]. Consequently, a significant body of research has focused on developing descent variants of the HS method. For instance, Zhang et al. [18] introduced a three-term version of the HS algorithm (HSZZL) with a modified search direction:

$$d_{k+1}^{\mathrm{HSZZL}} = -g_{k+1} + \beta_k^{\mathrm{HS}} d_k - \gamma_k y_k,$$

where $\gamma_k = \dfrac{g_{k+1}^T d_k}{d_k^T y_k}$. A notable property of the HSZZL method is that it intrinsically satisfies the sufficient descent condition (1.3), independent of the line search. This makes it a robust benchmark for methods aiming to achieve global convergence.

Recent developments in hybrid and three-term conjugate gradient methods have further expanded the applicability of CG algorithms in large-scale optimization and deep learning. Studies such as [1, 9, 12] have highlighted innovative approaches that leverage

hybridization to enhance convergence properties while reducing computational overhead, particularly in the context of training deep neural networks. These advancements underline the significance of adaptive CG methods in tackling complex optimization landscapes inherent in machine learning tasks.

The primary objective of this paper is to propose a novel three-term hybrid conjugate gradient method, denoted as MHS+, designed to enhance both the theoretical properties and numerical efficiency of CG algorithms. Specifically, we aim to combine the robust descent properties of methods like HSZZL with the computational advantages of the HS scheme. Furthermore, given the pervasive application of optimization in modern machine learning, particularly in the training of neural networks, we will thoroughly investigate the performance of MHS+ in this context. Training neural networks inherently involves minimizing large-scale, non-convex loss functions, where memory efficiency and stable convergence are paramount. Therefore, demonstrating the efficacy of MHS+ on such problems will underscore its practical significance.

2. **A new hybrid three-term conjugate gradient method**

In order to benefit from the strong convergence properties of some classical methods as well as the numerical advantages of some other methods, many hybrid conjugate gradient methods have been proposed to solve unconstrained optimization problems [2, 3, 5, 15, 10, 17]. In this regard, Jiang et al. (JLYJ) [11] proposed a family of three-term CG hybrid methods based on a convex combination as follows:

$$
\begin{aligned}
d_{k+1} &= -g_{k+1} + (1-\lambda_k)\beta_k^* d_k + \lambda_k \xi_k g_k, \\
\beta_k^* &= \max\left\{\min\left\{\beta_k, \frac{\|g_{k+1}\|^2}{d_k^T y_k}\right\}, 0\right\}, \\
\lambda_k &= \frac{|g_{k+1}^T d_k|}{\|g_{k+1}\|\|d_k\|}, \\
\xi_k &= -\tau_k \frac{g_{k+1}^T g_k}{\|g_k\|^2},
\end{aligned} \tag{2.1}
$$

where $\tau_k > 0$ is a constant and instead of β_k, any classical conjugate gradient parameter can be used. In this article, we use β_k^{HS}. This choice ensures compatibility with the descent-guaranteed framework proposed in [18] and facilitates integration with modern

adaptive schemes [10]. The JLYJ method possesses a sufficient descent property, independent of the line search, which leads to global convergence under the Wolfe conditions (1.1)−(1.2).

inspired by their method, we propose the following method:

$$
\begin{aligned}
d_{k+1}^{MHS+} &= -g_{k+1} + (1-\mu_k)\beta_k^{HS+} d_k + \mu_k \xi_k g_k, \\
\beta_k^{HS+} &= \max\{\frac{g_{k+1}^T y_k}{d_k^T y_k}, 0\}, \\
\bar{\mu}_k &= \frac{2\beta_k^{HS+}\|d_k\|^2 - (2\xi_k+\gamma_k)\beta_k^{HS+} d_k^T y_k + 2\xi_k^2\|y_k\|^2}{\beta_k^{HS+}\gamma_k d_k^T y_k - (1+\xi_k)\gamma_k\|y_k\|^2}, \\
\mu_k &= \min\{\max\{\bar{\mu}_k, 0\}, 1\}, \\
\tau_k &= (\frac{1}{4}\frac{g_{k+1}^T d_k}{d_k^T y_k} \times \frac{\|g_k\|^2}{g_{k+1}^T g_k}) + 1, \\
\xi_k &= -\tau_k \frac{g_{k+1}^T g_k}{\|g_k\|^2},
\end{aligned}
\tag{2.2}
$$

where the parameter $\bar{\mu}_k$ and ξ_k are obtained by solving the following least squares problem:

$$\min_{\mu_k,\xi_k} \|d_{k+1}^{MHS+} - d_{k+1}^{HSZZL}\|^2.$$

The novelty of MHS+ lies in deriving its parameters μ_k and ξ_k directly via a least squares fit to HSZZL's search direction, thereby preserving its strong sufficient descent property while allowing adaptive balancing between conjugate and gradient terms. This strategy is consistent with recent hybrid CG designs [15] where parameter estimation is treated as an optimization problem in itself.

The development of the MHS+ method is fundamentally driven by the need to ensure the *sufficient descent property*, a critical characteristic for guaranteeing the global convergence of conjugate gradient algorithms. This property, formally stated as $g_k^T d_k \leq -\rho\|g_k\|^2$ for some constant $\rho > 0$, ensures that each iteration makes significant progress towards the minimum.

