EUROPEAN JOURNAL OF PURE AND APPLIED MATHEMATICS
2025, Vol. 18, Issue 1, Article Number 5551
ISSN 1307-5543 – ejpam.com
Published by New York Business Global

New Newton Group Iterative Methods for Solving Large-Scale Multi-Objective Constrained Optimization Problems

Peng Cheng[1,2], Jumat Sulaiman[1,*], Khadizah Ghazali[1], Majid Khan Majahar Ali[3], Ming Ming Xu[4]

[1] *Faculty of Science and Natural Resources, Universiti Malaysia Sabah, Jalan UMS, 88400 Kota Kinabalu, Sabah, Malaysia*
[2] *Chongqing College of Finance and Economics, 402160 Chongqing, China*
[3] *School of Mathematical Sciences, Universiti Sains Malaysia, 11800 Gelugor, Penang, Malaysia*
[4] *School of Mathematics and Information Technology, Xingtai University, 054000 Xingtai, Hebei, China*

Abstract. With the rapid development of big data and artificial intelligence technologies, we are facing increasingly complex data and decision-making problems. Solving large-scale multi-objective constrained optimization problems can help to solve many practical engineering and scientific problems. The weighted and Lagrange multiplier methods are considered to be classical and effective methods for dealing with multiple objectives and constraints, but there are some difficulties in solving the processed unconstrained optimization problems. The Newton method is a commonly used method for solving this type of problem, but it requires a high computational complexity. In order to solve these difficulties, we combine four methods such as the Weighted method, Lagrange multiplier method, Newton method and Explicit Group Gauss-Seidel iterative method to propose new Newton Group iterative methods such as 2 and 4-point Explicit Group Gauss-Seidel iterative methods namely as Newton-2EGGS and Newton-4EGGS for solving large-scale multi-objective constrained optimization problems. Also, the convergence analysis of the proposed method is presented. To test the superiority of the proposed method by comparing the computational results, the Newton-4EGGS iteration is more efficient than both Newton-2EGGS iterative method and the Newton-Gauss-Seidel iterative method (Newton-GS), especially in terms of the number of iterations and the computational time.

2020 Mathematics Subject Classifications: 49M15, 65D15, 90C06, 90C29

Key Words and Phrases: Multi-objective constrained optimization, weighted method, Lagrange multiplier method, Newton method, Explicit Group iterative method

*Corresponding author.
DOI: https://doi.org/10.29020/nybg.ejpam.v18i1.5551

Email addresses: pchengmath@163.com (P. Cheng), jumat@ums.edu.my (J. Sulaiman), khadizah@ums.edu.my (K. Ghazali), majidkhanmajaharali@usm.my (M. K. M. Ali), xmmzg@sina.com (M. M. Xu)

1. Introduction

With the development of modern science and technology and the progress of society, many fields are faced with complex optimization problems, which often involve multiple conflicting or interrelated objectives and are subject to multiple constraints at the same time. Such problems are called multi-objective constrained optimization problems (MOCOPs). Especially in the fields of industrial engineering, economic management, environmental protection, travelling salesman problem[37] and transportation [16, 36], the emergence of multi-objective constrained optimization problems is increasingly frequent and the scale is increasing, which puts higher demands on the efficiency and performance of optimization algorithms. Traditional optimization algorithms are often incompetent in dealing with such problems, while simple multi-objective optimization algorithms are prone to fall into local optimum or computational inefficiency when facing large-scale problems [21, 25]. In addition, the existence of constraints further increases the complexity of the problem, and making the algorithm more difficult to search for the optimal solution. Therefore, the study of effective algorithms for solving large-scale multi-objective constrained optimization problems has important theoretical value and practical application significance. It can not only promote the development of optimization algorithm theory, but also provide strong technical support for solving practical engineering problems. The mathematical model of the multi-objective optimization problem (MOOP) with constraints, which is mainly solved in this paper, can be expressed as follows:

$$\begin{aligned} &min\ f(x) = (f_1(x), f_2(x), \cdots, f_n(x))^T, \\ &s.t.\ g_j(x) \leq 0, j = 1, 2, \cdots, p, \\ &\qquad x \in \Omega, \end{aligned} \tag{1}$$

where, $x = (x_1, x_2, \cdots, x_n)^T$, is the n-dimensional decision vector, $\Omega = [x_i^L, x_i^U]^n \subseteq R^n$, denotes the decision space, $F(x) = (f_1(x), f_2(x), \cdots, f_n(x))^T \in R^m$ is the objective function and R^m is the m-dimensional objective space, $g_j(x) \leq 0$ is the j-th inequality constraint, and let X denote the set of all feasible solutions to the problem (1), i.e.$X = \{x \in R^n : g_j(x) \leq 0, j = 1, 2, \cdots, p\}$.

The methods for solving multi-objective optimization problems primarily consist of classical optimization techniques and intelligent optimization algorithms. Classical multi-objective optimization methods encompass the linear weighted method, main objective method, ideal point method [6], and evaluation function method [15]. Intelligent optimization algorithms mainly include Genetic Algorithms (GA) [22], Forbidden Search Algorithm (Tabu Search) [18], Simulated Annealing Algorithm (SA) [3], Ant Colony Algorithm (ACA) [30], Particle Swarm Optimization (PSO) algorithm [9] among others. Intelligent optimization algorithms are a class of approaches that simulate the mechanism of biological evolution in nature to conduct global probabilistic searches. These algorithms do not require the search process to possess properties such as continuity, differentiability, or convexity within the objective function space. They can obtain a set of nondominated solutions by decomposing a multi-objective problem into sub-problems to be solved.

However, these algorithms face challenges related to fitness allocation and diversity control as dimensionality increases. Additionally, due to their population-based nature, they tend to have longer running times and are sensitive to parameter settings. On the other hand, classical multi-objective optimization algorithms leverage characteristics of the solution space such as differentiability and possess more well-established theories with lower computational complexity and faster convergence speeds along with definite termination criteria. Given that the linear weighted method in classical optimization techniques effectively transforms multiple objective functions into a single one through linear weighted, this paper primarily focuses on utilizing this approach for transforming problem (1).

Thus the problem (1) is converted into a solution on a single objective constrained optimization problem (SOOP), where constraints play a crucial role by limiting the range of feasible solutions and thus guiding the search process to find the optimal solution that satisfies the actual problem. There are various methods for dealing with constraints, including penalty function methods, Lagrange multiplier methods, projected gradient methods, feasible direction methods, genetic algorithms, evolutionary strategies, evolutionary planning, and constrained variable scale methods [17, 23, 29, 32]. Among them, the Lagrange method simplifies the optimization problem by introducing Lagrange multipliers, which transform the constraints of the original problem into the form of penalty terms. Also the Lagrange method has higher flexibility in dealing with constraints compared to traditional optimization algorithms. It can deal with a series of complex constraints, including some nonlinear conditions, which has a great advantage in dealing with problems with more complex constraints. Therefore, in this paper, the Lagrange method is used to deal with the constraints of problem (1).

As for the solution of unconstrained optimization problems, the methods include: the most rapid descent method [31], Newton method [20], the proposed Newton method [2], etc., among which Newton method is the classical method for solving unconstrained optimization problems, constructing Newton's iterative direction under the condition that the objective function and constraint function are second-order continuum differentiable, so that the direction is a descending direction for each objective function under the condition of feasibility, and then iteratively The optimal solution is finally obtained. Therefore, it is very widely used, and this paper also mainly considers the Newton method to solve the transformed unconstrained optimization problem.

