OPTIMIZATION ERUDITORUM
Volume 3 (2026), No. 1, 34–53
https://doi.org/10.69829/oper-026-0301-ta04
Tulipa Opera Scholarum

A NEW EFFICIENT ALTERNATIVE EXTENSION OF THE HAGER-ZHANG CONJUGATE GRADIENT METHOD FOR VECTOR OPTIMIZATION

XIAOQING OU[1], YUNYE WAN[2], ZHAO-HAN LIU[3], HUILIN HAN[3], WEIGUANG PENG[3], AND JIAWEI CHEN[3,*]

[1] *College of Management, Chongqing College of Humanities, Science & Technology, Chongqing 401524, China*
[2] *Beibei Power Supply Branch of State Grid Chongqing Electric Power Company, Chongqing 400700, China*
[3] *School of Mathematics and Statistics, Southwest University, Chongqing 400715, China*

Abstract. In this paper, a new efficient alternative extension of the Hager-Zhang conjugate gradient method is proposed to solve the vector optimization problem that ensures sufficient descent without relying on any linear search or convexity assumptions. We give the convergence of our proposed method combined with Wolfe line search under mild assumptions. Finally, we show through numerical experiments that our proposed method is more efficient than the extant related vector Hagar-Zhang conjugate method.

Keywords. Vector optimization, Hager-Zhang conjugate gradient method, Pareto critical point, Sufficiently descent, Convergence.

1. Introduction

In this paper, we focus on the following vector optimization problems:

$$\text{(P)} \qquad \min F(x) = (f_1(x), f_2(x), \ldots, f_q(x)) \tag{1.1}$$

where $f_j : \mathbb{R}^n \to \mathbb{R}$, $j = 1, \cdots, q$, are continuous differentiable functions. $K \subseteq \mathbb{R}^q$ is a closed, convex and pointed cone with a nonempty interior, i.e. $\text{int(K)} \neq \emptyset$. This partial order $\preceq_K$ ($\prec_K$) in $\mathbb{R}^q$ induced by K (int(K)) is defined by $a \preceq_K b (a \prec_K b)$ if and only if $a - b \in K (a - b \in \text{int(K)})$.

Vector optimization problems refer to maximizing or minimizing vector-valued functions under certain conditions. In vector optimization, two or more objective functions must be minimized simultaneously. However, in general, there is no point that minimizes all objective functions. Hence, the concept of optimality in vector optimization has to be replaced by the concept of Pareto-optimality or effciency. A point x_0 is called Pareto-optimal if there does not exist x such that $x \neq x_0$ and $F(x) \prec_K F(x_0)$. In practice, a very special and important case is when $K = \mathbb{R}^q_+$. This case is multi-objective optimization. The multi-objective optimization is widely used in various fields, such as engineering, finance, radiotherapy, environment analysis, science [1–13] and the references therein.

In recent years, the development of methods for solving vector optimization problems has attracted a lot of attention. The most common strategy is based on scalarization techniques [14, 15]. Geoffrion first proposed the scalarization method [16] to solve the vector optimization problems. These methods need to consider the order of importance and weighting coefficients of the components of the objective function vector, which increases the difficulty of parameter selection for solving the vector optimization problem and may lead to unbounded scalarization problems because of some selected parameters.

*Corresponding author.
E-mail address: ouxiaoqing413@163.com (X. Ou), w3108116616@163.com (Y. Wan), liuzhaohan6799@163.com (Z.H. Liu), hhllhh113@163.com (H. Han), pwgedu@swu.edu.cn (W. Peng), j.w.chen713@163.com (J. Chen)

2020 Mathematics Subject Classification: 65K05, 90C29.
Accepted: February 08, 2026.

Fliege and Svaiter [17] introduced the steepest descent method for solving unconstrained multiobjective optimization problem to overcome these shortcomings, where neither ordering information nor weighting factors are assumed to be known. Graña Drummond and Svaiter proposed the steepest descent method [18] for solving vector optimization problems based on [17]. After that, many single-objective optimization methods are extended to vector optimization, such as Newton, quasi-Newton, projected gradient, subgradient, interior point, and proximal methods, etc. Fliege, Drummond and Svaiter [19] gave the Newton methods for multiobjective optimization. Chuong [20] introduced the Newtom-like methods for efficicent solutions in vector optimization. Then Ansary [21] and Panda gave a modified quasi-Newton methods for vector optimization problem. Drummond and Iusem [22] introduced the projected gradient methods for solving the vector optimization problems. Cruz [23] gave a subgradient method for solving vector optimization problems, as an extension of the projected subgradient method to convex vector optimization. Bonnel and Iusem et al. [24] proposed the proximal point methods. Ceng and Yao [25] proposed the approximate proximal methods to solve vector optimization. Chen and Huang et al. [26] gave the generalized proximal point algorithms for finding a weak Pareto optimal solution in finite-dimensional spaces by the virtue of a Bregman distance function. Villacorta and Oliverira [27] introduced an interior proximal method for vector optimization.

Pérez and Prudente [28] extended the nonlinear conjugate gradient method for vector optimization without convex assumption. It is well known that the conjugate gradient method has the advantages of simple structure and small storage, and is widely used in unconstrained optimization. The conjugate gradient method generates a sequence of iterates by the following equation

$$x_{k+1} = x_k + \alpha_k d_k, \tag{1.2}$$

where $d_k \in \mathbb{R}^n$ is the search direction, and $\alpha_k > 0$ is the step size. The direction d_k is defined by

$$d_k = \begin{cases} v(x_k), & \text{if} \quad k = 0, \\ v(x_k) + \beta_k d_{k-1}, & \text{if} \quad k \geq 1, \end{cases} \tag{1.3}$$

where β_k is a conjugate parameter and $v(x_k)$ is the vector steepest descent direction can be obtained by solving another miniaturisation subproblem. Lucambio Pérez and Prudente [28] pointed the choose of conjugate parameter β_k is open. They also presented the vector extensions of the parameters of Fletcher-Reeves (FR), Conjugate descent (CD), Dai-Yuan (DY), Polak-Ribière-Polyak (PRP) and Hestenes-Stiefel (HS), which respectively defined by

$$\beta_k^{FR} = \frac{\varphi\left(x_k, v\left(x_k\right)\right)}{\varphi\left(x_{k-1}, v\left(x_{k-1}\right)\right)}, \tag{1.4}$$

$$\beta_k^{CD} = \frac{\varphi\left(x_k, v\left(x_k\right)\right)}{\varphi\left(x_{k-1}, d_{k-1}\right)}, \tag{1.5}$$

$$\beta_k^{DY} = \frac{-\varphi\left(x_k, v\left(x_k\right)\right)}{\varphi\left(x_k, d_{k-1}\right) - \varphi\left(x_{k-1}, d_{k-1}\right)}, \tag{1.6}$$

$$\beta_k^{PRP} = \frac{-\varphi\left(x_k, v\left(x_k\right)\right) + \varphi\left(x_{k-1}, v\left(x_k\right)\right)}{-\varphi\left(x_{k-1}, v\left(x_{k-1}\right)\right)}, \tag{1.7}$$

$$\beta_k^{HS} = \frac{-\varphi\left(x_k, v\left(x_k\right)\right) + \varphi\left(x_{k-1}, v\left(x_k\right)\right)}{\varphi\left(x_k, d_{k-1}\right) - \varphi\left(x_{k-1}, d_{k-1}\right)}, \tag{1.8}$$

where φ is defined in the section 2. For the ways of selecting parameter β_k, there are also some interesting works, such as the vector extension of Hager-Zhang (see [29, 30]), the vector extension of Liu-Storey (see [31]) and the variants of FR, CD and PRP (see [32]).

In [29], Gonçalves and Prudente first extended the HZ method to the vector optimization. This method overcomes the difficulty of directly expanding the HZ parameter in a vectorial sense without the descent property by utilizing a sufficiently accurate line search to find the direction of descent. But sufficiently accurate line searches are computationally expensive at each iteration. Then, Hu, Zhu and Chen [30] proposed an alternative extension of the Hager-Zhang nonlinear conjugate gradient method for vector optimization, which ensures that directions can be found that satisfy sufficient descent conditions without relying on any line search or convexity assumptions. In addition, a proof of global convergence under mild assumptions as well as numerical experiments in which the efficiency of the algorithm outperforms that of extant algorithms are given in the paper.

In this paper, we propose an efficient modified alternative extension of HZ nonlinear conjugate gradient method (MAVHZ) for vector optimization. We propose a new form of conjugate parameter β_k and β_k^{MAVHZ}. This new parameter not only ensures that we find directions that satisfy the sufficient descent condition without relying on any line searches, but also saves significant computational costs in the run. Besides, we obtain its global convergence of these methods with Wolfe-like line search strategy under certain wild assumptions. Numerical results show that the proposed method exceeds the existing methods in terms of efficiency. Meanwhile our proposed method has a very obvious advantage in high dimensional experiments. The main attributes are listed as follows

• In our proposed method, we give an efficient form of β_k. It not only avoids the problem of needing sufficiently accurate line search to ensure descent in HZ, but is also more efficient than the AVHZ method, which also does not require sufficiently accurate line search for descent property.

• The MAVHZ method has been shown through numerical experiments to have significant advantages over existing algorithms in high-dimensional experiments.