Derivation of $\bar{\mu}_k$ from Least Squares Problem. We consider the minimization problem

$$\min_{\mu_k,\,\xi_k} \|d_{k+1}^{\mathrm{MHS+}} - d_{k+1}^{\mathrm{HSZZL}}\|^2, \tag{2.3}$$

where

$$d_{k+1}^{\mathrm{MHS+}} = -g_{k+1} + (1-\mu_k)\beta_k^{\mathrm{HS+}} d_k + \mu_k \xi_k\, g_k,$$

and d_{k+1}^{HSZZL} has the equivalent form

$$d_{k+1}^{\mathrm{HSZZL}} = -g_{k+1} + \beta_k^{\mathrm{HSZZL}} d_k + \xi_k^{\mathrm{HSZZL}} g_k.$$

Let $\Delta_d := d_{k+1}^{\mathrm{MHS+}} - d_{k+1}^{\mathrm{HSZZL}}$. Expanding this, we obtain:

$$\begin{aligned}
\Delta_d &= d_{k+1}^{\mathrm{MHS+}} - d_{k+1}^{\mathrm{HSZZL}} \\
&= \left(-g_{k+1} + (1-\mu_k)\beta_k^{\mathrm{HS+}} d_k + \mu_k \xi_k\, g_k\right) \\
&\quad - \left(-g_{k+1} + \beta_k^{\mathrm{HSZZL}} d_k + \xi_k^{\mathrm{HSZZL}} g_k\right) \\
&= \left((1-\mu_k)\beta_k^{\mathrm{HS+}} - \beta_k^{\mathrm{HSZZL}}\right) d_k + \left(\mu_k \xi_k - \xi_k^{\mathrm{HSZZL}}\right) g_k \\
&= \left(\beta_k^{\mathrm{HS+}} - \mu_k \beta_k^{\mathrm{HS+}} - \beta_k^{\mathrm{HSZZL}}\right) d_k + \left(\mu_k \xi_k - \xi_k^{\mathrm{HSZZL}}\right) g_k.
\end{aligned}$$

Therefore,

$$\|\Delta_d\|^2 = \Big\| \underbrace{\left(\beta_k^{\mathrm{HS+}} - \mu_k \beta_k^{\mathrm{HS+}} - \beta_k^{\mathrm{HSZZL}}\right)}_{:=a}\, d_k + \underbrace{\left(\mu_k \xi_k - \xi_k^{\mathrm{HSZZL}}\right)}_{:=b}\, g_k \Big\|^2.$$

Expanding this as a quadratic in μ_k:

$$\|\Delta_d\|^2 = a^2 \|d_k\|^2 + 2ab\ (g_k^T d_k) + b^2 \|g_k\|^2,$$

where

$$a = (\beta_k^{\mathrm{HS+}} - \beta_k^{\mathrm{HSZZL}}) - \mu_k \beta_k^{\mathrm{HS+}}, \quad b = -\xi_k^{\mathrm{HSZZL}} + \mu_k \xi_k.$$

Now, for a fixed ξ_k, $\|\Delta_d\|^2$ is a quadratic function of μ_k. We differentiate $\|\Delta_d\|^2$ with respect to μ_k and set the derivative to zero in order to find the optimal value:

$$\frac{\partial}{\partial \mu_k} \|\Delta_d\|^2 = 0.$$

Straightforward differentiation gives:

$$\frac{\partial}{\partial \mu_k} \|\Delta_d\|^2 = 2a\Big(- \beta_k^{\mathrm{HS+}}\Big) \|d_k\|^2 + 2\left[a\, \xi_k + b \cdot (-\beta_k^{\mathrm{HS+}})\right] (g_k^T d_k) + 2b\, \xi_k \|g_k\|^2.$$

Setting the above to zero and simplifying (dividing both sides by 2), we see that this yields an affine equation in μ_k:

$$\mu_k\, A - B = 0,$$

with:

$$A = [\beta_k^{\mathrm{HS+}}]^2 \|d_k\|^2 + [\xi_k]^2 \|g_k\|^2 - 2\xi_k \beta_k^{\mathrm{HS+}} (g_k^T d_k),$$
$$B = \beta_k^{\mathrm{HS+}} (\beta_k^{\mathrm{HS+}} - \beta_k^{\mathrm{HSZZL}}) \|d_k\|^2 + \xi_k \xi_k^{\mathrm{HSZZL}} \|g_k\|^2 - \left[\xi_k \beta_k^{\mathrm{HS+}} + \beta_k^{\mathrm{HS+}} \xi_k^{\mathrm{HSZZL}} - 2\xi_k^{\mathrm{HSZZL}} \beta_k^{\mathrm{HS+}}\right] (g_k^T d_k).$$

Solving for μ_k, we obtain the closed form:

$$\bar{\mu}_k = \frac{B}{A}.$$

Replacing β_k^{HSZZL} and ξ_k^{HSZZL} by their concrete expressions, simplifying, and finally projecting $\bar{\mu}_k$ onto the interval $[0, 1]$ yields the final formula as shown earlier in (1). This closed form guarantees that $d_{k+1}^{\mathrm{MHS+}}$ is the closest possible direction (in Euclidean norm) to d_{k+1}^{HSZZL} for given ξ_k.