In addition the Explicit Group iteration originally originated from the exploration of numerical solutions of partial differential equations. Such equations have a wide range of practical applications, including fluid dynamics, thermodynamics, electromagnetism, and other fields. For such problems, it is often necessary to obtain their approximate solutions through numerical computation. The traditional numerical solution methods, such as the finite difference method, and the finite element method, although they can obtain satisfactory solutions in many cases, often face challenges such as large computational volume and poor stability when dealing with large-scale and high-complexity problems. Therefore, researchers have begun to explore new numerical solution methods to address these challenges. The group explicit iteration method is one of them. It was proposed by Evans et al. in 1983 with the idea of group explicit (GE) [8]. Subsequently Saudi et

al. proposed an Explicit Group iteration based on the GE method suitable for robot path planning [34, 35]. Sulaiman et al. [38] have proposed the concept of half-sweep and quarter-sweep iterations respectively for solving their proposed problem. Since the application of this method has been effectively promoted, Lung et al. [24] used the explicit group iteration method to compute the solution of the telegraph equation, and Ghazali et al. combined this method with Newton's method to propose a method for solving unconstrained optimization problems, which achieves computational efficiency and stability enhancement [12]. This brings ideas to our research, and in this paper, we consider extending the Explicit Group iteration into 2-point Explicit Group iteration and 4-point Explicit Group iteration, and combining them with Gauss-Seidel iteration method for solving linear systems for solving transformed unconstrained optimization problems.

Therefore, based on the research of Evans and Ghazali et al., we extend their works for solving the (1) in this paper. The next part of this paper consists of the following main parts: the second part introduces the weighted sum method and the Lagrange multiplier method for transforming the problem (1) into an unconstrained single-objective optimization problem, at the same time, we give the convergence proof; the third part presents the new Newton Group iterative methods proposed in this paper. Numerical experiments and presentation of results are given in the fourth part, and finally some conclusions are discussed in the fifth part.

2. Formulation of weighted Lagrange method and its convergence analysis

In this section, we give the main steps for transforming a multi-objective constrained optimization problem (1) into a single-objective unconstrained optimization problem and the relevant lemmas for establishing the equivalence of the corresponding problems. Based on the equivalence lemmas, we give a convergence row analysis of the new method.

2.1. Weighted Lagrange Multiplier Method

We consider transforming the multi-objective constrained optimization problem into a single-objective constrained optimization problem using the linear weighted sum method with the following expression:

$$\begin{aligned} min\ F(x) &= w_1 f_1(x) + w_2 f_2(x) + \cdots + w_n f_n(x), \\ s.t.\ g_j(x) &\leq 0, j = 1, 2, \cdots, p, \\ x &\in \Omega. \end{aligned} \tag{2}$$

where, $w_i \leq 0 (i = 1, \ldots, m)$ is the weight and $\sum_{i=1}^{m} = 1$.

Definition 1. *[26] It is known that $X \subseteq R^n$ is a feasible solution set for the model (1), and obviously for (2), if $X \subseteq X$ and there exists no $x \in X$ such that*

$$F(x) \leq F(x^*)$$

then $x^* \in X$ is said to be an optimal solution to the constrained optimization problem (2).

The following lemma establishes the equivalence between the linear weighted problem (2) and the solution of the original multi-objective optimization problem (1).

Lemma 1. *[26] The optimal solution of the linear weighted problem (2) is a weak pareto optimal solution of the multi-objective constrained optimization problem (1)). We denote the weakly efficient set of (1) by X^*.*

Lemma 2. *[26] Let the multi-objective constrained optimization problem (1) be convex, if x^* is its pareto optimal solution, then there exists a weight vector $w(w_i > 0, i = 1, \ldots, m, \sum_{i=1}^{m} = 1$ such that x^* is the optimal solution of the weighted optimization problem (2).*

In the following we transform (1) into the following unconstrained optimization problem by introducing a new variable λ, combining the inequality constraints as well as the objective function $F(x)$ using the Lagrange multiplier method, i.e.:

$$L(x, \mu) = F(x) + \sum_{j=1}^{m} \lambda_j g_j(x). \tag{3}$$

where, $F(x) = w_1 f_1(x) + w_2 f_2(x) + \cdots + w_n f_n(x)$, $\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_n)^T$ is the Lagrange multiplier.

Lemma 3. *[1]If x^* is an optimal solution of the problem (3) and satisfies the following KKT conditions, then x^* is an optimal solution of the problem (2). We denote $\widehat{X_n^*}$as the set of optimal solutions for (3).*

$$\begin{cases} \bigtriangledown_x L(x^*, \lambda) = \bigtriangledown f(x^*) + \sum_{j=1}^{m} \lambda_j g_j(x), \\ \lambda_j g_j(x)^* = 0, \ j = 1, 2, \cdots, p, \\ \lambda_j \geq 0. \end{cases}$$

2.2. Convergence analysis of the proposed methods

In the following theorem, we prove that under certain conditions, the optimal solution of the linear weighted problem (2) can be approximated by a convergent subsequence of the sequence of optimal solutions of the unconstrained single-objective optimization problem (3).

Theorem 1. *Let x_n^* be an optimal solution of (3), $x_{n_k}^*$ be a convergent subsequence of x_n^*, and $\lim_{k\to\infty} x_{n_k}^* = x^* \in X^*$, then the limit point of $x_{n_k}^*$ is an optimal solution of the linearly weighted problem (2).*

Proof. Since $x_{n_k}^* \in \widehat{x_{n_k}^*}$ then for all $\overline{x} \in X^*$, we have

$$\sum_{i=1}^{m} w_i f_i(x_{n_k}^*) + \sum_{j=1}^{p} \lambda_j g_j(x_{n_k}^*) \leq \sum_{i=1}^{m} f_i(\overline{x}) + \sum_{j=1}^{p} \lambda_j g_j(\overline{x}), k = 1, 2, \cdots. \tag{4}$$

We discuss this in the following two scenarios.

(i) If $g_j(x) = 0$, it is clear that there are

$$\lim_{k\to\infty} \sum_{j=1}^{p} \lambda_j g_j(x^*_{n_k}) = 0, \tag{5}$$

$$\lim_{k\to\infty} \sum_{j=1}^{p} \lambda_j g_j(\overline{x}) = 0, \tag{6}$$

Taking the limit of equation (4) and combining the Equations (5) and (6)

$$\lim_{k\to\infty} \sum_{i=1}^{m} w_i f_i(x^*_{n_k}) \leq \lim_{k\to\infty} \sum_{i=1}^{m} w_i f_i(\overline{x}),$$

i.e.

$$\sum_{i=1}^{m} w_i f_i(x^*) \leq \sum_{i=1}^{m} w_i f_i(\overline{x}),\ for\ all\ \overline{x} \in X^*.$$

So x^* is the optimal solution of (2).

(ii) If $g_j(x) < 0$, then there are

$$\lim_{k\to\infty} \sum_{j=1}^{p} \lambda_j g_j(\overline{x}) \leq 0, \tag{7}$$

Since x^*_n is an optimal solution of (3), and $x^*_{n_k}$ is a convergent subsequence of x^*_n, there are

$$\lim_{k\to\infty} \sum_{j=1}^{p} \lambda_j g_j(x^*_{n_k}) = 0, \tag{8}$$

Taking the limit of expression (3) and combining equations (7) and (8) yields

$$\lim_{k\to\infty} \sum_{i=1}^{m} w_i f_i(x^*_{n_k}) \leq \lim_{k\to\infty} \sum_{i=1}^{m} w_i f_i(\overline{x}) + \lim_{k\to\infty} \sum_{j=1}^{p} \lambda_j g_j(\overline{x}),$$

i.e.

$$\sum_{i=1}^{m} w_i f_i(x^*) \leq \sum_{i=1}^{m} w_i f_i(\overline{x}),$$

So the theorem is proved.

From Theorem 1, Lemma 1, and Lemma 2, it can be obtained that the convergent subsequence of the sequence of optimal solutions of the unconstrained single-objective nonlinear programming problem (3) can approximate the weak pareto optimal solution of the original multi-objective constrained optimization problem.