The rest of the paper is organized as follows. In Section 2, we some basic definitions, notations and auxiliary results in vector optimization. In Section 3, we present the MAVHZ method and study its descent property without any linear search procedure. Moreover, the global convergence of the method with Standard Wolfe linear search is given in 3. In Section 4, the numerical tests are reported to show the efficiency of the proposed algorithms by comparing with HZ and AVHZ methods.

2. Preliminaries

In this section, we present some definitions, notation and basic results used in this paper. Throughout this paper, we denote the usual inner product by $\langle\cdot,\cdot\rangle$ in $\mathbb{R}^n$ and $\langle q\rangle = \{1, 2, \cdots, q\}$.

Given a matrix $A \in \mathbb{R}^{m\times n}$ and two norms $\|\cdot\|_a$ and $\|\cdot\|_b$ on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively, the induced matrix norm $\|\cdot\|_{a,b}$ is defined by

$$\|A\|_{a,b} = \max_{x\in\mathbb{R}^n} \{\|Ax\|_b : \|x\|_a \leq 1\}.$$

When $a = b$, we use $\|\cdot\|_a$ instead of $\|\cdot\|_{a,a}$.

If $\|\cdot\|_a = \|\cdot\|_b = \|\cdot\|_2$, then the induced norm of a matrix $A \in \mathbb{R}^{m\times n}$ is given by $\|A\|_2 = \sqrt{\lambda_{\max}(A^TA)}$, where $\lambda_{\max}(A^TA)$ stands for the maximum eigenvalue of A^TA. Next, we recall some important definitions of dominate and Pareto optimal, which are given in [33, 34]. We also discuss the following problem (P), i.e.

$$\min_K F(x) = (f_1(x), f_2(x), \cdots, f_q(x))$$

where $f_j : \mathbb{R}^n \to \mathbb{R}$, $j = 1, \cdots, q$, are continuous differentiable functions. The previous section points out that K is a closed convex set, now let K^* be the positive polar cone of K defined as

$$K^* = \{u \in \mathbb{R}^q \mid \langle u, w\rangle \geq 0, \forall w \in K\}.$$

Since K is a closed convex set, we have $K = K^{**}$. Thus

$$-K = \{w \in \mathbb{R}^q \mid \langle w, u\rangle \leq 0, \forall u \in K^*\},$$

and

$$-\operatorname{int}(K)=\{w\in\mathbb{R}^q \mid \langle w,u\rangle<0,\forall u\in K^*\backslash\{0\}\}.$$

Let $X\subset K^*\backslash\{0\}$ be compact such that

$$K^*=\operatorname{cone}(\operatorname{conv}(X)),$$

i.e., K^* is the conic hull of the convex hull of X. Let $\mathbb{R}^n_+$ and $\mathbb{R}^n_{++}$ denote the non-negative orthant and positive orthant of $\mathbb{R}^n$. In classical optimization $K=\mathbb{R}_+$, $K^*=\mathbb{R}_+$, we can take $X=\{1\}$. For multiobjective optimization $K=\mathbb{R}^q_+$, $K^*=K$, we can take X as a canonical basis of $\mathbb{R}^q$. In this paper, as in [18], we assume that

$$X=\{u\in K^* \mid \|u\|_1=1\}. \tag{2.1}$$

Definition 2.1. For any points $x,y\in\mathbb{R}^n$, we define that x K-dominate y if $F(x)\preceq_K F(y)$.

Definition 2.2. A point $x^*\in\mathbb{R}^n$ is said to be K-Pareto optimal of problem (P) if there exists no $x\in\mathbb{R}^n$ such that $F(x)\preceq_K F(x^*)$ and $F(x)\neq F(x^*)$.

A necessary condition for the K-Pareto-optimality of x^* is

$$-\operatorname{int}(K)\cap JF(x^*)(\mathbb{R}^n)=\emptyset, \tag{2.2}$$

where $JF(x^*)(\mathbb{R}^n)=\{JF(x^*)d : d\in\mathbb{R}^n\}$ and

$$JF(x^*)d=\left(\langle\bigtriangledown f_1(x^*,d)\rangle,\langle\nabla f_2(x^*,d)\rangle,\cdots,\langle\nabla f_q(x^*,d)\rangle\right)^T.$$

For multi-objective, the first-order necessary condition proposed in [17] for the Pareto optimality of a point $x^*\in\mathbb{R}^n$ is

$$JF(x^*)(\mathbb{R}^n)\cap\left(-\mathbb{R}^q_{++}\right)=\emptyset.$$

We say point $x^*\in\mathbb{R}^n$ is K-Pareto critical if x^* satisfies first-order necessary condition (2.2). Therefore, if $x\in\mathbb{R}^n$ is not a K-Pareto critical point, then there exists a vector $d\in\mathbb{R}^n$ satisfying $JF(x)(\mathbb{R}^n)\cap-\operatorname{int}(K)\neq\emptyset$, which implies $JF(x)d\in-\operatorname{int}(K)$.

Here, we say that v is a K-descent direction for F at x if there exists $\epsilon>0$ such that

$$F(x+\alpha v)\prec_K F(x), \text{ for each } \alpha\in(0,\varepsilon].$$

Define $\varphi:\mathbb{R}^n\times\mathbb{R}^n\to\mathbb{R}$ by

$$\varphi(x,d):=\sup\{\langle JF(x)d,u\rangle \mid u\in X\}. \tag{2.3}$$

The function φ gives a characterization of K-descent directions of F at x and of critical points:

(i) d is a K-descent direction for F at x if and only if $\varphi(x,d)<0$;
(ii) x is critical if and only if $\varphi(x,d)\geq 0$ for all d.

According to the definition of φ, some useful properties of φ are recalled as follows.

Lemma 2.1. *[18, 35] For all $x,y\in\mathbb{R}^n$, $\varpi>0$ and $d_1,d_2\in\mathbb{R}^n$, then we have*
(i) $\varphi(x,\varpi d_1)=\varpi\varphi(x,d_1)$;
(ii) $\varphi(x,d_1+d_2)\leq\varphi(x,d_1)+\varphi(x,d_2)$ and $\varphi(x,d_1)-\varphi(x,d_2)\leq\varphi(x,d_1-d_2)$;
(iii) $|\varphi(x,d_1)-\varphi(y,d_2)|\leq\|JF(x)d_1-JF(y)d_2\|$;
(iv) If $F:\mathbb{R}^n\to\mathbb{R}^q$ is a continuously differentiable function and JF is Lipschitz continuous with constant L, then $\varphi(\cdot,d)$ is Lipschitz continuous with constant $L\|d\|$.

According to Lemma 2.1 (i) and (ii), we can easily get the following properties: For any $x, d_1, d_2 \in \mathbb{R}^n$, and $\zeta > 0$, we have

$$\varphi(x, d_1 + \zeta d_2) \leq \varphi(x, d_1) + \zeta\varphi(x, d_2). \tag{2.4}$$

Next, we extend the notion of steepest descent direction to the vector-valued case, so define $v : \mathbb{R}^n \to \mathbb{R}^n$ by

$$v(x) = \arg\min \left\{ \varphi(x, d) + \frac{\|d\|^2}{2} \,\middle|\, d \in \mathbb{R}^n \right\} \tag{2.5}$$

and the optimal value of the subproblem in (2.5) is given by

$$\theta(x) = \varphi(x, v(x)) + \frac{\|v(x)\|^2}{2}.$$

It's easy to see that $v(x)$ exists and is unique because $\varphi(x, \cdot)$ is a closed convex function. Note that, for the single-objection optimization, where $F : \mathbb{R}^n \to \mathbb{R}$, $K = \mathbb{R}_+$ and $X = \{1\}$, we can get $\varphi(x, d) = \langle \nabla F(x), d \rangle$, $v(x) = -\nabla F(x)$ and $\theta(x) = -\|\nabla F(x)\|^2/2$. For the multiobjective optimization, where $K = \mathbb{R}^q_+$ and X is a canonical basis of $\mathbb{R}^q$, $v(x)$ can be computed by solving the following problem

$$\begin{array}{l} \text{Minimize } \alpha + \frac{1}{2}\|v\|^2 \\ \text{s.t. } [JF(x)v]_i \leq \alpha, i = 1, 2, \ldots, m, \end{array} \tag{2.6}$$

which is a convex quadratic optimization problem with linear inequality constraints, see [17]. Let us now give the description of the K-critical for problem (1.1).

Lemma 2.2. *[18, Lemma 3.3]*
(i) If x is K-critical, then $v(x) = 0$, $\theta(x) = 0$;
(ii) If x is not K-critical, then $v(x) \neq 0$, $\theta(x) < 0$, $\varphi(x, v(x)) < -\frac{1}{2}\|v(x)\|^2 < 0$ and $v(x)$ is a K-descent direction;
(iii) The function $v(x)$ and $\theta(x)$ are continuous.

The following lemma is the final one of this section which will be useful for the convergent analysis of our method.

Lemma 2.3. *[29, Lemma 3] Let $x_1, x_2 \in \mathbb{R}^n$ with $\|x_1\| = \|x_2\| = 1$ and $\eta \geq 0$, then we have*

$$\|x_1 - x_2\| \leq 2\|x_1 - \eta x_2\|.$$

Now we recall Strong Wolfe condition in vector optimization.

