



Improved Efficient Global Optimization Algorithm for Shape Optimization of Aerodynamics

付, 航

(Degree)

博士 (工学)

(Date of Degree)

2025-09-25

(Date of Publication)

2026-09-01

(Resource Type)

doctoral thesis

(Report Number)

甲第9332号

(URL)

<https://hdl.handle.net/20.500.14094/0100498372>

※ 当コンテンツは神戸大学の学術成果です。無断複製・不正使用等を禁じます。著作権法で認められている範囲内で、適切にご利用ください。



Improved Efficient Global Optimization Algorithm for Shape Optimization of Aerodynamics
(空力形状最適化のための改良型効率的グローバル最適化アルゴリズム)

by

Fu Hang

A Dissertation

Submitted to Graduate School of System Informatics

July 2025

TABLE OF CONTENTS

ABSTRACT	I
ACKNOWLEDGEMENT	III
LIST OF TABLES	IV
LIST OF FIGURES	V
NOMENCLATURE	VII
CHAPTER 1. INTRODUCTION	1
1.1 Surrogate-based Optimization	1
1.2 Previous Investigations	1
1.3 Objective of the Present Research	4
1.4 Overview of the Present Research	4
CHAPTER 2. METHODOLOGIES OF SURROGATE-BASED OPTIMIZATION	6
2.1 Design of Experiments	6
2.1.1 Latin hypercube design	6
2.1.2 Optimal Latin hypercube design	7
2.2 Kriging Surrogate Models	9
2.2.1 Ordinary kriging	9
2.2.2 Gradient-enhanced kriging	11
2.3 Point Infills Criteria	14
2.3.1 Expected improvement	14
2.3.2 Pseudo expected improvement	14
2.4 Krigings Based Efficient Global Optimization Algorithm Systems	17
2.5 Gradient-enhanced Kriging Based Parallel Efficient Global Optimization Algorithm System	18
CHAPTER 3. INVESTIGATION OF THE INFLUENCE OF THE SPACE-FILLING QUALITY OF INITIAL SAMPLE POINTS ON THE PERFORMANCE OF THE PROPOSED OPTIMIZATION ALGORITHM SYSTEM	20
3.1 Classification of Space-filling Quality	20
3.2 Experimental Setting	22
3.3 Results and Discussion	22
CHAPTER 4. INVESTIGATION OF THE INFLUENCE OF THE PARALLEL POINT INFILLS STRATEGY ON THE PERFORMANCE OF THE PROPOSED OPTIMIZATION ALGORITHM SYSTEM	27
4.1 Experimental Setting	27
4.2 Results and Discussion	29

4.2.1 Influence of the parallel point infills strategy on the solution quality	29
4.2.2 Influence of the parallel point infills strategy on the convergence speed.....	31
4.2.3 Influence of the parallel point infills strategy on the optimization efficiency	33
4.2.4 Influence of the parallel point infills strategy on the diversity of parallel EGO algorithm	37
CHAPTER 5. CONSTRUCTION AND VALIDATION OF THE PROPOSED IMPROVED OPTIMIZATION ALGORITHM SYSTEM	42
5.1 Matching the Suitable Optimizer	42
5.1.1 Key role of optimizer in current optimization system	42
5.1.2 Experimental setting	44
5.1.3 Results and discussion	46
5.2 Performance Verification of the Proposed Optimization Algorithm System	52
5.2.1 Verification on benchmark test functions	52
5.2.2 Verification on airfoil aerodynamics optimization problem.....	54
5.2.3 Verification on vehicle aerodynamics optimization problem.....	61
5.2.4 Quantitative analysis on real-world optimization problem of aerodynamics	65
CHAPTER 6. CONCLUSIONS AND RECOMMENDATIONS	68
6.1 Conclusions.....	68
6.2 Recommendations for Future Work	70
REFERENCES	71

ABSTRACT

The parallel efficient global optimization (EGO) algorithm was developed to leverage the rapid advancements in high-performance computing. However, conventional parallel EGO algorithm based on ordinary kriging (OK) model faces limitations when applied to complex, nonlinear, and high-dimensional engineering optimization problems, as OK model relies only on sample point locations, restricting the available training information. To further enhance the optimization potential of the parallel EGO algorithm, an improved system that integrates the parallel EGO algorithm with gradient-enhanced kriging (GEK) is proposed. The integration serves two main purposes: on the one hand, GEK provides a more comprehensive representation of the design space, improving the surrogate model's predictive accuracy; on the other hand, as the number of infill points increases, GEK mitigates the unreliability of the pseudo expected improvement (PEI) function, facilitating the efficient search for high-quality global optima.