3. Newton-2EGGS and Newton-4EGGS

For the transformed problem (3), which is mainly a problem of unconstrained optimization, combined with the advantages of Newton's method of solving the problem they will be used in the next solution, the iterative formula of Newton's method can be expressed as follows [4, 5, 13]:

$$\begin{aligned} x^{(k+1)} &= x^{(k)} - \alpha_k d^{(k)}, \\ d^{(k)} &= G_k^{-1} g_k, \end{aligned} \tag{9}$$

where, $d^{(k)}$ is the search direction, the step factor $\alpha_k = 1, G_k = \bigtriangledown^2 L(x^{(k)}), g_k = \bigtriangledown L(x^{(k)})$, and G_k^{-1} is the inverse matrix of the Hessian matrix $G_k = \bigtriangledown^2 L(x^{(k)})$. Also, we know that when G_k is positive definite, its inverse matrix must exist and be positive definite, so the Newtonian direction will be a descent direction since it satisfies,

$$g_k^T d^{(k)} = -g_k^T G_k^{-1} g_k < 0.$$

The Hessian matrix G_k is solved [14] as follows:

$$G_k = \begin{bmatrix} \partial^2 L/\partial x_1^2 & \partial^2 L/(\partial x_1 \partial x_2) & \cdots & \partial^2 L/(\partial x_1 \partial x_n) & \cdots & \partial^2 L/(\partial x_1 \partial \lambda_p) \\ \partial^2 L/(\partial x_2 \partial x_1) & \partial^2 L/\partial x_2^2 & \cdots & \partial^2 L/(\partial x_2 \partial x_n) & \cdots & \partial^2 L/(\partial x_2 \partial \lambda_p) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \partial^2 L/(\partial x_n \partial x_1) & \partial^2 L/(\partial x_n \partial x_2) & \cdots & \partial^2 L/\partial x_n^2 & \cdots & \partial^2 L/(\partial x_n \partial \lambda_p) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \partial^2 L/(\partial \lambda_p \partial x_1) & \partial^2 L/(\partial \lambda_p \partial x_2) & \cdots & \partial^2 L/(\partial \lambda_p \partial x_n) & \cdots & \partial^2 L/\partial \lambda_p^2 \end{bmatrix},$$

The Newton equation (9) mentioned above, we can convert it into the following linear system that

$$G_k d_k = g_k. \tag{10}$$

After the k-th iteration, the specific coefficient matrix of the system of linear equations (10) can be obtained by bringing x^k into the Hessian matrix G_k. This coefficient matrix we denote by the symbol A. Therefore, we write the system of equations (10) in the following simple form:

$$Ad = b, \tag{11}$$

where,

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} & a_{1,n+1} & \cdots & a_{1,s} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} & a_{2,n+1} & \cdots & a_{2,s} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} & a_{n,n+1} & \cdots & a_{n,s} \\ a_{n+1,1} & a_{n+1,2} & \cdots & a_{n+1,n} & a_{n+1,n+1} & \cdots & a_{n+1,s} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{s,1} & a_{s,2} & \cdots & a_{s,n} & a_{s,n+1} & \cdots & a_{s,s} \end{bmatrix}, d = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{s-1} \\ d_s \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{s-1} \\ b_s \end{bmatrix},$$

$a_{1,1}, \cdots, a_{s,s}, d_1, \cdots, d_s, b_1, \cdots, b_s \in R$ and $s = n + p$, the diagonal elements contain at least p zeros. Combined with the Gauss-Seidel iterative method for solving the linear system, we propose the 2-point Explicit Group Gauss-Seidel (2EGGS) iterative method and the 4-point Explicit Group Gauss-Seidel (4EGGS) iterative method iteration. The computational principles of the two methods are given below.

3.1. Formulation of Newton-2EGGS

To present our iterative method, we decompose the matrix, A as $A = D_1 - L_1 - U_1$, where D_1, L_1, U_1 are of the following form:

$$D_1 = \begin{bmatrix} a_{1,1} & a_{1,2} & 0 & 0 & 0 & \cdots & 0 & 0 \\ a_{2,1} & a_{2,2} & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & a_{3,3} & a_{3,4} & 0 & \cdots & 0 & 0 \\ 0 & 0 & a_{4,3} & a_{4,4} & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & a_{s-1,s-1} & a_{s-1,s} \\ 0 & 0 & 0 & 0 & 0 & \cdots & a_{s,s-1} & a_{s,s} \end{bmatrix},$$

$$L_1 = \begin{bmatrix} 0 & 0 & a_{1,3} & a_{1,4} & a_{1,5} & \cdots & a_{1,s-1} & a_{1,s} \\ 0 & 0 & a_{2,3} & a_{2,4} & a_{2,5} & \cdots & a_{2,s-1} & a_{2,s} \\ 0 & 0 & 0 & 0 & a_{3,5} & \cdots & a_{3,s-1} & a_{3,s} \\ 0 & 0 & 0 & 0 & a_{4,5} & \cdots & a_{4,s-1} & a_{4,s} \\ 0 & 0 & 0 & 0 & \ddots & \ddots & a_{5,s-1} & a_{5,s} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix},$$

$$U_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ a_{3,1} & a_{3,2} & 0 & 0 & 0 & \cdots & 0 & 0 \\ a_{4,1} & a_{4,2} & 0 & 0 & 0 & \cdots & 0 & 0 \\ a_{5,1} & a_{5,2} & a_{5,3} & a_{5,4} & \ddots & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ a_{s-1,1} & a_{s-1,2} & a_{s-1,3} & a_{s-1,4} & a_{s-1,5} & \cdots & 0 & 0 \\ a_{s,1} & a_{s,2} & a_{s,3} & a_{s,4} & a_{s,5} & \cdots & 0 & 0 \end{bmatrix},$$

If the last module of the block diagonal matrix, D_1 is less than 4 elements, the actual division of the elements will prevail. So the 2-point Explicit Group Gauss-Seidel iteration for solving the linear system (11) is as follows:

$$\begin{cases} d^{(0)}(\textit{initial vector}), \\ d^{(k+1)} = (D_1 - L_1)^{-1} U_1 d^{(k)} + (D_1 - L_1)^{-1} b, \ k = 0, 1, \cdots. \end{cases} \tag{12}$$

The formula for calculating its components is given below and denoted as

$$d^{(k)} = \left(d_1^{(k)}, d_2^{(k)}, \cdots, d_i^{(k)}, \cdots, d_s^{(k)}\right)^T. \tag{13}$$

From equation (12) we have

$$(D_1 - L_1)d^{(k+1)} = U_1 d^{(k)} + b,$$

$$D_1 d^{(k+1)} = L_1 d^{(k)} + U_1 d^{(k)} + b,$$

i.e.

$$a_{i,i}d_i^{(k+1)} + a_{i,i+1}d_{i+1}^{(k+1)} = -\sum_{j=1}^{i-1} a_{i,j}d_j^{(k+1)} - \sum_{j=i+2}^{s} a_{i,j}d_j^{(k)} + b_i, i = 1, 2, \cdots, s. \tag{14}$$

$$a_{i+1,i}d_i^{(k+1)} + a_{i+1,i+1}d_{i+1}^{(k+1)} = -\sum_{j=1}^{i-1} a_{i+1,j}d_j^{(k+1)} - \sum_{j=i+2}^{s} a_{i+1,j}d_j^{(k)} + b_{i+1}, i = 1, 2, \cdots, s. \tag{15}$$

Solving equations (14) and (15) yields the following formula for solving the 2-point Explicit Group Gauss-Seidel iterative method component of the linear system (11).

$$\begin{cases} d^{(0)} = \left(d_1^{(0)}, d_2^{(0)}, \cdots, d_i^{(0)}, \cdots, d_s^{(0)}\right)^T, \\ d_i^{(k+1)} = \frac{a_{i+1,i+1}Q_1 - a_{i,i+1}Q_2}{\omega}, \\ d_{i+1}^{(k+1)} = \frac{-a_{i+1,i}Q_1 + a_{i,i}Q_2}{\omega}, \\ Q_1 = -\sum_{j=1}^{i-1} a_{i,j}d_j^{(k+1)} - \sum_{j=i+2}^{s} a_{i,j}d_j^{(k)} + b_i, \\ Q_2 = -\sum_{j=1}^{i-1} a_{i+1,j}d_j^{(k+1)} - \sum_{j=i+2}^{s} a_{i+1,j}d_j^{(k)} + b_{i+1}, \\ \omega = a_{i,i}a_{i+1,i+1} - a_{i,i+1}a_{i+1,i}, \\ i = 1, 2, \cdots, s; k = 0, 1, \cdots. \end{cases} \tag{16}$$

So we get the iteration steps of the Newton-2EGGS iterative method as follows:

Algorithm 1: Newton-2EGGS Scheme

I. Calculate the expression of the objective function by using equation (3).

II. Give the initial value x_0 and the accuracy threshold $\varepsilon_1 = 10^{-5}$, $\varepsilon_2 = 10^{-10}$ and let $k := 0$.