Definition 2.3. [28, Definition 3.1] Let $d \in \mathbb{R}^n$ be a K-descent direction for F at x. Consider $0 < \rho < \sigma < 1$. We say that $\alpha > 0$ satisfies the strong Wolfe conditions if

$$F(x + \alpha d) \preceq_K F(x) + \rho\alpha\varphi(x, d)\, e, \tag{2.7}$$

$$|\varphi(x + \alpha d, d)| \leq \sigma |\varphi(x, d)|. \tag{2.8}$$

It is easy to see that $e \in K$ always exists [28]. Specially, for multiobjective optimization, we can take $e = [1, \ldots, 1]^\top \in \mathbb{R}^q$. It needs to be noted that the strong Wolfe line search are the inexact line search, and if $\alpha \geq 0$ is obtained by an exact line search, then $F(x + \alpha d, d) = 0$.

Remark 2.1. The strong Wolfe condition are well defined, and the positive step size can be found, see [28].

3. Modified Alternative Extension of the Hager-Zhang Conjugate Gradient Method for Vector Optimization

In this section, we will propose a new efficient modified alternative extension of the Hager-Zhang conjugate gradient method and give the algorithm description. Throughout the paper, we have assumed that the following two assumptions hold true.

Assumption 3.1. Assume $f_j : \mathbb{R}^n \to \mathbb{R}$, $j \in < q >$ are continuously differentiable. And the level set $L(x_0) = \{x \in \mathbb{R}^n \mid F(x) \preceq F(x_0)\}$ is bounded. $F(L(x_0))$ are bounded below, i.e., there exists $A \in \mathbb{R}^q$ such that $A \preceq_K F(x)$ for all $x \in L(x_0)$.

Assumption 3.2. Suppose there exists an open set $\mathcal{N}$ such that $L(x_0) \subset \mathcal{N}$ and $JF(x)$ is Lipschitz continuous on $\mathcal{N}$ with constant $L > 0$, i.e., $\forall x, y \in L(x_0), L > 0$, then $\|JF(x) - JF(y)\| \leq L\|x - y\|$.

Under Assumption 3.1 and 3.2, the following proposition propose the general method

$$x_{k+1} = x_k + \alpha_k d_k \tag{3.1}$$

satisfies Zoutendijk conditions, where α_k is step size, d_k is the search direction.

Proposition 3.1. *[28, Proposition 3.3] Suppose that Assumptions 3.1 and 3.2 hold. Consider an iteration of the form (3.1), where d_k is a descent direction for F at x_k and α_k satisfies the strong Wolfe conditions (2.7)-(2.8). Then*

$$\sum_{k \geq 0} \frac{\varphi^2(x_k, d_k)}{\|d_k\|^2} < \infty. \tag{3.2}$$

Next, we describe our proposed MAVHZ method. The MAVHZ method for solving problem (1.1) is given by (3.1) and the search direction d_k is defined by

$$d_k = \begin{cases} v(x_k), & if \quad k = 0, \\ v(x_k) + \beta_k d_{k-1}, & if \quad k \geq 1, \end{cases} \tag{3.3}$$

where β_k is an algorithm parameter.

In [30], the conjugate parameter is defined by $\beta_k = \max\left\{0, \beta_k^{AVHZ}\right\}$, where

$$\beta_k^{AVHZ} = \frac{1}{\tau_2 - \tau_3}\left(\tau_1 - \mu\tau_2 \frac{\|JF(x_k) - JF(x_{k-1})\|_{2,\infty}^2}{\tau_2 - \tau_3}\right)$$

and $\mu > \frac{1}{2}$, $\tau_1 = -\varphi(x_k, v(x_k)) + \varphi(x_{k-1}, v(x_k))$, $\tau_2 = \varphi(x_k, d_{k-1})$, $\tau_3 = \varphi(x_{k-1}, d_{k-1})$, $\tau_4 = \varphi(x_k, v(x_{k-1})) - \varphi(x_{k-1}, v(x_{k-1}))$.

In [36], a modified HZ conjugate gradient algorithm without gradient Lipschitz continuous condition for single-objective optimization is proposed. The conjugate parameter is defined by $\beta_k = \max\left\{\eta_k, \beta_k^{MHZ}\right\}$, where $\eta_k = \frac{-1}{\|d_{k-1}\| \min\{\eta, \|g_{k-1}\|\}}$. And β_k^{MHZ} is defined by

$$\beta_k^{MHZ} = \frac{y_k^T g_k \left(d_{k-1}^T y_k\right) - 2\|y_k\|^2 \left(g_k^T d_{k-1}\right)}{\max\left\{\mu \|y_k\|^2 \|d_{k-1}\|^2, \left(d_{k-1}^T y_k\right)^2\right\}}$$

where $\mu > \frac{1}{2}$.

Inspired by [30] and [36], we propose a new parametric β_k^{MAVHZ} is defined by

$$\begin{aligned}\beta_k^{MAVHZ} = & \frac{\tau_1}{\max\left\{\mu_1 \|JF(x_k) - JF(x_{k-1})\|_{2,\infty} \|d_{k-1}\|, (\tau_2 - \tau_3)\right\}} \\ & - \frac{\mu\tau_2 \|JF(x_k) - JF(x_{k-1})\|_{2,\infty}^2}{(\tau_2 - \tau_3)^2},\end{aligned} \tag{3.4}$$

where μ and μ_1 satisfies $\mu_1^2\mu > \frac{1}{2}$. It is this difference that makes the MAVHZ method to preserve the good scalar properties of the HZ method, i.e., the generated search direction is sufficiently descent without any line search. On the other hand, our algorithm performs better in time as a performance compared to the AVHZ method which also achieves the descending direction without sufficient line search.

Now we present a new efficient alternative extension of the Hager-Zhang conjugate gradient method for vector optimization for solving vector optimization.

Algorithm 3.1. [MAVHZ]

Step 1. Given starting point $x_0 \in \mathbb{R}^n$. Let $0 < \rho < \sigma < 1$, $\epsilon > 0$. Choose $\mu_1 > 0$, $\mu > 0$ such that it satisfies $p := \mu_1^2\mu > \frac{1}{2}$ and $e \in K$ such that $0 < \langle u, e\rangle \leq 1$ for any $u \in X$. Let $k := 0$.

Step 2. Compute $v(x_k)$ from (2.5). If $|\theta(x_k)| \leq \epsilon$, then stop.

Step 3. Compute $\beta_k = \max\left\{0, \beta_k^{MAVHZ}\right\}$ by the formula (3.4), and search direction d_k by the formula (3.3), i.e.

$$d_k = \begin{cases} v(x_k), & if \quad k = 0, \\ v(x_k) + \beta_k d_{k-1}, & if \quad k \geq 1. \end{cases}$$

Step 4. Compute step size α_k by the strong Wolfe line search procedure.

Step 5. Let $x_{k+1} = x_k + \alpha_k d_k$ and $k = k + 1$, turn to Step 2.

Remark 3.1. (i) In Algorithm 3.1, the subproblem (2.5) is solved only once at each iteration. However, in reference [29], they used an inner loop to determine a direction that satisfies the sufficient descent condition. This approach makes their algorithm computationally more expensive, since subproblems are likely to be solved multiple times in a single iteration. In [30], AVHZ produces a search direction that satisfies the sufficient descent condition without any line search operations. This is achieved by imposing restrictions on the nonnegativity of the covariance parameter (see [30] Property 4.1). Our proposed Algorithm 3.1, MAVHZ presents a new parametric form that also satisfies the method of finding a direction that satisfies the sufficient descent condition independently of any line search operation, the proof of which is given in Proposition 3.2. And our proposed Algorithm 3.1 (MAVHZ), in contrast to VHZ, does not require the computationally more expensive inner loop guaranteeing to find directions that satisfy the sufficient descent condition, and also can be faster and more efficient than the AVHZ method (from numerical experiments can be concluded).

(ii) In the VHZ method, the value range of the parameters μ in β_k^{VHZ} is $\mu > \frac{1}{4}$, while in AVHZ, the value range of the parameter μ in β_k^{AVHZ} is $\mu > \frac{1}{2}$, while in the MAVHZ method the value range of the two parameters μ_1 and μ only need to satisfy $p := \mu_1^2\mu > 1/2$, while we are adjusting the experimental effect, the adjustability of the two parameters can allow us to adjust the experimental effect to the optimum as far as possible during the experiment.

In some cases, more stringent descent conditions are introduced.

Definition 3.1. [28] We call d_k satisfies the sufficient descent condition at x_k if

$$\varphi(x_k, d_k) \leq c\varphi(x_k, v(x_k)), \tag{3.5}$$

for some $c > 0$ and for all $k \geq 0$.

Lemma 3.1. *Suppose that Assumption 3.2 holds. Then we can get*

$$|\varphi(x, d) - \varphi(y, d)| \leq \|d\|\|JF(x) - JF(y)\|_{2,\infty}. \tag{3.6}$$

Proof. The proof can see in [30, Lemma 4.1] □

The following proposition shows the sequence d_k generated by MAVHZ can satisfy the the sufficient descent condition.

Proposition 3.2. *Suppose that Assumption 3.2 holds. Let the search direction d_k be generated by Algorithm 3.1. Then, for arbitrary $k \geq 0$,*

$$\varphi\left(x_k, d_k\right) \leq \left(1 - \frac{1}{2p}\right) \varphi\left(x_k, v\left(x_k\right)\right),$$

where $p := \mu_1^2 \mu > \frac{1}{2}$.