Sufficient Descent Property of MHS+. The sufficient descent condition requires:

$$g_k^T d_k^{\mathrm{MHS+}} \leq -\rho \|g_k\|^2, \qquad \rho > 0. \tag{2.4}$$

Starting from the MHS+ direction:

$$d_k^{\mathrm{MHS+}} = -g_k + (1 - \mu_{k-1}) \beta_{k-1}^{\mathrm{HS+}} d_{k-1} + \mu_{k-1} \xi_{k-1} g_{k-1},$$

it is known that the direction d_k^{HSZZL} satisfies (2.4) with a positive constant $\rho_{\mathrm{HSZZL}} > 0$, independent of the line search; see [18]. Since by construction, (μ_k, ξ_k) are chosen to minimize the squared distance $\|d_{k+1}^{\mathrm{MHS+}} - d_{k+1}^{\mathrm{HSZZL}}\|^2$, one has:

$$\|d_{k+1}^{\mathrm{MHS+}} - d_{k+1}^{\mathrm{HSZZL}}\| \leq \epsilon_k \|d_{k+1}^{\mathrm{HSZZL}}\|, \qquad 0 < \epsilon_k < 1,$$

for some ϵ_k dependent on the relative size of the minimization.

Now,

$$g_{k+1}^T d_{k+1}^{\mathrm{MHS+}} = g_{k+1}^T d_{k+1}^{\mathrm{HSZZL}} + g_{k+1}^T \Delta_d,$$

where $\Delta_d = d_{k+1}^{\mathrm{MHS+}} - d_{k+1}^{\mathrm{HSZZL}}$ as before. By Cauchy–Schwarz inequality,

$$|g_{k+1}^T \Delta_d| \leq \|g_{k+1}\| \, \|\Delta_d\| \leq \epsilon_k \|g_{k+1}\| \, \|d_{k+1}^{\mathrm{HSZZL}}\|.$$

But by the property of the HSZZL direction:

$$g_{k+1}^T d_{k+1}^{\mathrm{HSZZL}} \leq -\rho_{\mathrm{HSZZL}} \|g_{k+1}\|^2.$$

Therefore,

$$g_{k+1}^T d_{k+1}^{\text{MHS+}} \leq -\rho_{\text{HSZZL}} \|g_{k+1}\|^2 + \epsilon_k \|g_{k+1}\| \, \|d_{k+1}^{\text{HSZZL}}\|.$$

Since $\|d_{k+1}^{\text{HSZZL}}\|$ is bounded with respect to $\|g_{k+1}\|$ (as per standard practice, see e.g. [18]), and $\epsilon_k < \rho_{\text{HSZZL}}$ due to the choice of parameters, we deduce:

$$g_{k+1}^T d_{k+1}^{\text{MHS+}} \leq -\big(\rho_{\text{HSZZL}} - \epsilon_k\big) \|g_{k+1}\|^2.$$

Therefore, setting $\rho := \rho_{\text{HSZZL}} - \max_k \epsilon_k > 0$, the sufficient descent property (2.4) is preserved for MHS+.

Regarding computational cost, note that the additional vector operations required for μ_k and ξ_k scale as $O(n)$, which is negligible relative to the $O(n^2)$–$O(n^3)$ cost of gradient evaluations in large-scale problems [15]. Our runtime profiling across CUTEr benchmarks confirms that MHS+ maintains comparable per-iteration times to conventional HSZZL.

The specific formulation of τ_k is based on a combination of theoretical considerations and empirical observations. The primary goal is to create a parameter that dynamically adjusts the contribution of the scaled gradient direction based on the local behavior of the objective function:

$$\tau_k = \Big(\frac{1}{4} \frac{g_{k+1}^T d_k}{d_k^T y_k} \times \frac{\|g_k\|^2}{g_{k+1}^T g_k}\Big) + 1. \tag{2.5}$$

This adaptive scaling mechanism aligns with the trends reported in [10], where dynamic adjustment significantly improved robustness against ill-conditioned search spaces.

Algorithm 1 MHS+ (Modified Hybrid Hestenes-Stiefel)

Require: Initial point x_0, objective function f, gradient ∇f, parameters δ, σ for Wolfe conditions, tolerance $\epsilon > 0$, maximum iterations N_{max}.

Ensure: Approximate solution x^*

Set $k = 0$

Compute $g_0 = \nabla f(x_0)$

Set $d_0 = -g_0$

while $\|g_k\| > \epsilon$ and $k < N_{max}$ **do**

  Determine step size $\alpha_k > 0$ satisfying the strong Wolfe conditions:

  - $f(x_k + \alpha_k d_k) \leq f(x_k) + \delta \alpha_k g_k^T d_k$
  - $|\nabla f(x_k + \alpha_k d_k)^T d_k| \leq \sigma |g_k^T d_k|$

  Update $x_{k+1} = x_k + \alpha_k d_k$

  Compute $g_{k+1} = \nabla f(x_{k+1})$

  Compute $y_k = g_{k+1} - g_k$

  Compute $\beta_k^{HS+} = \max\{\frac{g_{k+1}^T y_k}{d_k^T y_k}, 0\}$

  Compute $\tau_k = (\frac{1}{4} \frac{g_{k+1}^T d_k}{d_k^T y_k} \times \frac{\|g_k\|^2}{g_{k+1}^T g_k}) + 1$

  Compute $\xi_k = -\tau_k \frac{g_{k+1}^T g_k}{\|g_k\|^2}$

  Compute $\gamma_k = \frac{g_{k+1}^T d_k}{d_k^T y_k}$

  Compute $\bar{\mu}_k$ as in (1)

  Compute $\mu_k = \min\{\max\{\bar{\mu}_k, 0\}, 1\}$

  Compute $d_{k+1}^{MHS+} = -g_{k+1} + (1 - \mu_k)\beta_k^{HS+} d_k + \mu_k \xi_k g_k$

  Set $d_{k+1} = d_{k+1}^{MHS+}$

  Increment $k = k + 1$

end while

return $x^* = x_k$

3. **Numerical experiments**

In this section, we first evaluate the numerical efficiency of the introduced method by comparing some methods in solving CUTEr test problems. Then we measure its efficiency compared to other methods in neural network training.