III. Calculate matrix G_k and gradient g_k If $\|g_k\| < \varepsilon_1$, that is, the extreme point is reached, the iteration is stopped, otherwise, go to IV.

IV. Calculate the following

a. Calculate the matrix D_1,L_1 and U_1.

b. Calculate the search direction by using equation (16).

c. Calculate the convergence condition, if $\|d^{(k+1)} - d^{(k)}\| < \varepsilon_2$, then go to step V, otherwise go to step IV(b).

V. Calculate the new iteration point as $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$.

VI. Let $k = k + 1$, go to step III.

3.2. Formulation of Newton-4EGGS

Similar to the 2EGGS iterative method, we decompose the matrix, A as $A = D_2 - L_2 - U_2$, where

$$D_2 = \begin{bmatrix} V_1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & V_2 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & V_i & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & V_q \end{bmatrix}, L_2 = - \begin{bmatrix} 0 & 0 & \cdots & 0 & \cdots & 0 \\ W_{2,1} & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ W_{i,1} & W_{i,2} & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ W_{q,1} & W_{q,2} & \cdots & W_{q,i} & \cdots & 0 \end{bmatrix},$$

$$U_2 = - \begin{bmatrix} 0 & W_{1,2} & \cdots & W_{1,i} & \cdots & W_{1,q} \\ 0 & 0 & \cdots & W_{2,i} & \cdots & W_{2,q} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & W_{i,q} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 0 \end{bmatrix},$$

$$V_{i,i} = \begin{bmatrix} a_{i,i} & a_{i,i+1} & a_{i,i+2} & a_{i,i+3} \\ a_{i+1,i} & a_{i+1,i+1} & a_{i+1,i+2} & a_{i+1,i+3} \\ a_{i+2,i} & a_{i+2,i+1} & a_{i+2,i+2} & a_{i+2,i+3} \\ a_{i+3,i} & a_{i+3,i+1} & a_{i+3,i+2} & a_{i+3,i+3} \end{bmatrix}, W_{i,j} = \begin{bmatrix} a_{i,s-3} & a_{i,s-2} & a_{i,s-1} & a_{i,s} \\ a_{i+1,s-3} & a_{i+1,s-2} & a_{i+1,s-1} & a_{i+1,s} \\ a_{i+2,s-3} & a_{i+2,s-2} & a_{i+2,s-1} & a_{i+2,s} \\ a_{i+3,s-3} & a_{i+3,s-2} & a_{i+3,s-1} & a_{i+3,s} \end{bmatrix}.$$

If there are less than 16 elements inside the final modular matrix V_q, then the actual possession will prevail. So the 4-point Explicit Group Gauss-Seidel iterative method for solving the linear system (11) is as follows.

$$\begin{cases} d^{(0)}(initial\ vector), \\ d^{(k+1)} = (D_2 - L_2)^{-1} U_2 d^{(k)} + (D_2 - L_2)^{-1} b,\ k = 0, 1, \cdots. \end{cases} \tag{17}$$

The formula for calculating its components is (13). From equation (17) we have

$$D_2 d^{(k+1)} = L_2 d^{(k)} + U_2 d^{(k)} + b,$$

i.e.

$$\sum_{j=i}^{i+3} a_{i,j} d_j^{(k+1)} = -\sum_{j=1}^{i-1} a_{i,j} d_j^{(k+1)} - \sum_{j=i+4}^{s} a_{i,j} d_j^{(k+1)} + b_i, \tag{18}$$

$$\sum_{j=i}^{i+3} a_{i+1,j} d_j^{(k+1)} = -\sum_{j=1}^{i-1} a_{i+1,j} d_j^{(k+1)} - \sum_{j=i+4}^{s} a_{i+1,j} d_j^{(k+1)} + b_{i+1}, \tag{19}$$

$$\sum_{j=i}^{i+3} a_{i+2,j} d_j^{(k+1)} = -\sum_{j=1}^{i-1} a_{i+2,j} d_j^{(k+1)} - \sum_{j=i+4}^{s} a_{i+2,j} d_j^{(k+1)} + b_{i+2}, \tag{20}$$

$$\sum_{j=i}^{i+3} a_{i+3,j} d_j^{(k+1)} = -\sum_{j=1}^{i-1} a_{i+3,j} d_j^{(k+1)} - \sum_{j=i+4}^{s} a_{i+3,j} d_j^{(k+1)} + b_{i+3}, \tag{21}$$

Solving equations (18)-(21) yields

$$\begin{cases} d^{(0)} = \left(d_1^{(0)}, d_2^{(0)}, \cdots, d_i^{(0)}, \cdots, d_s^{(0)}\right)^T, \\ d_i^{(k+1)} = \frac{B_1}{|V_i|}, \\ d_{i+1}^{(k+1)} = \frac{B_2}{|V_i|}, \\ d_{i+2}^{(k+1)} = \frac{B_3}{|V_i|}, \\ d_{i+3}^{(k+1)} = \frac{B_4}{|V_i|}, \\ i = 1, 2, \cdots, s; k = 0, 1, \cdots. \end{cases} \tag{22}$$

where,

$$B_1 = \begin{vmatrix} Q_i & a_{i,i+1} & a_{i,i+2} & a_{i,i+3} \\ Q_{i+1} & a_{i+1,i+1} & a_{i+1,i+2} & a_{i+1,i+3} \\ Q_{i+2} & a_{i+2,i+1} & a_{i+2,i+2} & a_{i+2,i+3} \\ Q_{i+3} & a_{i+3,i+1} & a_{i+3,i+2} & a_{i+3,i+3} \end{vmatrix}, B_2 = \begin{vmatrix} a_{i,i} & Q_i & a_{i,i+2} & a_{i,i+3} \\ a_{i+1,i} & Q_{i+1} & a_{i+1,i+2} & a_{i+1,i+3} \\ a_{i+2,i} & Q_{i+2} & a_{i+2,i+2} & a_{i+2,i+3} \\ a_{i+3,i} & Q_{i+3} & a_{i+3,i+2} & a_{i+3,i+3} \end{vmatrix},$$

$$B_3 = \begin{vmatrix} a_{i,i} & a_{i,i+1} & Q_i & a_{i,i+3} \\ a_{i+1,i} & a_{i+1,i+1} & Q_{i+1} & a_{i+1,i+3} \\ a_{i+2,i} & a_{i+2,i+1} & Q_{i+2} & a_{i+2,i+3} \\ a_{i+3,i} & a_{i+3,i+1} & Q_{i+3} & a_{i+3,i+3} \end{vmatrix}, B_4 = \begin{vmatrix} a_{i,i} & a_{i,i+1} & a_{i,i+2} & Q_i \\ a_{i+1,i} & a_{i+1,i+1} & a_{i+1,i+2} & Q_{i+1} \\ a_{i+2,i} & a_{i+2,i+1} & a_{i+2,i+2} & Q_{i+2} \\ a_{i+3,i} & a_{i+3,i+1} & a_{i+3,i+2} & Q_{i+3} \end{vmatrix},$$

i.e.

$$Q_i = -\sum_{j=1}^{i-1} a_{i,j} d_j^{(k+1)} - \sum_{j=i+4}^{s} a_{i,j} d_j^{(k+1)} + b_i,$$

$$Q_{i+1} = -\sum_{j=1}^{i-1} a_{i+1,j} d_j^{(k+1)} - \sum_{j=i+4}^{s} a_{i+1,j} d_j^{(k+1)} + b_{i+1},$$

$$Q_{i+2} = -\sum_{j=1}^{i-1} a_{i+2,j} d_j^{(k+1)} - \sum_{j=i+4}^{s} a_{i+2,j} d_j^{(k+1)} + b_{i+2},$$

$$Q_{i+3} = -\sum_{j=1}^{i-1} a_{i+3,j} d_j^{(k+1)} - \sum_{j=i+4}^{s} a_{i+3,j} d_j^{(k+1)} + b_{i+3},$$

Algorithm 2: Newton-4EGGS Scheme

I. Calculate the expression of the objective function by using equation (3).

II. Give the initial value x_0 and the accuracy threshold $\varepsilon_1 = 10^{-5}$, $\varepsilon_2 = 10^{-10}$ and let $k := 0$.