Proof. If $\beta_k = 0$, then $d_k = v(x_k)$ from (3.2). In view of Lemma 2.3 (ii), we conclude that

$$\varphi\left(x_k, d_k\right) = \varphi\left(x_k, v\left(x_k\right)\right) \leq \left(1 - \frac{1}{2p}\right) \varphi\left(x_k, v\left(x_k\right)\right).$$

If $\beta_k > 0$, from the definition of d_k , (3.3) and Lemma 2.2 (ii), we have

$$\varphi\left(x_k, d_k\right) \leq \varphi\left(x_k, v\left(x_k\right)\right) + \beta_k \varphi\left(x_k, d_{k-1}\right). \tag{3.7}$$

On the other hand, from Lemma 3.1, we can get

$$\left|-\varphi\left(x_k, v\left(x_k\right)\right) + \varphi\left(x_{k-1}, v\left(x_k\right)\right)\right| \leq \left\|v\left(x_k\right)\right\| \left\|JF\left(x_k\right) - JF\left(x_{k-1}\right)\right\|_{2,\infty}. \tag{3.8}$$

Hence,

$$\begin{aligned}
\beta_k^{MAVHZ} \varphi(x_k, d_{k-1}) =& \frac{\tau_1 \tau_2}{\max\left\{\mu_1 \left\|JF(x_k) - JF(x_{k-1})\right\|_{2,\infty} \left\|d_{k-1}\right\|, (\tau_2 - \tau_3)\right\}} \\
& - \frac{\mu \tau_2^2 \left\|JF(x_k) - JF(x_{k-1})\right\|_{2,\infty}^2}{(\tau_2 - \tau_3)^2}, \\
\leq& \frac{\left|-\varphi\left(x_k, v\left(x_k\right)\right) + \varphi\left(x_{k-1}, v\left(x_k\right)\right)\right| \left|\varphi\left(x_k, d_{k-1}\right)\right|}{\max\left\{\mu_1 \left\|JF(x_k) - JF(x_{k-1})\right\|_{2,\infty} \left\|d_{k-1}\right\|, (\tau_2 - \tau_3)\right\}} \\
& - \frac{\mu \tau_2^2 \left\|JF(x_k) - JF(x_{k-1})\right\|_{2,\infty}^2}{(\tau_2 - \tau_3)^2}, \\
\leq& \frac{\left\|JF(x_k) - JF(x_{k-1})\right\|_{2,\infty} \left\|v(x_k)\right\| \left|\varphi(x_k, d_{k-1})\right|}{\mu_1 \left\|JF(x_k) - JF(x_{k-1})\right\|_{2,\infty} \left\|d_{k-1}\right\|} \\
& - \frac{\mu \left\|JF(x_k) - JF(x_{k-1})\right\|_{2,\infty}^2 \tau_2^2}{\left\|JF(x_k) - JF(x_{k-1})\right\|_{2,\infty}^2 \left\|d_{k-1}\right\|^2}.
\end{aligned} \tag{3.9}$$

For the numerator of the first part in the above second inequality, we take

$$\begin{aligned}
a_1 &= \tfrac{\sqrt{2\mu_1\mu}\left|\varphi(x_k, d_{k-1})\right|}{\left\|d_{k-1}\right\|}, \\
a_2 &= \tfrac{\left\|v(x_k)\right\|}{\sqrt{2\mu_1\mu}}.
\end{aligned} \tag{3.10}$$

In view of $a_1 a_2 \leq \frac{1}{2}\left(a_1^2 + a_2^2\right)$, we obtain

$$\begin{aligned}
\beta_k \varphi\left(x_k, d_{k-1}\right) &\leq \frac{1}{\mu_1}\left(\frac{\left\|v(x_k)\right\|^2}{4\mu_1\mu} + \mu_1\mu \frac{\varphi^2(x_k, d_{k-1})}{\left\|d_{k-1}\right\|^2}\right) - \frac{\mu \tau_2^2}{\left\|d_{k-1}\right\|^2} \\
&\leq \frac{\left\|v(x_k)\right\|^2}{4\mu_1^2 \mu} \\
&= \frac{\left\|v(x_k)\right\|^2}{4p},
\end{aligned} \tag{3.11}$$

where $p := \mu_1^2\mu > \frac{1}{2}$.

Hence, it follows from the above inequality, (3.7) and Lemma 2.2 (ii)

$$\begin{aligned}\varphi\left(x_k, d_k\right) &\leq \varphi\left(x_k, v\left(x_k\right)\right)+\frac{1}{4p}\left\|v\left(x_k\right)\right\|^2 \\ &\leq \varphi\left(x_k, v\left(x_k\right)\right)-\frac{1}{2p}\varphi\left(x_k, v\left(x_k\right)\right) \\ &=\left(1-\frac{1}{2p}\right)\varphi\left(x_k, v\left(x_k\right)\right).\end{aligned} \tag{3.12}$$

□

Lemma 3.2. *[28] Suppose Assumptions 3.1 - 3.2 hold. Then, there exists a constant $\bar{r}>0$ such that*

$$\left\|v\left(x_k\right)\right\| \leq \bar{r}, \quad \forall k \geq 0.$$

We begin to establish the estimate for the parameter β_k .

Lemma 3.3. *Suppose Assumptions 3.1 - 3.2 hold. Assume that there exists a constant $\underline{r}>0$ such that*

$$\left\|v\left(x_k\right)\right\| \geq \underline{r}, \quad \forall k \geq 0. \tag{3.13}$$

Then, for each $k \geq 0$, we have

$$\beta_k \leq M\left\|x_k-x_{k-1}\right\|,$$

where

$$M=\frac{2}{(1-\sigma)\left(1-\frac{1}{2p}\right)\underline{r}^2}\left(L\bar{r}+2\mu L^2 Q \max\left\{\frac{\sigma}{1-\sigma}, 1\right\}\right).$$

Proof. If $\beta_k=0$, then $\beta_k=0 \leq M\left\|x_k-x_{k-1}\right\|$. If $\beta_k>0$, then

$$\begin{aligned}\beta_k^{MAVHZ} &= \frac{\tau_1}{\max\left\{\mu_1\left\|JF(x_k)-JF(x_{k-1})\right\|_{2,\infty}\left\|d_{k-1}\right\|,(\tau_2-\tau_3)\right\}} \\ &\quad -\frac{\mu\tau_2\left\|JF(x_k)-JF(x_{k-1})\right\|_{2,\infty}^2}{(\tau_2-\tau_3)^2} \\ &\leq \frac{\left|\tau_1\right|}{\left|\tau_2-\tau_3\right|}+\frac{\mu\left|\tau_2\right|\left\|JF(x_k)-JF(x_{k-1})\right\|_{2,\infty}^2}{\left(\tau_2-\tau_3\right)^2} \\ &\leq \frac{1}{\left|\tau_2-\tau_3\right|}\left(\left|\tau_1\right|+\mu\left|\tau_2\right|\frac{\left\|JF\left(x_k\right)-JF\left(x_{k-1}\right)\right\|_{2,\infty}^2}{\left|\tau_2-\tau_3\right|}\right).\end{aligned} \tag{3.14}$$

From the definition of τ_1, τ_2, τ_3 , Assumption 3.2 and Lemma 3.1, Lemma 3.2, we obtain

$$\left|\tau_1\right| \leq L\bar{r}\left\|x_k-x_{k-1}\right\|, \tag{3.15}$$

and

$$\tau_2-\tau_3 \geq \frac{1}{2}(1-\sigma)\left(1-\frac{1}{2p}\right)\underline{r}^2. \tag{3.16}$$

Then for arbitrary $k \geq 0$, $\sigma \in (0,1)$ then

$$\frac{\left|\tau_2\right|}{\left|\tau_2-\tau_3\right|} \leq \max\left\{\frac{\sigma}{1-\sigma}, 1\right\}, \tag{3.17}$$

In view of the (2.7)-(2.8) and Assumption 3.1, the following relationships hold,

$$x_{k-1}, x_k \in L(x_0) \Rightarrow \left\|x_{k-1}\right\| \leq Q, \left\|x_k\right\| \leq Q.$$

Then

$$\|JF(x_k) - JF(x_{k-1})\|_{2,\infty}^2 \leq 2QL^2 \|x_k - x_{k-1}\|. \tag{3.18}$$

The detailed proof procedure for the above four formulas, i.e. (3.15) - (3.18) are given in the proof procedure for Lemma 4.3 in [30] and is omitted here. According to (3.15) - (3.18), we can obtain

$$\beta_k \leq \frac{2}{(1-\sigma)\left(1-\frac{1}{2p}\right)\underline{r}^2}\left(L\bar{r} + 2\mu L^2 Q \max\left\{\frac{\sigma}{1-\sigma}, 1\right\}\right)\|x_k - x_{k-1}\|. \tag{3.19}$$

The proof is complete. □

The following lemma is crucial in the proof of global convergence.

Lemma 3.4. *Assume Assumptions 3.1-3.2 and (3.13) hold. Then, $d_k \neq 0$, for each $k \geq 0$, we have*

$$\sum_{k=1}^{\infty} \frac{1}{\|d_k\|^2} < \infty, \quad \sum_{k=1}^{\infty} \|q_k - q_{k-1}\|^2 < \infty, \tag{3.20}$$

where $q_k = \frac{d_k}{\|d_k\|}$.