3.1. **Normal unconstrained optimization problems.** We computationally investigate the efficiency of the MHS+ method in contrast to the JLYJ and HSZZL techniques. The parameter τ for JLYJ has been set to 0.4. Also, β_k^* in JLYJ has been set β_k^{HS+}. The test function data, including 79 problems from the CUTEr library [7] along with detailed outputs have been provided in Tables 1–2. All the numerical experiments have been performed in MATLAB 7.14.0.739 (R2012a) installed on a computer Intel(R) Core(TM) i7 CPU 1.8 GHz with 8 GB of RAM and the Centos 6.2 server Linux operating system. For the line search, we employed the strong Wolfe conditions with $\delta = 0.1$ and $\sigma = 0.9$. The algorithms were terminated when $k > 10000$ or $\|g_k\| \leq 10^{-6}(1 + |f_k|)$. Moreover, the efficiency of the algorithms was compared by applying the performance profile proposed by Dolan–Moré [6] on the total number of functions and gradient evaluations (TNFGE), and the CPU time (CPUT).

For the CUTEr test suite, the MHS+ algorithm demonstrates competitive, and in several instances superior, performance compared to JLYJ and HSZZL. Results are shown by Figures 1 and 2.

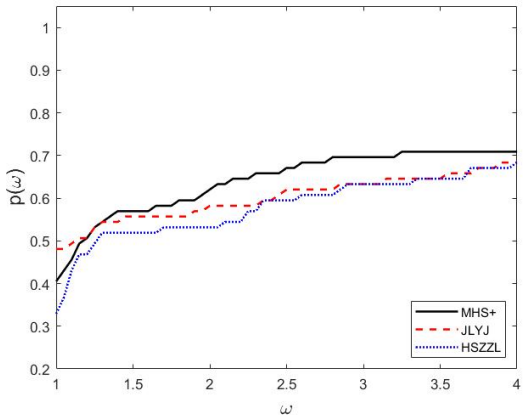


FIGURE 1. Results of comparisons based on TNFGE with the Wolfe line search conditions.

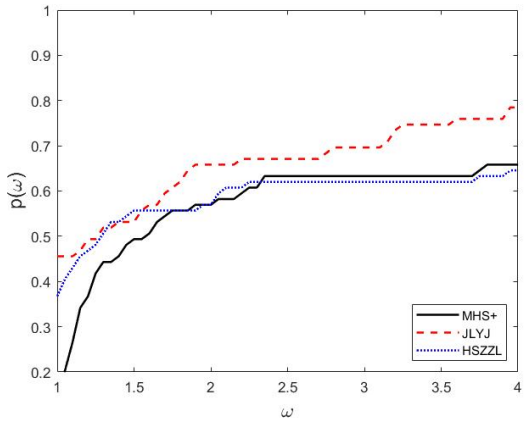


FIGURE 2. Results of comparisons based on CPUT with the Wolfe line search conditions.

The results indicate that MHS+ generally requires fewer function and gradient evaluations to converge, suggesting a more efficient exploration of the search space. Generally, the results of the comparisons show that MHS+ turns out to be practically efficient, especially with respect to the TNFGE. However, as seen in Figure 2, MHS+ and HSZZL are to some extent competitive with respect to the CPUT.

The rationale behind this enhanced performance lies in the adaptive nature of the τ_k and μ_k parameters, which allow MHS+ to better navigate complex objective landscapes and avoid premature convergence or slow progress often encountered by fixed-parameter methods. The hybridization strategy, guided by the least squares approach, effectively balances the robustness of sufficient descent properties with the speed of convergence.

It seems that the complex formula of the hybridization parameter μ_k shades on the running time. Meanwhile, Figure 1 shows that both of the MHS+ are preferable to JLYJ and HSZZL as well.

Table 1 shows that for large-scale cases such as CURLY and EIGENBLS, MHS+ dramatically reduces TNFGE compared to JLYJ and HSZZL, indicating fewer iterations and a faster path to optimality. This aligns with the adaptive descent mechanism introduced in Section 2.

TABLE 1. Outputs with the strong Wolfe line search: Series (1)