III. Calculate matrix G_k and gradient g_k If $\|g_k\| < \varepsilon_1$, that is, the extreme point is reached, the iteration is stopped, otherwise, go to IV.

IV. Calculate the following

   a. Calculate the matrix D_2,L_2,U_2,V_i and $W_{i,q}$.

   b. Calculate the search direction by using equation (22).

   c. Calculate the convergence condition, if $\|d^{(k+1)} - d^{(k)}\| < \varepsilon_2$, then go to step V, otherwise go to step IV(b).

V. Calculate the new iteration point as $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$.

VI. Let $k = k + 1$, go to step III.

4. Numerical experiments

In this section, numerical experiments are performed to verify the effectiveness and superiority of the algorithm, the test functions are mainly taken from references [10, 11, 19]. They are all chosen based on the fact that the Hessian matrix is a full matrix. These test cases were all computed at $n = (10, 50, 100, 150, 150)$ for a total of five different orders of Hessian matrices. Thus, this corresponds to a total of 20 test cases provided. All calculations and drawings in this paper are done by using MATLAB software, version: MATLAB2020a. Below we give the specific expressions for the four cases.

Example 1[10]

$$min\ f_1(x) = x_1 + g,$$

$$
\begin{aligned}
min\ f_2(x) &= 1 - x_1^2 + g, \\
g &= \sum_{i=2}^{n} (x_i - sin(0.5\pi x_1))^2, \\
s.t.\ c(x) &= sin(a\pi x_1) - b \geq 0, x \in [0,1],
\end{aligned}
$$

where $a > 0, b \in [-1, 1]$. As an example, $a = 1, b = 0.5$ are set here.

Firstly, the Example1 is transformed into a single objective optimization problem using weighted as follows:

$$
\begin{aligned}
min\ F(x) &= w_1 f_1(x) + w_2 f_2(x), \\
s.t.\ c(x) &= sin(a\pi x_1) - b \geq 0, \\
x &\in \Omega.
\end{aligned}
$$

where, w is the weight coefficient, and $\sum_{i=1}^{2} w_i = 1$. The Lagrange multiplier method is then used to transform Example 1 into an unconstrained optimization problem as follows:

$$
min\ F(x) = w_1 f_1(x) + w_2 f_2(x) + \lambda(-c(x)),
$$

where, $\lambda > 0$ is the Lagrange multiplier. This example has an external Newton method iteration number of 3 for dimensions 10, 50, and 4 for dimensions 100, 150, and 250 for an initial point of $(1, 1, ..., 1)$, and the objective function values are all approximated as $f_1(x) = 0.8333, f_2(x) = 0.3056$. Figure 1 illustrates a three-dimensional stereogram of the weighted Lagrange function $F_1(x)$ for dimension $n = 2$. The following examples are converted using a similar method as in Example 1.

Example 2[11]

$$
\begin{aligned}
min\ f_1(x) &= x_1 + g_1, \\
min\ f_2(x) &= 1 - x_1^2 + g_2, \\
g_1 &= \sum_{i \in I_1} \left[x_i - sin(0.5\pi x_1)\right]^2, \\
g_2 &= \sum_{i \in I_2} \left[x_i - cos(0.5\pi x_1)\right]^2, \\
s.t.\ c(x) &= sin(a\pi x_1) - b \geq 0, \\
I_1 &= \{i | i\ is\ odd\ and\ 2\ \leq i \leq n\}, \\
I_2 &= \{i | i\ is\ even\ and\ 2\ \leq i \leq n\}, \\
x &\in [0,1],
\end{aligned}
$$

where $a > 0, b \in [-1, 1]$. As an example, $a = 2$, $b = 0.1$ are set here.

The weighted Lagrange function for Example 2 is

$$
min\ F_2(x) = w_1(x_1 + g_1) + w_2(1 - x_1^2 + g_2) + \lambda(-sin(2\pi x_1) - 0.1).
$$

This example has an external Newton method iteration number of 2 for dimensions 10, 50, and 3 for dimensions 100, 150, and 250 for an initial point of (0.5,0.4,0.5, 0.4,. . . ,0.5,0.4), and the objective function values are all approximated as $f_1(x) = 0.4841$, $f_2(x) = 0.7657$. Figure 2 illustrates a three-dimensional stereogram of the weighted Lagrange function $F_2(x)$ for dimension $n = 2$.

Example 3[11]

$$\begin{gathered}
min\ f_1(x) = x_1 + g_1, \\
min\ f_2(x) = 1 - \sqrt{x_1} + g_2, \\
g_1 = \sum_{i \in I_1} \left[x_i - sin(0.5\pi x_1)\right]^2, \\
g_2 = \sum_{i \in I_2} \left[x_i - cos(0.5\pi x_1)\right]^2, \\
s.t.\ c(x) = sin(a\pi x_1) - b \geq 0, \\
I_1 = \{i | i\ is\ odd\ and\ 2\ \leq i \leq n\}, \\
I_2 = \{i | i\ is\ even\ and\ 2\ \leq i \leq n\}, \\
x \in [0, 1],
\end{gathered}$$

where $a > 0, b \in [-1, 1]$. As an example, $a = 2$, $b = 0.2$ are set here.

The weighted Lagrange function for Example 2 is

$$min\ F_3(x) = w_1(x_1 + g_1) + w_2(1 - \sqrt{x_1} + g_2) + \lambda(-sin(2\pi x_1) - 0.2).$$

Example 3 has an external Newton method iteration number of 2 for dimensions 10 and 3 for dimensions 50, 100, 150, and 250 for an initial point of (0.5,0.8,0.5,0.8,. . . , 0.5,0.8), and the objective function values are all approximated as $f_1(x) = 0.4680$,$f_2(x) = 0.3159$. Figure 3 illustrates a three-dimensional stereogram of the weighted Lagrange function $F_3(x)$ for dimension n=2.

Example 4[19]

$$\begin{gathered}
min\ f_1(x) = (1+g)cos(\frac{x_1\pi}{2}) \cdots cos(\frac{x_{M-1}\pi}{2}), \\
min\ f_1(x) = (1+g)cos(\frac{x_1\pi}{2}) \cdots sin(\frac{x_{M-1}\pi}{2}), \\
\vdots \\
min\ f_M(x) = (1+g)sin(\frac{x_1\pi}{2}), \\
g = \sum_{i=M}^{n} (x_i - 0.5)^2, \\
s.t.\ c(x) = \sum_{i=1}^{M} (f_i(x) - \sigma)^2 - r^2 \geq 0, x \in [0, 1],
\end{gathered}$$

where $M = 2$, $\sigma = \frac{1}{M}\sum_{i=1}^{M} f_i(x)$, $r = 0.225$. The weighted Lagrange function for Example 2 is

$$min\ F_2(x) = w_1 f_1(x) + w_2 f_2(x) + \lambda\left\{-\left[\left(f_1(x) - \frac{1}{2}(f_1(x) + f_2(x))\right)^2 + \left(f_2(x) - \frac{1}{2}(f_1(x) + f_2(x))\right)^2 - 0.225^2\right]\right\},$$

where, $f_1(x) = (1+g)cos(\frac{x_1\pi}{2})$, $f_2(x) = (1+g)sin(\frac{x_1\pi}{2})$, $g(x) = (x_2 - 0.5)^2$.