Proof. By Proposition 3.2, it follows that the direction d_k generated by Algorithm 3.1 satisfies the sufficiently descent condition for all $k \geq 0$. Then $d_k = 0$ and q_k is well-defined. Considering an iteration $x_{k+1} = x_k + \alpha_k d_k$, where d_k is a K-descent direction for F at x_k and α_k satisfies the strong Wolfe conditions, then the following Zoutendijk-type condition holds (see Proposition 3.3 in [28]):

$$\sum_{k=0}^{\infty} \frac{\varphi^2(x_k, d_k)}{\|d_k\|^2} < \infty.$$

According to Lemma 4.4 in [30], Lemma 3.4 in this paper is proved. □

Let us now present the main convergence results of our paper.

Theorem 3.1. *Let Assumptions 3.1 - 3.2 hold. Then,*

$$\lim_{k\to\infty} \inf \|v(x_k)\| = 0. \tag{3.21}$$

Proof. Its proof is similar to the one of Theorem 3.1 in [29] and is omitted. □

4. Numerical Experiments

In this section, we present some numerical experiments to demonstrate the effectiveness of the MAVHZ method. Comparisons with the AVHZ method [30] and the HZ method [29] is discussed. All codes runs in MATLAB (2018a) and executed on a laptop Intel(R) Core(TM) i5-7300HQ CPU(2.50 GHz) and 20GB RAM.

In our experiments, the chosen test problems consist of 33 convex and nonconvex multiobjective optimization problems. These problems are taken from the multiobjective optimization literature, i.e., $K = \mathbb{R}^n_+$, C is the canonical basis of $\mathbb{R}^n$, and $e = [1, ..., 1]^T \in \mathbb{R}^n$. More detailed and main characteristics of each problem is presented in Table 1. The first two columns identify the problem name and the corresponding reference. The third and fourth columns "q" and "n" represent the numbers of objectives and variables of test problems, respectively. "Convex" informs whether the problem is convex or not. For each problem, the starting points are generated belonging to the box constraints $\{x \in \mathbb{R}^n | lb \leq x \leq ub\}$, where the lower bound $lb \in \mathbb{R}^n$ and the upper bound $ub \in \mathbb{R}^n$ are reported in the fourth and fifth columns of the table, respectively. For the box constraints, we simply take the starting point inside the corresponding box and ignore the constraints.

TABLE 1. List of test problems

Problem	References	q	n	lb^T	ub^T	Convex
BK1	[37]	2	2	(-5, -5)	(10, 10)	Y
DD1	[38]	2	5	(-20,..., -20)	(20, ..., 20)	N
DGO1	[37]	2	1	-10	13	N
DGO2	[37]	2	1	-9	9	Y
FDS-1	[19]	3	5	(-2,..., -2)	(2, ..., 2)	Y
FDS-2	[19]	3	1000	(-2,..., -2)	(2, ..., 2)	Y
FDS-3	[19]	3	2000	(-2,..., -2)	(2, ..., 2)	Y
FDS-4	[19]	3	4000	(-2,..., -2)	(2, ..., 2)	Y
FF1	[37]	2	2	(-1,-1)	(1,1)	N
Hil1	[39]	2	2	(0,0)	(1,1)	N
JOS1-1	[40]	2	10	(-100,-100)	(100,100)	Y
JOS1-2	[40]	2	100	(-100,...,-100)	(100,...,100)	Y
JOS1-3	[40]	2	1000	(-100,...,-100)	(100,...,100)	Y
MLF1	[37]	2	1	0	20	N
MLF2	[37]	2	2	(-100,-100)	(100,100)	N
MMR1	[37]	2	2	(0,0)	(1,1)	N
MOP1	[37]	2	1	-10^5	10^5	N
MOP2	[37]	2	2	(-1,-1)	(1,1)	N
MOP3	[37]	2	2	$(-\pi, -\pi)$	(π, π)	N
PNR	[41]	2	2	(-2,-2)	(2,2)	Y
SLCDT2-1	[42]	3	10	(-1,...,-1)	(1,...,1)	Y
SLCDT2-2	[42]	3	100	(-1,...,-1)	(1,...,1)	Y
SP1	[37]	2	2	(-100,-100)	(100,100)	Y
SSFYY2	[37]	2	1	(-100,-100)	(100,100)	N
SK1	[37]	2	1	-100	100	N
SK2	[37]	2	4	(-10,...,-10)	(10,...,10)	N
TE1	[43]	2	2	(-1,-1)	(1,1)	Y
TE2	[43]	2	2	(-2,-2)	(2,2)	Y
TE4	[43]	2	10	(-10,...,-10)	(10,...,10)	Y
TE6	[43]	2	2	(0,0)	(100,100)	Y
TE7	[43]	3	3	(0,0,0)	(30,30,30)	Y
Toi4	[44]	2	4	(-2,...,-2)	(5,...,5)	Y
VU1	[37]	2	2	(-3,-3)	(3,3)	N

During the numerical experiments, we compare the AVHZ method with the following three methods:
1. The VHZ method [29], with $\beta_k = \max\left(\beta_k^{VHZ}, \frac{-1}{\|d_k\| \min\{\lambda, \|v(x_k)\|\}}\right)$, where

$$\begin{aligned} \beta_k^{VHZ} = &\frac{-H\left(x_k, v\left(x_k\right)\right)+H\left(x_{k-1}, v\left(x_k\right)\right)}{H\left(x_k, d_{k-1}\right)-H\left(x_{k-1}, d_{k-1}\right)} \\ &-\mu H\left(x_k, d_{k-1}\right) \frac{-H\left(x_k, v\left(x_k\right)\right)+H\left(x_{k-1}, v\left(x_k\right)\right)+H\left(x_k, v\left(x_{k-1}\right)\right)-H\left(x_{k-1}, v\left(x_{k-1}\right)\right)}{\left(H\left(x_k, d_{k-1}\right)-H\left(x_{k-1}, d_{k-1}\right)\right)^2}, \end{aligned} \tag{4.1}$$

and $\mu = 1$. The VHZ method here is Algorithm 1 in [29].

2. The AVHZ method [30], with $\beta_k = \max\left\{0, \beta_k^{AVHZ}\right\}$, where

$$\beta_k^{AVHZ} = \frac{1}{\tau_2 - \tau_3}\left(\tau_1 - \mu\tau_2 \frac{\|JF(x_k) - JF(x_{k-1})\|_{2,\infty}^2}{\tau_2 - \tau_3}\right),$$

and $\mu = 1$.

As in [29, 30], we choose the parameter $\mu_1 = 2$, $\mu = 0.25$, and then $p = 1$ in the MAVHZ method. Regarding the steepest descent direction $v(x)$ and the optimal value $\theta(x)$, we use the built-in functions quadprog in Matlab to solve the subproblems (2.6). For computing a step size, we implemented an adapted version of the line search algorithm proposed in [45]. The MAVHZ method and the AVHZ method all use the strong Wolfe step size strategy, that is, to find a step size $\alpha > 0$ satisfying (2.7)- (2.8), where $\rho = 10^{-4}$, $\delta = 0.1$. The VHZ method utilizes a sufficiently accurate Wolfe line search, details of which can be found in [29]. In this experiments, the parameter selections of VHZ method are the same as the ones in [29]. In each iteration k, the choice of the initial trial value of the step size is crucially important for the performance of the line search. For $k = 0$ we take $\alpha_0 = \min\left(\max\left(1/\left\|v\left(x_0\right)\right\|, 1\right), \alpha_{\max}\right)$, for further iterations, we set α_k to

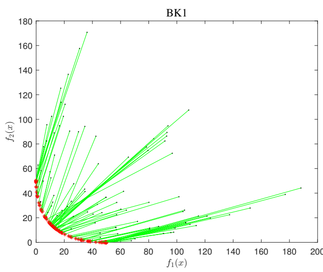
$$\min\left(\max\left(\alpha_{k-1}\frac{H\left(x_{k-1}, d_{k-1}\right)}{H\left(x_k, d_k\right)}, 0.02\right), 100\right),$$

where $\alpha_{max} = 10^{10}$. The selection of the initial step size is the same as the one in [29]. The termination condition of these algorithms is $\theta(x) \geq -5 \times \text{eps}^{1/2}(\text{eps} = 2^{-52} \approx 2.22 \times 10^{-16})$, or the number of iterations exceeds 5000.

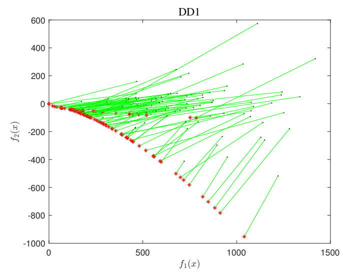
4.1. **Pareto frontiers.** Firstly, for each method, we solve all problems 100 times within the box indicated in the last two columns of Table 1 by using a uniform random distribution of starting points. If an approximate critical point is found, the run is considered successful regardless of the value of the objective function. The Pareto-frontiers for the problems are recorded in FIGURE 1. In FIGURE 1, the red points indicate the final iteration, while the blue starting point of a straight line segment indicates the corresponding initial point. The results in FIGURE 1 indicate that the MAVHZ method is capable of estimating the Pareto front for the considered problems with the appropriate number of starting points. To avoid redundancy in the article, we will only present the Pareto front of 9 questions here. FIGURE 1 present the Pareto frontiers of BK1, DD1, Hil1, PNR, MOP3, VU1, DGO1, DGO2 and FF1 problems. The Pareto frontiers of the other questions will not be displayed, but their generated Pareto frontiers are also good.