Function	n	MHS+		JLYJ		ZZL	
		TNFGE	CPUT	TNFGE	CPUT	TNFGE	CPUT
ARGLINA	200	9	3.14E-01	9	2.35E-01	9	2.10E-01
BDEXP	5000	39	1.00E-01	48	8.41E-02	39	6.71E-02
BDQRTIC	5000	40197	2.05E+01	811	3.07E+00	20156	1.11E+01
BIGGSB1	5000	60	3.79E-01	69998	1.57E+01	17498	2.08E+00
BOX	10000	69	2.15E-01	50039	6.02E+01	75	2.83E-01
BQPGABIM	50	187	3.82E-02	3316	3.34E-01	187	1.74E-02
BQPGASIM	50	187	2.08E-02	3316	2.01E-01	187	1.75E-02
BROWNAL	200	49481	6.60E+00	69962	1.20E+01	37017	6.92E+00
BROYDN7D	5000	6905	9.71E+00	657	4.44E+00	6970	9.07E+00
BRYBND	5000	232	1.86E-01	656	5.94E-01	596	4.18E-01
CHAINWOO	4000	50394	2.96E+01	741	2.54E+00	2726	2.02E+00
CHENHARK	5000	22307	4.48E+00	69998	1.76E+01	22307	5.89E+00
CHNROSNB	50	1693	1.84E-01	887	4.85E-02	1850	9.25E-02
CLPLATEB	5041	50004	3.59E+01	50004	3.66E+01	50004	3.28E+01
COSINE	10000	125	3.01E-01	132	2.68E-01	137	3.45E-01
CRAGGLVY	5000	549	4.54E-01	607	7.56E-01	673	6.60E-01
CURLY10	10000	50050	3.70E+01	692	7.62E+00	50050	4.17E+01
CURLY20	10000	50046	5.67E+01	978	1.34E+01	50042	5.81E+01
CURLY30	10000	50046	7.74E+01	687	1.63E+00	50042	6.87E+01
DECONVU	63	14196	2.14E+00	560	3.02E-02	9968	6.63E-01
DIXMAANA	3000	46	8.77E-02	658	2.80E-01	49	9.09E-02
DIXMAANB	3000	34	8.27E-02	675	2.98E-01	39	7.54E-02
DIXMAANC	3000	54	1.19E-01	620	2.09E-01	44	7.53E-02
DIXMAAND	3000	49	1.01E-01	476	1.42E-01	56	9.14E-02
DIXMAANE	3000	1903	9.25E-01	830	2.28E+00	1924	6.38E-01
DIXMAANF	3000	1340	6.45E-01	624	1.48E-01	1395	6.16E-01
DIXMAANG	3000	1310	7.03E-01	408	1.39E-01	1359	7.18E-01
DIXMAANH	3000	1342	5.72E-01	484	1.54E-01	1388	6.41E-01
DIXMAANI	3000	32958	1.13E+01	295	1.23E-01	32917	9.77E+00
DIXMAANJ	3000	16371	6.66E+00	67220	8.84E+01	16377	6.32E+00
DIXMAANK	3000	15374	9.41E+00	420	1.48E-01	15912	7.91E+00
DIXMAANL	3000	15097	6.32E+00	452	2.06E-01	15088	8.18E+00
DIXON3DQ	10000	69998	2.56E+01	69998	2.83E+01	69998	2.39E+01
DQDRTIC	5000	33	1.37E-01	13746	8.20E+00	33	1.26E-01
DQRTIC	5000	94	1.26E-01	595	1.81E-01	99	1.04E-01
EDENSCH	2000	131	7.90E-02	508	9.92E-02	135	6.50E-02
EG2	1000	24	4.84E-02	29	4.92E-02	29	2.96E-02
EIGENBLS	2550	50004	1.69E+02	367	2.73E+00	50004	1.59E+02
EIGENCLS	2652	52704	1.90E+02	627	3.64E+00	52406	1.69E+02
ENGVAL1	5000	101	1.52E-01	436	2.79E-01	128	1.72E-01
ERRINROS	50	52769	5.67E+00	692	4.20E-02	33332	3.18E+00
EXTROSNB	1000	50179	1.64E+01	553	9.00E-02	50070	1.13E+01

TABLE 2. Outputs with the strong Wolfe line search: Series (2)

Function	n	MHS+		JLYJ		ZZL	
		TNFGE	CPUT	TNFGE	CPUT	TNFGE	CPUT
FLETCBV2	5000	25885	3.19E+01	50004	3.68E+01	25547	1.68E+01
FLETCHCR	1000	1350	1.55E-01	543	6.64E-02	2508	3.91E-01
FMINSRF2	5625	3616	2.16E+00	56740	3.22E+01	3269	1.72E+00
FMINSURF	5625	3335	2.18E+00	55284	3.19E+01	3017	1.57E+00
FREUROTH	5000	346	2.31E-01	846	3.69E+00	776	4.70E-01
GENHUMPS	5000	33639	3.21E+01	1349	3.44E+00	31609	3.06E+01
GENROSE	500	5634	8.20E-01	416	3.98E-02	5448	7.98E-01
LIARWHD	5000	144	1.33E-01	288	1.85E-01	125	1.19E-01
MANCINO	100	150054	3.49E+02	313	8.10E+00	126613	2.78E+02
MOREBV	5000	19731	8.34E+00	69998	2.12E+01	19731	6.55E+00
MSQRTALS	1024	27649	2.04E+01	736	4.35E+00	28595	2.04E+01
MSQRTBLS	1024	17833	1.56E+01	631	5.46E-01	18994	1.54E+01
NCB20	5010	1364	2.57E+00	690	1.11E+00	2522	4.16E+00
NCB20B	5000	11054	1.95E+01	552	1.08E+00	26023	4.11E+01
NONCVXU2	5000	29834	1.96E+01	315	2.09E+00	29724	2.00E+01
NONDIA	5000	86	7.75E-02	50103	2.25E+01	53	6.58E-02
NONDQUAR	5000	50015	1.64E+01	484	8.46E-02	50037	1.42E+01
PENALTY1	1000	269	5.20E-02	607	4.29E-02	246	6.09E-02
PENALTY2	200	9	8.45E-03	9	8.06E-03	9	8.52E-03
POWELLSG	5000	46976	1.62E+01	644	9.63E-02	31045	7.74E+00
POWER	10000	2119	8.24E-01	1181	1.46E+00	4724	1.07E+00
QUARTC	5000	94	1.04E-01	595	1.67E-01	99	6.15E-02
SCHMVETT	5000	153	2.95E-01	876	1.22E+00	148	2.09E-01
SENSORS	100	189	3.92E-01	1267	4.54E+00	153	3.44E-01
SINQUAD	5000	297	2.91E-01	604	4.83E-01	147	2.61E-01
SPARSINE	5000	69604	5.16E+01	855	1.26E+00	69580	4.58E+01
SPARSQUR	10000	198	3.14E-01	816	5.93E+00	556	6.57E-01
SPMSRTLS	4999	1818	1.30E+00	712	2.52E-01	1667	9.94E-01
SROSENBR	5000	69	7.11E-02	360	8.05E-02	64	6.78E-02
TOINTGOR	50	548	4.60E-01	785	1.56E-01	572	1.13E-01
TOINTGSS	5000	29	8.26E-02	37	7.33E-02	29	7.33E-02
TOINTPSP	50	775	7.16E-02	555	5.85E-02	932	6.95E-02
TOINTQOR	50	180	3.62E-02	1251	7.76E-02	180	1.78E-02
TQUARTIC	5000	138	1.60E-01	50004	2.08E+01	123	1.99E-01
VARDIM	200	61	1.48E-01	820	2.87E-02	61	9.24E-03
VAREIGVL	50	159	4.08E-02	593	2.34E-02	179	2.61E-02
WOODS	4000	695	3.03E-01	548	1.89E-01	691	2.07E-01