This example has 4 external Newton's method iterations for dimension 10, 5 external for dimension 50, 6 external for dimensions 100 and 150, and 8 external for dimension 250 with initial points (0.2,0.4,0.3,0.4,...,0.3,0.4), and the objective function values are all approximated by $f_1(x) = 0.9579$, $f_2(x) = 0.2871$. Figure 4 illustrates a three-dimensional stereogram of the weighted Lagrangian function $F_4(x)$ for dimension $n = 2$. The three-dimensional stereograms of Figures 1-Figure 4 show the location of the minimum value of the weighted Lagrange function in the feasible domain for the four examples. The explanation of the symbols is first given by Table 1.

Table 1: Description of symbols used in the result display.

Notation	Description
n	Test number
N	Number of variables
GS	Gauss-Seidel method
$2EGGS$	2-point Explicit Group GS
$4EGGS$	4-point Explicit Group GS
NGS	Newton-GS method
$2NEGS$	Newton-2EGGS method
$4NEGS$	Newton-4EGGS method
TM	Computational time(Unit: Second)
NOI	Internal iterations
$L2-g$	The $L2$ Norm of the gradient g at the end of the computation

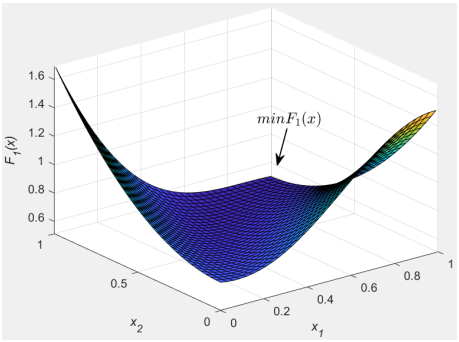

Figure 1: 3D diagram for Example 1.

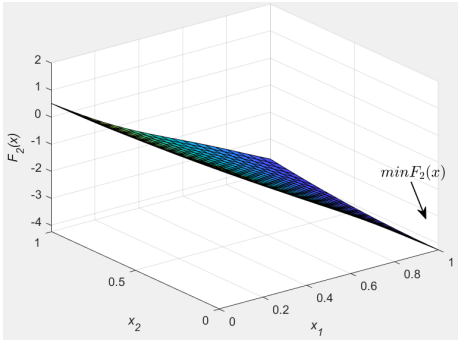

Figure 2: 3D diagram for Example 2.

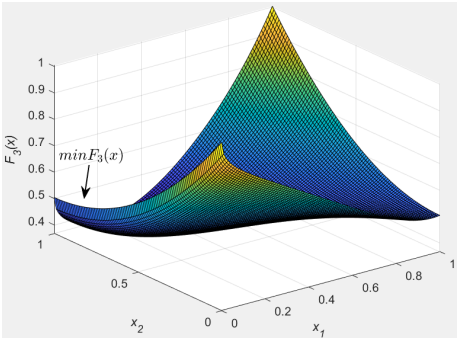

Figure 3: 3D diagram for Example 3.

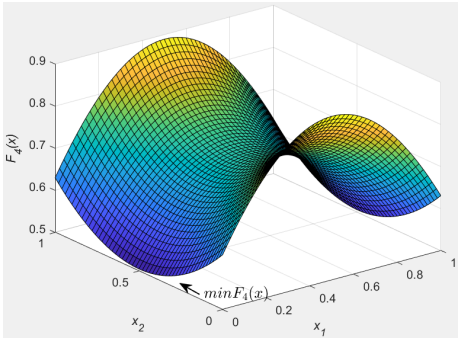

Figure 4: 3D diagram for Example 4.

Table 2: Comparison of the results of the above three different methods for Example 1.

P	method	N 10	50	100	150	250
NOI	NGS	105	903	3788	8111	21055
	$2NEGGS$	100	893	3761	8071	20992
	$4NEGGS$	91	872	3709	7994	20867
TM	NGS	0.8978	10.2485	79.5447	291.4592	1785.2087
	$2NEGGS$	0.8658	9.3863	75.8597	290.8716	1757.3603
	$4NEGGS$	0.8136	9.1721	74.8187	280.9366	1718.8262
L2-g	NGS	8.9251E-07	5.3210E-06	5.0076E-09	9.2859E-09	1.9899E-08
	$2NEGGS$	8.9251E-07	5.3210E-06	5.0072E-09	9.2687E-09	1.9836E-08
	$4NEGGS$	8.9251E-07	5.3210E-06	4.8564E-09	9.1065E-09	1.9702E-08

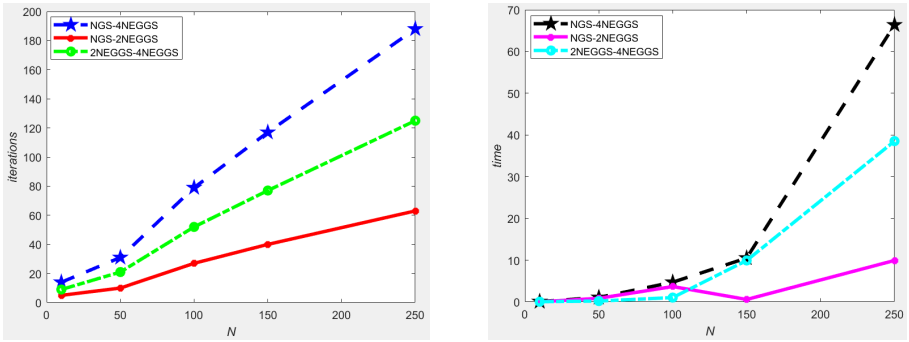

Figure 5: iterations and time (seconds) difference in different dimensions for Example 1.

5. Comparison Results

We show specific computational results for four different example problems in Tables 2-5, which indicate that our proposed new explicit group iteration method successfully solves large-scale multi-objective constrained optimization problems. Since the same pareto optimal solution is obtained using different iteration methods for the same latitude case in each example problem. So we only show the number of iterations inside the algorithm, the

Table 3: Comparison of the results of the above three different methods for Example 2.

P	method	N 10	50	100	150	250
NOI	NGS	272	2679	7140	11425	23228
	$2NEGGS$	262	2577	7110	11398	23203
	$4NEGGS$	260	2534	7040	11333	23129
TM	NGS	1.4114	20.1857	118.9396	352.2824	1790.0120
	$2NEGGS$	1.3655	19.5404	115.9887	349.8387	1779.2555
	$4NEGGS$	1.3561	19.2925	114.1318	345.3602	1763.0075
L2-g	NGS	9.5525E-07	5.3173E-06	7.2858E-09	1.8530E-08	4.4736E-08
	$2NEGGS$	9.5525E-07	5.3173E-06	7.3544E-09	1.8463E-08	4.4585E-08
	$4NEGGS$	9.5525E-07	5.3174E-06	7.5201E-09	1.8393E-08	4.4297E-08

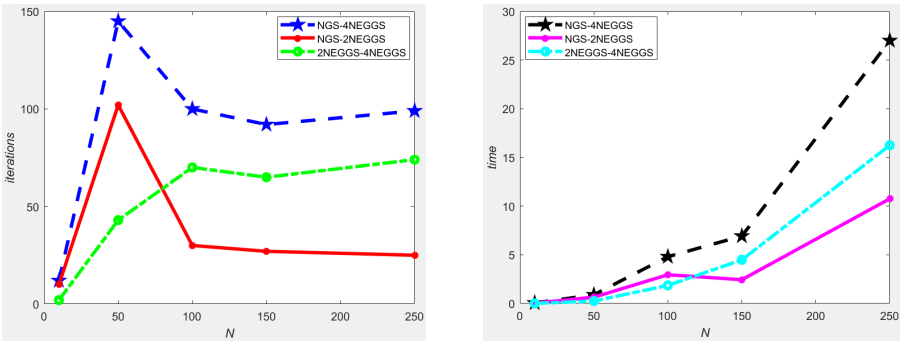

Figure 6: iterations and time (seconds) difference in different dimensions for Example 2.