We use the MAVHZ method to solve the FDS, SLCDT2, T8 problem generating the Pareto frontier displayed in FIGURE 2. In the above FDS, SLCDT2 and T8 problems, m is all 3 and n is 100, 10 and 30 respectively. The Pareto frontier generated by our MAVHZ method to solve the FDS, SLCDT2, T8 problem is shown in FIGURE 2. Observing FIGURE 2, it can be noticed that the generated Pareto frontier works well. It shows that our proposed MAVHZ method is also very effective for solving 3-objective optimization problems.

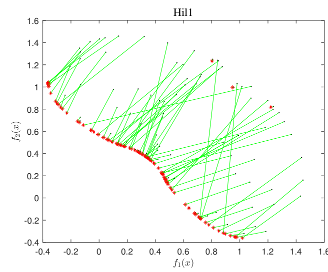
4.2. **Efficiency.** In this experiment, for each method, we solve all problems 100 times within the box indicated in the last two columns of Table 1 by using a uniform random distribution of starting points. The numerical results are presented in TABLE 2. The median value solution time (Time), the median value of number of estimated function values (nf), the median value of number of estimated gradient values (ng), and the median value of number of iterations (iter), are shown in TABLE 2.



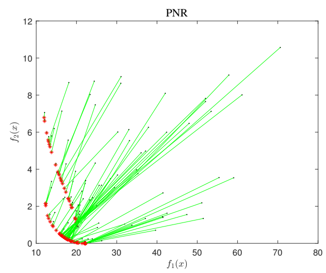
(A) BK1



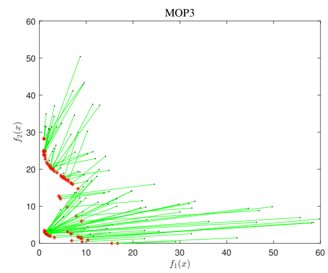
(B) DD1



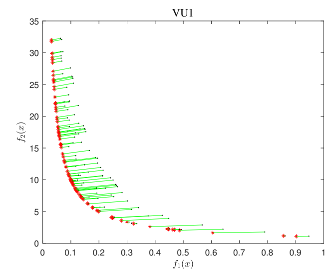
(C) Hil1



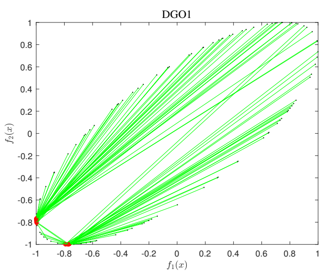
(D) PNR



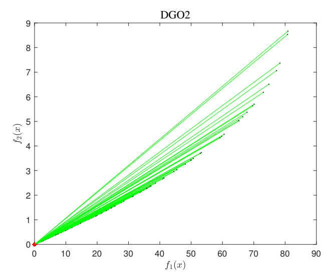
(E) MOP3



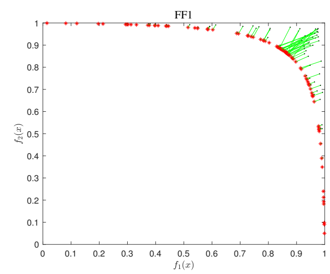
(F) VU1



(G) DGO1



(H) DGO2



(I) FF1

FIGURE 1. Pareto frontiers of BK1, DD1, Hil1, PNR, MOP3, VU1, DGO1, DGO2, FF1 generated by MAVHZ method for 100 random starting points

TABLE 2. Results of numerical experiments with three methods dealing with 27 problems

Problem	q	n	convex	method	iter	nf	ng	nv	time
BK1	2	2	Y	VHZ	6.5	112.5	40	7.5	0.0380
				AVHZ	5	1212.5	130.5	6	0.0337261
				MAVHZ	5	1237	142.5	6	0.0358
DGO1	2	1	N	VHZ	2	30	9	3	0.0151
				AVHZ	2	477.5	70	3	0.0237564
				MAVHZ	2	477.5	70	3	0.0083
DGO2	2	1	Y	VHZ	6.5	101	35.5	7.5	0.04372075
				AVHZ	3	1094.5	105.5	4	0.0317252
				MAVHZ	3	1108	103.5	4	0.02852305

continued on next page

Table 2. Results of numerical experiments with three methods dealing with 27 problems (continued)

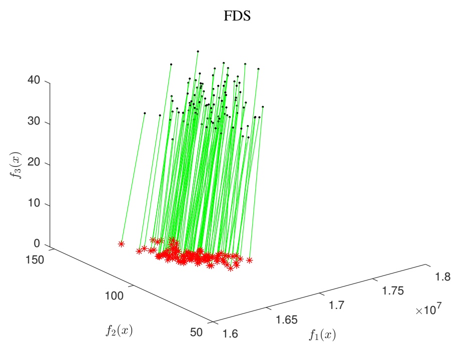
Problem	q	n	convex	method	iter	nf	ng	nv	time
FF1	2	2	N	VHZ	7	101	38	8	0.04749005
				AVHZ	14	257	168	15	0.13555085
				MAVHZ	14	226.5	146	15	0.08464915
JOS1-1	2	10	Y	VHZ	7	249	39	8	0.07339845
				AVHZ	4	2214	81	5	0.0452304
				MAVHZ	4	2301	86	5	0.0433043
JOS1-2	2	100	Y	VHZ	119	4605	711	120	1.20934465
				AVHZ	4	2214	81	5	0.0533879
				MAVHZ	5	3081	111.5	6	0.0528752
JOS1-3	2	1000	Y	VHZ	114	4410	681	115	15.6235261
				AVHZ	5	2951	107	6	0.60502795
				MAVHZ	5	3224	123	6	0.58618355
MLF1	2	1	N	VHZ	2	7	7	3	0.01199595
				AVHZ	2	7	7	3	0.0139261
				MAVHZ	2	7	7	3	0.01301455
MLF2	2	2	N	VHZ	6	124.5	37.5	7	0.04377725
				AVHZ	5	1855	140.5	6	0.0597104
				MAVHZ	6	1893	153	7	0.04780855
TE1	2	2	Y	VHZ	7	109.5	40	8	0.0496057
				AVHZ	4	1245	94.5	5	0.04715075
				MAVHZ	4	1246	116	5	0.03989395
TE2	2	2	Y	VHZ	5	57.5	27	6	0.0374122
				AVHZ	4	231.5	87	5	0.04913145
				MAVHZ	4	135	83	5	0.0306218
TE4	2	10	Y	VHZ	19	328	109	20	0.1400238
				AVHZ	7	2155.5	174.5	8	0.08047715
				MAVHZ	8	2165	185	9	0.0760076
TE6	2	2	Y	VHZ	4	93	23	5	0.02917795
				AVHZ	5	1512.5	97	6	0.05867275
				MAVHZ	4	1355.5	81	5	0.0378474
TE7	3	3	Y	VHZ	7	128	40	8	0.0580934
				AVHZ	4	1939	96.5	5	0.04867355
				MAVHZ	4	1951	101.5	5	0.04366045
SP1	2	2	Y	VHZ	9	174	54	10	0.0678989
				AVHZ	6.5	2116.5	152.5	7.5	0.073256
				MAVHZ	6	2042.5	145	7	0.04601215
SSFYY2	2	1	N	VHZ	4	69	19	5	0.02552565
				AVHZ	3	1041.5	76	4	0.0259069
				MAVHZ	3	1040	73	4	0.01979175
SK1	2	1	N	VHZ	3	104	17	4	0.01710975
				AVHZ	3	1778	86	4	0.0341301
				MAVHZ	3	1778	86	4	0.0235814

continued on next page

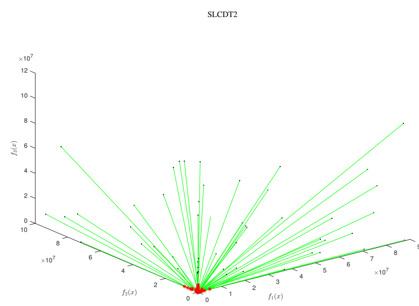
TABLE 2. Results of numerical experiments with three methods dealing with 27 problems (continued)

Problem	q	n	convex	method	iter	nf	ng	nv	time
SK2	2	4	N	VHZ	12	199.5	78.5	13	0.0949967
				AVHZ	8.5	1412	178	9.5	0.09487715
				MAVHZ	8	1355	161	9	0.0614213
VU1	2	2	N	VHZ	21	157.5	111	22	0.14975805
				AVHZ	102.5	857.5	682	103.5	0.9096893
				MAVHZ	101.5	840.5	679	102.5	0.7402291
MOP1	2	1	N	VHZ	14	432	81	15	0.0862149
				AVHZ	4	1855.5	75	5	0.0385516
				MAVHZ	4	1827	72	5	0.03325
MOP2	2	2	N	VHZ	1	3	3	2	0.01034505
				AVHZ	1	3	3	2	0.0121187
				MAVHZ	1	3	3	2	0.0101677
MOP3	2	2	N	VHZ	8	122.5	47	9	0.057948
				AVHZ	7	708.5	185.5	8	0.07260785
				MAVHZ	7	673	183.5	8	0.06137405
DD1	2	5	N	VHZ	9	105	50	10	0.0658685
				AVHZ	13	1372.5	198	14	0.1319392
				MAVHZ	12	1366.5	186	13	0.1024599
Toi4	2	4	Y	VHZ	9	124	52	10	0.06602925
				AVHZ	4	1083.5	129	5	0.0487049
				MAVHZ	5	1122	138	6	0.04071285
Hil1	2	2	N	VHZ	7	106	47	8	0.0541381
				AVHZ	11	327	157	12	0.1155667
				MAVHZ	11	269.5	159.5	12	0.09403925
PNR	2	2	Y	VHZ	7	72.5	40	8	0.0538202
				AVHZ	7	177	123.5	8	0.07391875
				MAVHZ	7	175.5	119.5	8	0.05821845
MMR1	2	2	N	VHZ	6	61	33	7	0.04725005
				AVHZ	10	162	106.5	11	0.0987681
				MAVHZ	10	175	109.5	11	0.07795605