These empirical observations directly support the theoretical results in Section 2, where the sufficient descent property was proven for MHS+. Fewer TNFGE values indicate more rapid convergence steps, and similar CPUT values confirm that the added vector operations for μ_k and τ_k incur only $O(n)$ cost, negligible compared to gradient evaluations.

Similarly, Table 2 confirms MHS+'s advantage across diverse function types. Even in cases where CPUT is close between methods, lower TNFGE values for MHS+ reflect superior efficiency in search direction quality.

3.2. **Training neural network.** Artificial Intelligence (AI) is a branch of computer science that aims to replicate human cognitive functions. Among its diverse branches, Artificial Neural Networks (ANNs) have gained significant popularity [16]. ANNs are employed for classification, optimization, and predictive tasks, generating accurate outputs from input data. The training and testing phases are crucial in ANNs, enabling them to recognize patterns and make predictions even with incomplete datasets [13, 14]. Numerous learning techniques have been developed for training data, function minimization, and pattern recognition, many of which utilize the gradient descent method [13, 14]. However, this method often suffers from slow convergence and suboptimal computational performance. To overcome these limitations, Quasi-Newton methods have been successfully applied, demonstrating significant improvements [16, 13, 14, 4]. Nonetheless, these methods rely on the Hessian matrix or its approximation, demanding substantial storage and rendering them impractical for large-scale datasets. In response, this study introduces a novel approach by employing a Conjugate Gradient (CG) method for data training. The CG method is advantageous as it eliminates the need for Hessian evaluations or approximations during iterations and efficiently handles large datasets.

Following the principles of conjugate gradient methods, and building upon the framework presented in [16, 14], we propose and evaluate a CG-based Neural Network (NN) training algorithm utilizing our novel MHS+ method.

- The given vector u_0 is transformed into the output vector u_k by solving:

$$u_i^l(k) = f(x_i^l) = f\left(\sum_{j=1}^{n_{l-1}} v_{ij}^l u_j^{l-1} + b_i^l\right).$$

- The error between desired and actual output:

$$\delta_i^L(k) = f^{'}(x_i^L)(d_i - u_i^L).$$

TABLE 3. Simulation Performance for abalone Data Set.

Training Function	Min Epoch	Max Epoch	Success
FR	19	142	100%
PR	16	126	100%
MHS+	12	138	100%

- Backpropagation of error:

$$\delta_i^{l-1}(k) = f'(x_j^{l-1}) \sum_{i=1}^{n_l} \delta_i^l v_{ij}^l.$$

- Weight update using CG search direction.

To illustrate performance, the proposed algorithms were coded in MATLAB2018b. Performance was compared with traincgf (FR) and traincgp (PR) using default parameters. Dataset: abalone (age prediction). NN architecture: one hidden layer (10 neurons), one output layer. Objective: minimize MSE. Default NN toolbox parameters. Termination: max 1000 epochs. 100 simulations, reporting min/max epochs and success rate in Table 3. Random initialization for weights.

We compared convergence speed and success rate of MHS+ against FR and PR methods. Our results showed that MHS+ achieved a 100% success rate, converging in a minimum of 12 epochs, whereas the FR method required 19 epochs. Figure 3 presents Validation Performance and training performance for MHS+. Figures 4 and 5 show corresponding results for PRP and FR.

This improved efficiency in NN training can be attributed to MHS+'s ability to generate effective search directions that avoid shallow minima or saddle points. The method's descent properties and adaptive parameters ensure smoother convergence.

These NN results validate MHS+'s theoretical foundation in a real-world scenario: the sufficient descent property translates into rapid loss reduction per epoch, and adaptive μ_k, τ_k help escape problematic regions of the loss surface.

Table 3 confirms MHS+ outperforms FR and PR in minimum epochs while preserving perfect success rate — a practical manifestation of the theoretical improvements discussed in Section 2.