Table 4: Comparison of the results of the above three different methods for Example 3.

P	method	N 10	50	100	150	250
NOI	NGS	323	3615	8069	12715	26828
	$2NEGGS$	305	3306	8015	12676	26755
	$4NEGGS$	280	3258	7914	12594	26587
TM	NGS	1.6175	27.9083	130.9371	389.1743	2031.1232
	$2NEGGS$	1.5249	25.3495	128.6421	382.9091	2029.2933
	$4NEGGS$	1.4355	25.0829	125.1750	380.2002	1995.1464
L2-g	NGS	6.2168E-06	3.2971E-09	8.9170E-09	1.3044E-08	4.0726E-08
	$2NEGGS$	6.2168E-06	4.9737E-09	8.7289E-09	1.3035E-08	4.0622E-08
	$4NEGGS$	6.2168E-06	4.4817E-09	8.3910E-09	1.3006E-08	4.0363E-08

total computation time and the Norm of Function Gradient at Termination of Calculation inside the table. For comparison purposes we keep the decimals in Tables 2-5 to 4 decimal places. In addition, Figure 5-8 shows the difference between the number of iterations and the computation time between different algorithms for each example problem in different

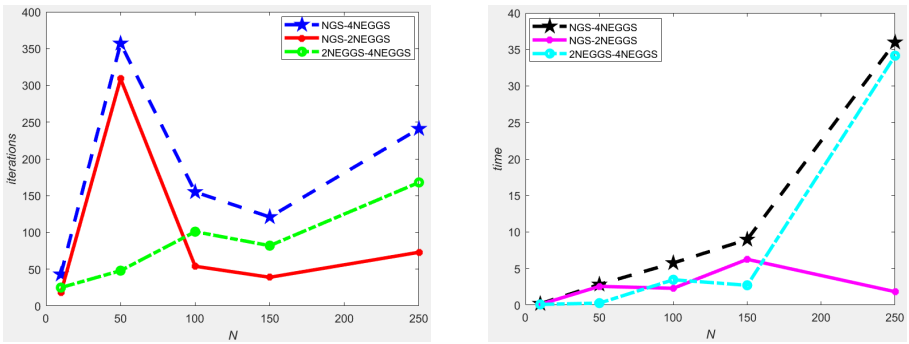

Figure 7: iterations and time (seconds) difference in different dimensions for Example 3.

Table 5: Comparison of the results of the above three different methods for Example 4.

		N				
P	method	10	50	100	150	250
NOI	NGS	91	177	1281	7269	114176
	$2NEGGS$	91	172	1167	6234	86083
	$4NEGGS$	90	157	913	4267	48913
TM	NGS	1.1596	11.1179	69.6460	352.2824	8487.6144
	$2NEGGS$	1.1349	10.1160	64.7557	339.1660	6584.3840
	$4NEGGS$	1.0560	10.0277	62.9561	305.9793	4115.1276
L2-g	NGS	1.7766E-07	4.3005E-08	2.4160E-06	8.9724E-06	1.8916E-06
	$2NEGGS$	1.7766E-07	4.3005E-08	2.4160E-06	8.9724E-06	1.8916E-06
	$4NEGGS$	1.7766E-07	4.3005E-08	2.4160E-06	8.9724E-06	1.8916E-06

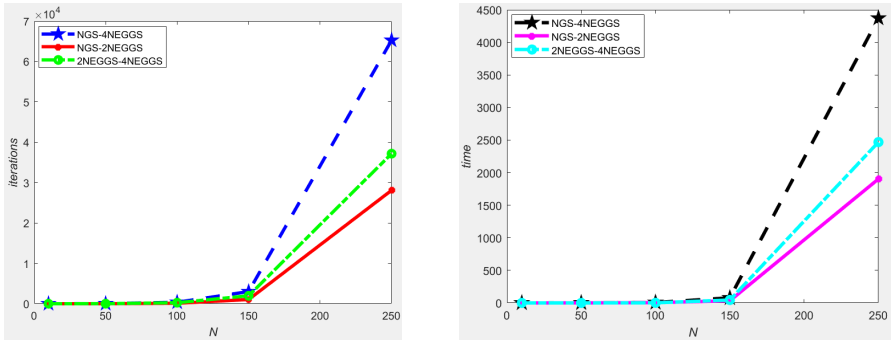

Figure 8: iterations and time (seconds) difference in different dimensions for Example 4.

dimensions (NGS-4NEGGS denotes the result of the Newton-GS method minus the computation result of Newton-4EGGS), so the clear advantage of Newton-4EGGS can be seen from both Tables 2-5 and Figures 5-8.

From the results in Table 6, it can be seen that the number of iterations using the Newton-2EGGS method is less than that using the Newton-GS method and the percentage reduction can be up to 24.60. And here the Newton-4EGGS method is the most computationally efficient and the reduction in the number of iterations compared to Newton-GS

Table 6: Decreasing Percentage of Iterations of Newton-4EGGS Relative to Newton-GS and Newton-2EGGS

n	Range of decreasing percentage of iterations		
	$NGS-2NEGGS$	$NGS-4NEGGS$	$2NEGGS-4NEGGS$
1	$0.3 \backsim 4.76$	$0.89 \backsim 13.33$	$0.60 \backsim 9.00$
2	$0.11 \backsim 3.81$	$0.43 \backsim 5.41$	$0.32 \backsim 1.67$
3	$0.27 \backsim 8.55$	$0.90 \backsim 13.31$	$0.63 \backsim 8.20$
4	$0.00 \backsim 24.6$	$1.11 \backsim 57.16$	$1.11 \backsim 43.18$

Table 7: Decreasing Percentage of computational time of Newton-4EGGS Relative to Newton-GS and Newton-2EGGS

n	Computing time					
	$NGS(I)$	$2NEGGS(II)$	$4NEGGS(III)$	$\frac{(I-II)}{I}$	$\frac{(I-III)}{I}$	$\frac{(II-III)}{II}$
1	2167.36	2134.34	2084.57	1.52	3.82	2.33
2	2282.83	2265.99	2243.15	0.74	1.74	1.01
3	2580.76	2567.72	2527.04	0.51	2.08	1.58
4	8908.70	6966.37	4450.22	21.8	50.05	36.12

and Newton-2EGGS can be up to 54.16% and 43.18%.

As expected, the two methods we used are much faster to compute than the reference method. Also in Table 7, it can be seen that the computation time of our proposed Newton-4EGGS is up to 50.05% faster than the Newton-GS method, and the Newton-4EGGS method is in turn 36.12% faster than the Newton-2EGGS, so it can be concluded that our proposed iterative method is able to achieve a substantial improvement in the number of iterations and the execution time over the reference Newton-GS method and Newton-4EGGS performs better.

6. Conclusion

This paper focuses on algorithms for solving large-scale multi-objective constrained optimization problems. Based on the characteristics of the objective functions and constraints, new Newton Group iterative methods is proposed based on the classical optimization algorithms: weighting, Lagrange multiplier, and Newton method. The convergence of the method is also theoretically analyzed, and then the method is used in four different examples, and the computational results of the new method and the reference method in terms of the number of iterations and the computational time are shown in the form of tables and graphs. Finally, a comparative analysis verifies the obvious superiority of our proposed method compared with the reference Newton-Gauss-Seidel method, especially in terms of the number of iterations and computational time, which greatly reduces the computational complexity. This provides a new idea for solving large-scale multi-objective constrained optimization problems. Then how to innovate the method to make the computation more efficient or to compute the ultra-large scale multi-objective optimization problem will be the direction of our next research. For future work, this study will be

extended to investigate on the use of two-step iterative methods such as arithmetic mean [27, 33], Geometric mean[28] and AGE[7] as a smoother combined with the Newton method to solve the proposed multi-objective constrained optimization problems.