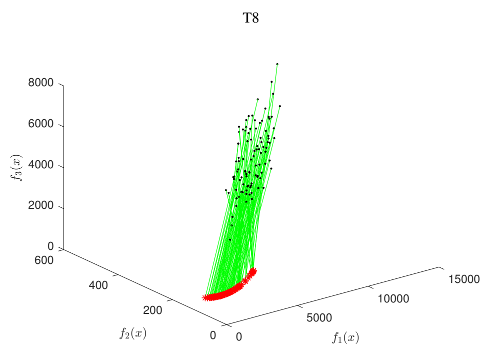
Based on Table 2, when considering the CPU time as an evaluation metric, the MAVHZ method outperforms the AVHZ method in the following all problem instances. In comparison to the AVHZ and VHZ methods, the MAVHZ method performs best in terms of time in the following 16 problem instances: BK1, DGO1, DGO2, JOS1 (the first to the third instance), T1, T27, T4, T7, SP1, SSFYY2, SK2, MOP1, MOP2, Toi4. When we consider the three metrics nf, ng, nv, MAVHZ performs better than AVHZ in these following questions: DG01, FF1, MLF1, T27, T6, SP1, SSFYY2, SK1, SK2, VU1, MOP1, MOP2, MOP3, DD1, HiL1 and PNR. And for all experimental examples, we find that MAVHZ has fewer iterations than AVHZ. Meanwhile, we list the results of solving the high-dimensional FDS problem with the three methods MAVHZ, AVHZ, and VHZ in Tables 3-5. We can see that for the high-dimensional FDS problem, MAVHZ and AVHZ are able to solve the problem very well and successfully, and MAVHZ is more advantageous in all four aspects: iter, nf, ng, nv, and time. The VHZ method is not effective in solving the high-dimensional FDS problem, and is basically unable to solve the problem.



(A) FDS



(B) SLCDT2



(C) T8

FIGURE 2. Pareto frontiers of FDS, SLCDT2, T8 generated by MAVHZ method for 100 random starting points

TABLE 3. Numerical results for MAVHZ and AVHZ and VHZ solving FDS high-dimensional data with $n = 1000$

FDS				
n		VHZ	AVHZ	MAVHZ
1000	iter	NaN	90	88
	nf	NaN	1125	1141.5
	ng	NaN	743	757
	nv	NaN	91	89
	time	4.7496	12.3485	12.1688

TABLE 4. Numerical results for MAVHZ and AVHZ and VHZ solving FDS high-dimensional data with $n = 2000$

FDS				
n		VHZ	AVHZ	MAVHZ
2000	iter	NaN	95	93
	nf	NaN	1111.5	1098.5
	ng	NaN	795.5	802
	nv	NaN	96	94
	time	33.3442 19	51.7039	48.8044

TABLE 5. Numerical results for MAVHZ and AVHZ and VHZ solving FDS high-dimensional data with $n = 4000$

FDS				
n		VHZ	AVHZ	MAVHZ
4000	iter	NaN	100	98
	nf	NaN	1121	1131.5
	ng	NaN	827	841.5
	nv	NaN	101	99
	time	324.7496	353.4476	335.472

In order to compare the performance of the tested methods more clearly. We adopt the performance profiles recommended by Dolan and Mor¨¦ [46]. In this figure,

$$\rho(\tau) = \frac{1}{|P|} \left|\{p \in P : r_{p,z} \leq \tau\}\right|, \quad r_{p,z} = \frac{t_{p,z}}{\min\{t_{p,z}, z \in Z\}},$$

where P represents the problem set consisting of the test problem p, Z represents the algorithm set consisting of the algorithm z, and $t_{p,z}$ is the performance (for example, number of iterations) obtained by using the algorithm z to solve the problem p. In the performance profile graphic, $\rho(1)$ is the fraction of problems for which the solver z is the most efficient over all the algorithms. On the other hand, the robustness is to verify the capacity of the methods to solve the greatest possible number of problems. Therefore, the efficiency and robustness of the algorithm can be accessed on the extreme left (at 1 in the domain) and right of the graph, respectively.

To provide a more intuitive representation of the numerical results, FIGURE 3 illustrates the performance profiles associated with the number of iterations ($iter$), function evaluations (nf), gradient evaluations (ng), vector steepest descent evaluations (nv), and CPU time (time). It is observed that the MAVHZ method significantly outperforms its competitors in terms of CPU time for finding smooth points on the test problem. Comparison with the VHZ method requires more functions. When $\tau < 2$, the MAVHZ method outperforms the other methods in terms of the number of iterations and the number of vector steepest descent evaluations. The MAVHZ method outperforms AVHZ in terms of the number of functions evaluated and the number of gradients evaluated.

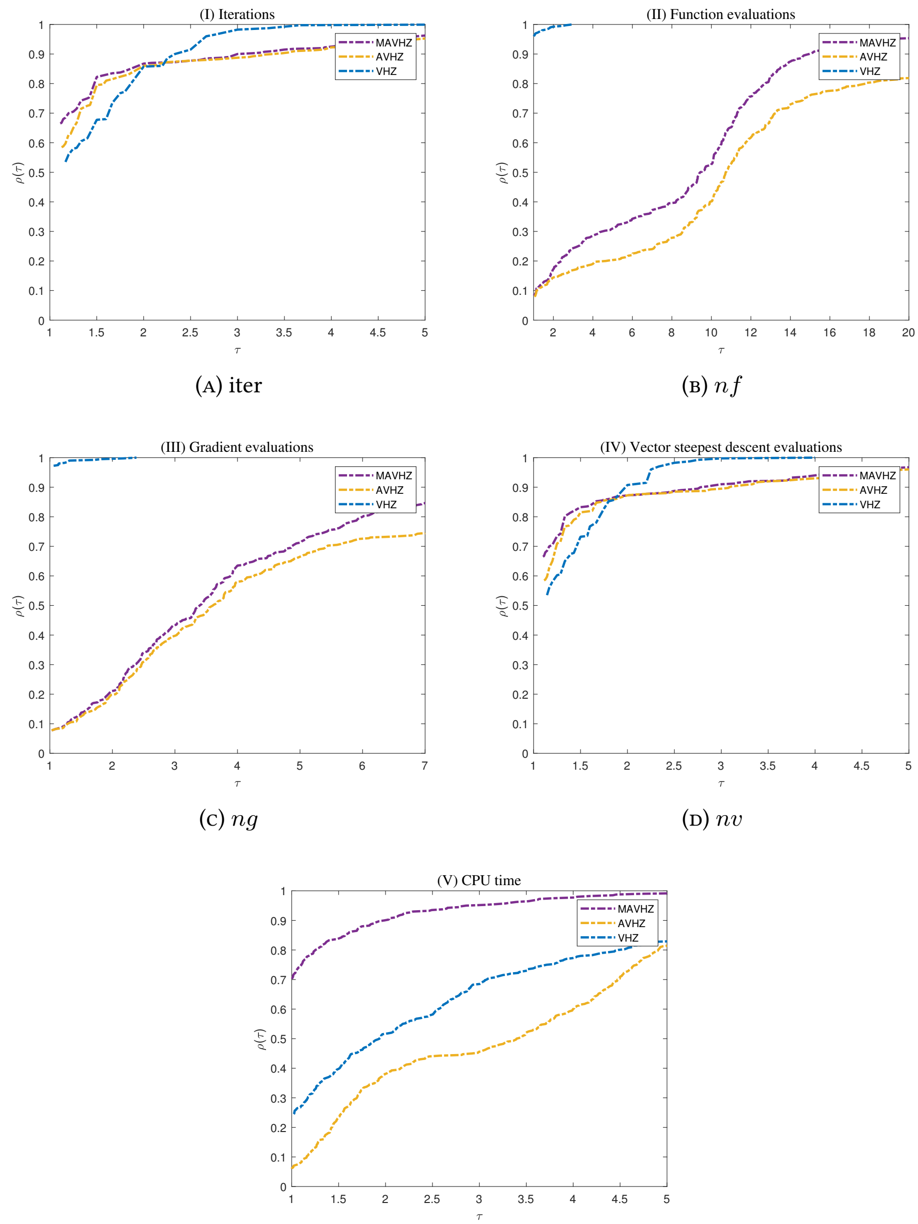
The MAVHZ method demonstrates superior performance compared to the other two methods, as evident from Tables 2-5, and FIGURE 1-3. Therefore, the proposed method shows great promise.