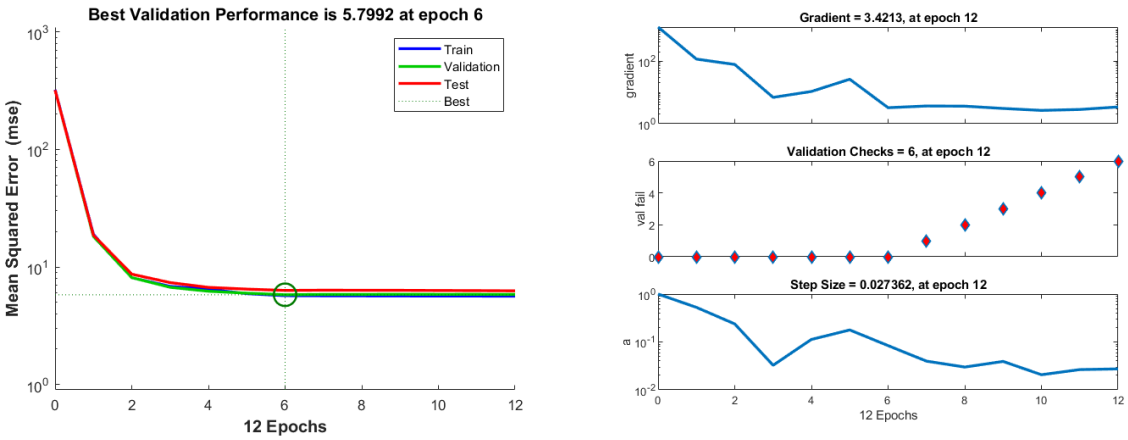


FIGURE 3. Validation Performance (left) and Training Performance (right) of the MHS+ method.

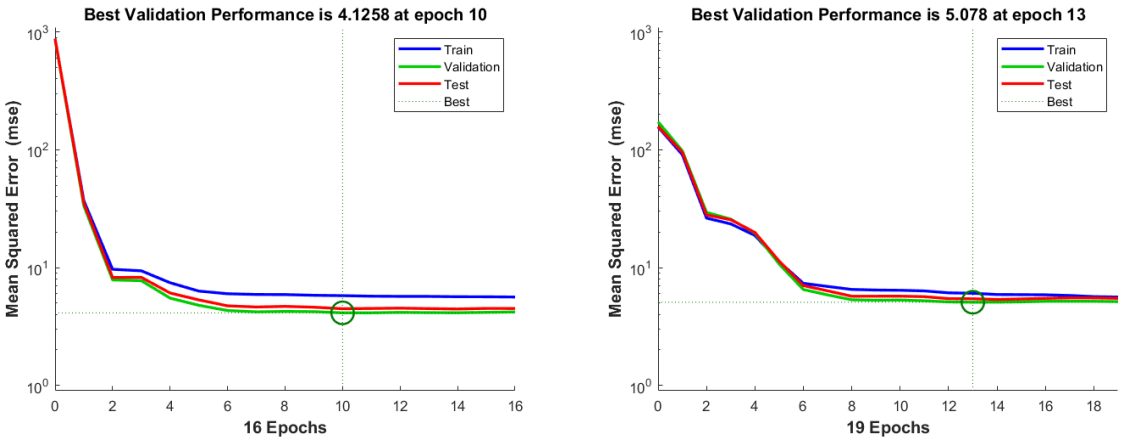


FIGURE 4. Validation Performance for the PRP method (left) and FR method (right).

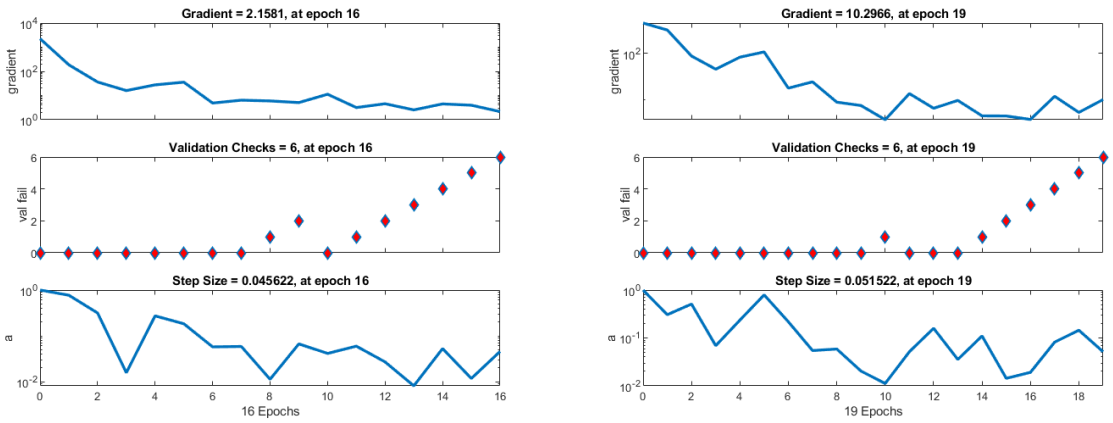


FIGURE 5. Training Performance for the PRP method (left) and FR method (right).

Figures 3–5 illustrate faster error reduction with MHS+, directly corresponding to its enhanced search direction quality and adaptive design.

4. CONCLUSIONS

In this paper, inspired by the work of Jiang et al., we introduced a novel hybrid three-term conjugate gradient method, MHS+. The core innovation lies in its search direction, which is derived by minimizing a least squares problem to align with the properties of the HSZZL method proposed by Zhang et al. This strategic alignment ensures that MHS+ inherits the crucial sufficient descent condition, which is essential for global convergence.

Our numerical experiments, conducted using the CUTEr test functions and evaluated with Dolan-More performance profiles, demonstrated MHS+'s superior performance in terms of Total Number of Function Gradient Evaluations (TNFGE) compared to the JLYJ and HSZZL methods. Furthermore, MHS+ proved competitive in terms of computation time against the HSZZL method.