Acknowledgements

The research was found by the Universiti Malaysia Sabah's UMSGreat Research Grant No. GUG0568-1/2022.

References

[1] M S Bazaraa, H D Sherali, and C M Shetty. *Nonlinear multiobjective optimization.* John wiley & sons, New York, USA, 2006.

[2] R H Byrd and J. Nocedal. A tool for the analysis of quasi-Newton methods with application to unconstrained minimization. *SIAM Journal on Numerical Analysis*, 26(3):727–739, 1989.

[3] B Cakir, F Altiparmak, and B Dengiz. Multi-objective optimization of a stochastic assembly line balancing: A hybrid simulated annealing algorithm. *Computers & Industrial Engineering*, 60(3):376–384, 2011.

[4] P Cheng, J Sulaiman, K Ghazali, M K M Ali, and M M Xu. Application of newton-sor iteration with linear weighted lagrange approach for solving multi-objective constrained optimization problems. In *In International Conference on Advances in Computational Science and Engineering*, pages 3–16, Manila, Philippines, 2024. Springer Nature Singapore.

[5] P Cheng, J Sulaiman, K Ghazali, M K M Ali, and M M Xu. Newton-SOR Iterative Method with Lagrangian Function for Large-Scale Nonlinear Constrained Optimization Problems. *Journal of Advanced Research in Applied Sciences and Engineering Technology*, 46(2):251–262, 2025.

[6] K Deb, K Sindhya, and J Hakanen. *Multi-objective optimization.* Decision sciences, CRC Press, 2016.

[7] D J Evans. The alternating group explicit (AGE) matrix iterative method. *Applied Mathematical Modelling*, 11(4):256–263, 1987.

[8] D J Evans and A R B Abdullah. Group explicit methods for parabolic equations. *International journal of computer mathematics*, 14(1):73–105, 1983.

[9] S K S Fan and J M Chang. A parallel particle swarm optimization algorithm for multi-objective optimization problems. *Engineering Optimization*, 41(7):673–697, 2009.

[10] Z Fan, W Li X Cai, H Li C Wei, Q Zhang, K Deb, and E Goodman. Difficulty adjustable and scalable constrained multiobjective test problem toolkit. *Evolutionary computation*, 28(3):339–378, 2020.

[11] Z Fan, W Li, X Cai, H Li, K Hu, and H Yin. Difficulty controllable and scalable constrained multi-objective test problems. In *2015 International Conference on Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration.*, pages 76–83, Wuhan, China, 2015. IEEE.

[12] K Ghazali, J Sulaiman, Y Dasril, and D Gabda. Newton method with explicit group iteration for solving large scale unconstrained optimization problems. In *Journal of Physics: Conference Series.*, volume 1132. IOP Publishing, Putrajaya, Malaysia, 2018.

[13] K Ghazali, J Sulaiman, Y Dasril, and D Gabda. Newton-MSOR Method for Solving Large-Scale Unconstrained Optimization Problems with an Arrowhead Hessian Matrices. *Transactions on Science and Technology*, 6(2-2):228–234, 2019.

[14] K Ghazali, J Sulaiman, Y Dasril, and D Gabda. An improvement of computing newton's direction for finding unconstrained minimizer for large-scale problems with an arrowhead hessian matrix. In *Computational Science and Technology: 6th ICCST 2019.*, volume 1132. Springer Singapore, Kota Kinabalu, Malaysia, 2020.

[15] N Gunantara. A review of multi-objective optimization: Methods and its applications. *Cogent Engineering*, 5(1):1–16, 2018.

[16] Y Y Han, D W Gong, X Y Sun, and Q K Pan. An improved NSGA-II algorithm for multi-objective lot-streaming flow shop scheduling problem. *International Journal of Production Research*, 52(8):2211–2231, 2014.

[17] M Hassan and A Baharum. A new logarithmic penalty function approach for nonlinear constrained optimization problem. *Decision Science Letters*, 8(3):353–362, 2019.

[18] D M Jaeggi, G T Parks, T Kipouros, and P J Clarkson. The development of a multi-objective Tabu Search algorithm for continuous optimisation problems. *European Journal of Operational Research*, 185(3):1192–1212, 2008.

[19] H Jain and K Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. *IEEE Transactions on evolutionary computation*, 18(4):602–622, 2013.

[20] J E Dennis Jr and R B Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations.* Society for Industrial and Applied Mathematics, USA, 1996.

[21] J R Kasprzyk, P M Reed, B R Kirsch, and G W Characklis. Managing population and drought risks using many-objective water portfolio planning under uncertainty. *Water Resources Research*, 45(12):1–18, 2009.

[22] A Konak, D W Coit, and A E Smith. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability engineering & system safety*, 91(9):992–10078, 2006.

[23] Y P Laptin and T O Bardadym. Problems related to estimating the coefficients of exact penalty functions. *Cybernetics and systems analysis*, 55:400–412, 2019.

[24] J V Lung and J Sulaiman. On quarter-sweep finite difference scheme for one-dimensional porous medium equations. *International Journal of Applied Mathematics*, 33(3):439, 2020.

[25] R J Lygoe, M Cary, and P J Fleming. A many-objective optimisation decision-making process applied to automotive diesel engine calibration. In K Deb, editor, *Simulated Evolution and Learning.*, volume 11b. Springer, Kanpur, India, 2010.

[26] K Miettinen. *Nonlinear multiobjective optimization.* Springer Science & Business Media, New York, USA, 1999.

[27] M S Muthuvalu and J Sulaiman. Quarter-Sweep Arithmetic Mean (QSAM) iterative method for second kind linear Fredholm integral equations. *Appl. Math. Sci*, 4:2943–2953, 2010.

[28] M Sundaram Muthuvalu and J Sulaiman. Half-sweep geometric mean iterative method for the repeated Simpson solution of second kind linear Fredholm integral equations. *Proyecciones (Antofagasta)*, 31(1):65–79, 2012.

[29] V H Nguyen and J J Strodiot. On the convergence rate for a penalty function method of exponential type. *Journal of Optimization Theory and Applications*, 27:495–508, 1979.

[30] J Ning, C Zhang, P Sun, and Y Feng. Comparative study of ant colony algorithms for multi-objective optimization. *Information*, 10(1):11, 2018.

[31] J Nocedal. Theory of algorithms for unconstrained optimization. *Acta numerica*, 1:199–242, 1992.

[32] R T Rockafellar. The multiplier method of Hestenes and Powell applied to convex programming. *Journal of Optimization Theory and applications*, 12(6):555–562, 1973.

[33] V Ruggiero and E Galligani. An iterative method for large sparse linear systems on a vector computer. *Computers & Mathematics with Applications*, 20(1):25–28, 1990.

[34] A Saudi and J Sulaiman. Red-black strategy for mobile robot path planning. In *Proceedings of the international multiconference of engineers and computer scientists*, Hong Kong, China, 2010. International Association of Engineers.

[35] A Saudi and J Sulaiman. Robot path planning using four point-explicit group via nine-point laplacian (4EG9L) iterative method. *Procedia Engineering*, 41:182–188, 2012.

[36] M. K. Sharma, A. K. Bhargava, S. Kumar, L. Rathour, L. N. Mishra, and S. Pandey. A fermatean fuzzy ranking function in optimization of intuitionistic fuzzy transportation problems. *Advanced Mathematical Models and Applications*, 7(2):191–204, 2022.

[37] M K Sharma, S Chaudhary, L Rathour, and V N Mishra. Modified genetic algorithm with novel crossover and mutation operator in travelling salesman problem. *Sigma Journal of Engineering and Natural Sciences*, 42(6), 2024.

[38] J Sulaiman, M Othman, and M K Hasan. Red-black half-sweep iterative method using triangle finite element approximation for 2d poisson equations. In Y Shi, G D Albada, J Dongarra, and P M A Sloot, editors, *Computational Science–ICCS 2007: 7th International Conference.*, volume 4487. Springer Berlin Heidelberg, Beijing, China, 2007.