5. CONCLUSIONS

In this paper, we build the MAVHZ method for solving the vector optimization problems. The convergence results of the MAVHZ method are established under some mild conditions. The proposed method is ultimately employed to tackle unconstrained multiobjective optimization problems, and the numerical results demonstrate that the MAVHZ method surpasses the existing approaches in terms of efficiency. This method not only generates search directions that descend sufficiently independently of any line searches, but also has a faster running time than the AVHZ method, which also generates independently sufficiently descending search directions. Our future work focuses on developing faster algorithms for solving high-dimensional multiobjective problems.

ACKNOWLEDGEMENTS

This paper was partially supported by the the NSFC (12471299, 12201514, 12071379), the Natural Science Foundation of Chongqing (CSTB2025NSCQ-GPX0529), and the Southwest University Graduate Research Innovation Program (No. SWUS25065).



(A) iter

(B) nf

(C) ng

(D) nv

(E) $time$

Figure 3. Performance profiles with 100 starting points for each test problem

Statements and Declarations

The authors declare that they have no conflict of interest.

References

[1] U. Bagchi. Simultaneous minimization of mean and variation of flow time and waiting time in single machine systems. *Operational Research*, 37:118-125, 1989.

[2] H. Eschenauer, J. Koski, and A. Osyczka. *Multicriteria Design Optimization*. Springer, Berlin, 1990.

[3] T. Leschine, H. Wallenius, and W. Verdini. Interactive multiobjective analysis and assimilative capacity-based ocean disposal decisions. *European Journal of Operational Research*, 56:278-289, 1992.

[4] S. Lipovetsky and W. Michael Conklin. Ridge regression in two-parameter solution. *Applied Stochastic Models in Business and Industry*, 21:525-540, 2005.

[5] E. Schreibmann, M. Lahanas, L. Xing and D. Baltas. Multiobjective evolutionary optimization of the number of beams, their orientations and weights for intensity-modulated radiation therapy. *Physics in Medicine and Biology*, 49:747-770, 2004.

[6] M. Wiecek. Advances in cone-based preference modeling for decision making with multiple criteria. *Decision Making in Manufacturing and Services*, 1:153-173, 2007.

[7] G. P. Rangaiah and A. Bonilla-Petriciolet. *Multi-Objective Optimization in Chemical Engineering: Developments and Applications*, John Wiley and Sons, 2013.

[8] Y. C. Jin. *Multi-Objective Machine Learning*. Springer-Verlag, Berlin, 2006.

[9] M. Tavana, Mariya A. Sodenkamp, and L. Suhl. A soft multi-criteria decision analysis model with application to the European Union enlargement. *Annals of Operations Research*, 181:393-421, 2010.

[10] J. Chen and Y. Dai. Multiobjective optimization with least constraint violation: optimality conditions and exact penalization. *Journal of Global Optimization*, 87:807-830, 2023.

[11] J. Chen, Y. Bai, G. Yu, X. Ou, and X. Qin. A PRP type conjugate gradient method without truncation for nonconvex vector optimization. *Journal of Optimization Theory and Applications*. 204(1):Article ID 13, 2025.

[12] J. Chen, J. Liu, X. Qin, and J. C. Yao. A non-monotone proximal gradient algorithm for solving nonsmooth multiobjective optimization problems with an extending application to robust multiobjective optimization. *Journal of Computational and Applied Mathematics*, 460:Article ID 116422, 2025.

[13] J. Chen, Z. Liu, Y. Lv, X. Ou, and K. Zhao. An adaptive combined conjugate gradient method for multiobjective optimization problems beyond convexity. *Numerical Algorithms*, 2025. https://doi.org/10.1007/s11075-025-02278-4.

[14] R. Burachik, C. Kaya, and M. M. Rizvi. A new scalarization technique and new algorithms to generate pareto fronts. *SIAM Journal on Optimization*, 27:1010-1034, 2017.

[15] G. Eichfelder. An adaptive scalarization method in multiobjective optimization. *SIAM Journal on Optimization*, 19:1694-1718, 2009.

[16] A. M. Geoffrion. Proper efficiency and the theory of vector maximization. *Journal of Mathematical Analysis and Applications*, 22:618-630, 1968.

[17] J. Fliege and B. F. Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, 51:479-494, 2000.

[18] L. M. Graña. Drummond and B. F. Svaiter. A steepest descent method for vector optimization. *Journal of Computational and Applied Mathematics*, 175:395-414, 2005.

[19] J. Fliege, L. M. G. Drummond, and B. F. Svaiter. Newtons method for multiobjective optimization. *SIAM Journal on Optimization*, 20:602-626, 2009.

[20] J. Fliege, L. M. G. Drummond, and B. F. Svaiter. Newton-like methods for efficient solutions in vector optimization. *Computational Optimization and Applications*, 26:495-516, 2013.

[21] M. A. T. Ansary and G. Panda. A modified quasi-Newton method for vector optimization problem. *Optimization*, 64:2289-2306, 2015.

[22] L. M. Graña Drummond and A. N. Iusem. A projected gradient method for vector optimization problems. *Computational Optimization and Applications*, 28:5-29, 2004.

[23] J. Y. B. Cruz. A subgradient method for vector optimization problems. *SIAM Journal on Optimization*, 23:2169-2182, 2013.

[24] H. Bonnel, A. Iusem, and B. Svaite. Proximal methods in vector optimization. *SIAM Journal on Optimization*, 15:953-970, 2005.

[25] L. C. Ceng and J. C. Yao. Approximate proximal methods in vector optimization. *European Journal of Operational Research*, 183:1-19, 2007.

[26] Z. Y. Chen, X. X. Huang, and X. Q. Yang. Generalized proximal point algorithms for multiobjective optimization problems. *Applicable Analysis*, 90:935-949, 2011.

[27] K. D. Villacorta and P. R. Oliverira. An interior proximal method in vector optimization. *European Journal of Operational Research*, 214:485-492, 2011.

[28] L. R. Lucambio Pérez and L. F. Prudente. Nonlinear conjugate gradient methods for vector optimization. *SIAM Journal on Optimization*, 28:2690-2720, 2018.

[29] M. L. N. Goñalves and L. F. Prudente. On the extension of the Hager-Zhang conjugate gradient method for vector optimization. *Computational Optimization and Applications*, 76:889-916, 2020.

[30] Q. Hu, L. Zhu, and Y. Chen. Alternative extension of the Hager-Zhang conjugate gradient method for vector optimization. *Computational Optimization and Applications*, 88:217-250, 2024.

[31] M. L. N. Gonalves, F. S. Lima, and L. F. Prudente. A study of Liu-Storey conjugate gradient methods for vector optimization. *Applied Mathematics and Computation*, 425:Article ID 127099, 2022.

[32] J. Yahaya and P. Kumam. Efficient hybrid conjugate gradient techniques for vector optimization. *Results in Control and Optimization*, 14:Article ID 100348, 2024.

[33] H. C. Zhang, A. Conn, and K. Scheinberg, A derivative-free algorithm for least-squares minimization. *SIAM Journal on Optimization*, 20:3555-3576, 2010.

[34] G. Liuzzi, S. Lucidi, and F. Rinaldi. A derivative free approach to constrained multiobjective nonsmooth optimization. *SIAM Journal on Optimization*, 26:2744-2774, 2016.

[35] E. H. Fukuda and L. M. Grãna Drummond. A survey on multiobjective descent methods. *Pesquisa Operacional*, 34:585-620, 2014.

[36] G. L. Yuan, A. L. Jian, M. X. Zhang, and J. J. Yu. A modified HZ conjugate gradient algorithm without gradient Lipschitz continuous condition for non convex functions. *Journal of Applied Mathematics and Computing*, 68:4691-4712, 2022.

[37] S. Huband, P. Hingston, L. Barone, and L. While. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10:477-506, 2006.

[38] I. Das and J. E. Dennis. Normal-boundary intersection: a new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8:631-657, 1998.

[39] C. Hillermeier. Generalized homotopy approach to multiobjective optimization. *Journal of Optimization Theory and Applications*, 110:557-583, 2001.

[40] C. Hillermeier. Dynamic weighted aggregation for evolutionary multi-objective optimization: why does it work and how? In Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation, pages 1042-1049. San Francisco, California, USA, 2001.

[41] M. Preuss, B. Naujoks, and G. Rudolph. *Pareto Set and EMOA Behavior for Simple Multimodal Multiobjective Functions*, In International Conference on Parallel Problem Solving from Nature, pages 513-522. Springer-Berlin, Heidelberg, 2006.

[42] O. Schütze, M. Laumanns, C. A. Coello, M. Dellnitz, and E. G. Talbi. Convergence of stochastic search algorithms to finite size Pareto set approximations. *Journal of Global Optimization*, 41:559-577, 2008.

[43] J. Thomann and G. Eichfelder. Numerical results for the multiobjective trust region algorithm MHT. *Data in Brief*, 25:Article ID 104103, 2019.

[44] P. L. Toint. *Test Problems for Partially Separable Optimization and Results for the Routine PSPMIN*. Technical Report, The University of Namur, Department of Mathematics, Belgium, 1983.

[45] L. R. Lucambio Pérez and L.F. Prudente. A Wolfe line search algorithm for vector optimization. *ACM Transactions on Mathematical Software*, 45:1-23, 2019.

[46] E. D. Dolan and J. J. More. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201-213, 2002.