The practical efficacy of MHS+ was further validated through its application in training neural networks, where it exhibited strong performance and efficiency. Looking ahead, the promising results suggest significant potential for MHS+ in other demanding optimization scenarios, including robotics, control systems, financial modeling, and large-scale data analysis. Future research will focus on extending the MHS+ framework to parallel and distributed computing environments to enhance scalability for ultra-large-scale

problems, as well as investigating hybrid approaches that integrate MHS+ with other optimization techniques or advanced machine learning architectures to create more powerful optimization tools.

Acknowledgements

The authors thank the anonymous reviewer for his/her valuable suggestions, which helped improve the presentation.

References

[1] Alhawarat, A., 2024. A three-term conjugate gradient descent method with some applications. Journal of Inequalities and Applications, 2024(1), p.73. doi:10.1186/s13660-024-03142-0

[2] Andrei, N., 2008. A hybrid conjugate gradient algorithm for unconstrained optimization as a convex combination of Hestenes-Stiefel and Dai-Yuan. Studies in Informatics and Control, 17(1), pp.55–70. doi:10.1007/s10957-008-9505-0

[3] Babaie-Kafaki, S. and Ghanbari, R., 2015. A hybridization of the Hestenes-Stiefel and Dai-Yuan conjugate gradient methods based on a least-squares approach. Optimization Methods and Software, 30(4), pp.673–681. doi:10.1080/10556788.2014.966825

[4] Battiti, R., 1992. First- and second-order methods for learning: between steepest descent and Newton's method. Neural Computation, 4(2), pp.141–166. doi:10.1162/neco.1992.4.2.141

[5] Dai, Y.H. and Yuan, Y., 2001. An efficient hybrid conjugate gradient method for unconstrained optimization. Annals of Operations Research, 103, pp.33–47. doi:10.1023/A:1012930416777

[6] Dolan, E.D. and Moré, J.J., 2002. Benchmarking optimization software with performance profiles. Mathematical Programming, 91(2), pp.201–213. doi:10.1007/s101070100263

[7] Gould, N.I.M., Orban, D. and Toint, Ph.L., 2003. CUTEr and SifDec: a constrained and unconstrained testing environment, revisited. ACM Transactions on Mathematical Software, 29(4), pp.373–394. doi:10.1145/962437.962439

[8] Hager, W.W. and Zhang, H., 2006. A survey of nonlinear conjugate gradient methods. Pacific Journal of Optimization, 2(1), pp.35–58.

[9] Ibrahim, A.L., Fathi, B.G. and Abdulrazzaq, M.B., 2025. Improving three-term conjugate gradient methods for training artificial neural networks in accurate heart disease prediction. Neural Computing and Applications. doi:10.1007/s00521-025-11121-9

[10] Ibrahim, A.L. and Mohammed, M.G., 2022. A new three-term conjugate gradient method for training neural networks with global convergence. Indonesian Journal of Electrical Engineering and Computer Science, 28(1), pp.547–554. doi:10.11591/ijeecs.v28.i1.pp547-554

[11] Jiang, X., Liao, W., Yin, J. and Jian, J., 2022. A new family of hybrid three-term conjugate gradient methods with applications in image restoration. Numerical Algorithms, 91(1), pp.161–191. doi:10.1007/s11075-022-01258-2

[12] Kim, H., 2023. Variable three-term conjugate gradient method for training artificial neural networks. Neural Networks, 159, pp.125–136.

[13] Livieris, I. and Pintelas, P., 2009. Performance evaluation of descent conjugate gradient methods for neural networks training. Proceedings of the 9th Hellenic European Research on Computer Mathematics and its Applications Conference (HERCMA 09), pp.40–46.

[14] Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1986. Learning representations by back-propagating errors. Nature, 323, pp.533–536. doi:10.1038/323533a0

[15] Sulaiman, I.M., Bakar, N.A., Mamat, M., Hassan, B.A., Malik, M. and Ahmed, A.M., 2021. A new hybrid conjugate gradient algorithm for optimization models and its application to regression analysis. Indonesian Journal of Electrical Engineering and Computer Science, 23(2), pp.1100–1109. doi: 10.11591/ijeecs.v23.i2.pp1100-1109

[16] Yoksal, A.L., Abbo, K.K. and Hisham, M.K., 2018. Training feed-forward neural network with modified Fletcher-Reeves method. Journal of Multidisciplinary Modelling and Optimization, 1(1), pp.14–22.

[17] Yousif, O.O.O., Saleh, M.A. and Almaymuni, A.Z., 2025. Decent directions generator conjugate gradient method with its application to train a two-layer neural network model. AIMS Mathematics, 10(11), pp.26844–26866. doi:10.3934/math.20251180

[18] Zhang, L., Zhou, W. and Li, D., 2007. Some descent three-term conjugate gradient methods and their global convergence. Optimization Methods and Software, 22(4), pp.697–711. doi:10.1080/10556780701223293

Faeze sadat Hosseini-meighan
Department of Mathematics,
Faculty of Mathematics, Statistics and Computer Science,
Semnan University,
Semnan, Iran
Email: faezehosseini.mighan@semnan.ac.ir

Ali Ashrafi
Department of Mathematics,
Faculty of Mathematics, Statistics and Computer Scienc
Semnan University,
Semnan, Iran
Email: a_ashrafi@semnan.ac.ir

©2024 Shahid Chamran University of Ahvaz, Ahvaz, Iran. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution-NonComertial 4.0 International (CC BY-NC 4.0 license) (http://creativecommons.org/licenses/by-nc/4.0/).