In employing a GEK based parallel EGO optimization system, it is essential to investigate how optimal sample size and space-filling properties of sampling points contribute to reducing computational resource consumption while preserving high optimization performance. The influence of sample size and space-filling quality of sample points on the optimization performance of GEK based parallel EGO optimization system has been investigated. The convergence performance of the current optimization system is compared across various multi-modal test functions using different sample sizes and space-filling quality levels. Statistical analysis results shown that, increasing sample size does not consistently improve optimization performance under the same space-filling quality level. Additionally, under the same sample size, the non-completely convergent optimal Latin-hypercube design (OLHD) effectively promotes the optimization system to rapidly converge to a high-quality solution within a limited number of iterations.

The impact of parallel infill strategies on the optimization performance of parallel EGO algorithms based on kriging surrogate models of different fidelity was investigated through a comparative study. The nine benchmark test functions with different optimization problem characteristics were selected as optimization test objects, and the results were systematically analyzed from the perspectives of convergence performance, optimization efficiency, and algorithmic diversity. The analysis indicates that increasing the number of parallel infill points per optimization cycle has minimal impact on the performance of the GEK based parallel EGO algorithm system, highlighting the role of high-fidelity surrogate models in improving convergence stability, optimization efficiency, and algorithmic robustness.

To ensure high performance, various benchmark functions were used to evaluate the impact of different optimizer on the proposed system, ultimately identifying the most effective optimizer.

Additionally, optimization tests were conducted on benchmark functions, airfoil aerodynamic optimization, and automotive aerodynamic optimization to compare the proposed system with traditional OK based parallel EGO system in optimization performance. The results demonstrate that the modified Gaussian bare-bones differential evolution (MGBDE) optimizer consistently accelerated convergence to high-quality solutions in most benchmark functions, achieving the highest average ranking across all tests. Compared to traditional OK based parallel EGO system, the proposed method showed superior convergence speed, solution quality, and computational efficiency. Specifically, for airfoil and automotive aerodynamic optimization, when achieving the target performance improvements of 2.4% and 7.0%, the proposed system reduced the wall-clock time cost from 1.65h to 0.33h and from 203h to 14.5h, respectively, significantly improving optimization efficiency.

ACKNOWLEDGEMENT

Completing this doctoral dissertation would not have been possible without the support, guidance, and encouragement of many individuals to whom I owe my deepest gratitude.

First and foremost, I would like to express my sincerest appreciation to my supervisor, Prof. Makoto Tsubokura, for his invaluable guidance, patience, and unwavering support throughout my PhD journey. His profound knowledge, insightful feedback, and constant encouragement have been instrumental in shaping my research and academic growth. I am deeply grateful for the freedom he granted me to explore my ideas while always providing timely and constructive advice.

I am also deeply indebted to Prof. Takuji Nakashima, Prof. Rahul Bale and Dr. Qingyu Wang for their thoughtful suggestions and critical feedback, which have significantly improved the quality of this work. Their expertise and rigorous standards pushed me to refine my research and broaden my perspectives. My special thanks go to the defense committee members for their insightful feedback on my research.

My heartfelt thanks go to my colleagues and friends at the Computational Fluid Dynamics Laboratory for their academic support and companionship throughout my campus life. I am especially grateful to Ms. Yuriko Uwamori, our laboratory secretary, for serving as an invaluable bridge between me, my supervisor, and the university administration.

I would also like to acknowledge the financial support provided by China Scholarship Council, which enabled me to focus on my research without financial constraints.

Finally, none of this would have been possible without the unconditional love and support of my family. To my parents, thank you for your endless belief in me and for always encouraging me to pursue my dreams.

To all those who have contributed to my academic and personal growth in ways big and small—I am forever grateful.

Hang Fu

Computational Fluid Dynamics Lab.

Graduate School of System Informatics

Kobe University, Japan

May, 2025

LIST OF TABLES

Table 2.1	Pseudocode of GEK based parallel EGO algorithm
Table 3.1	Classification of space-filling quality for LHD and OLHD
Table 3.2	Wilcoxon signed-rank test between sample size of 10×5 and other sample size, respectively, under each space-filling quality level
Table 3.3	Mean rankings of space-filling quality levels with each sample size for all benchmark test functions
Table 4.1	Details of benchmark test functions
Table 4.2	Statistical results of the converged solutions obtained by the parallel EGO algorithms based on OK and GEK models under different point-filling strategies
Table 4.3	The statistics of the number of cycles required to reach the expected stopping criterion under different point-filling strategies using parallel EGO algorithms based on OK and GEK, respectively
Table 5.1	Parameters setting for each optimizer
Table 5.2	Performance comparison of different optimizers
Table 5.3	Mean rankings of optimizers for all tested functions
Table 5.4	Details of benchmark test functions
Table 5.5	Details of design variables
Table 5.6	Validation of aerodynamic characteristics of C_l and C_d of airfoil with previous studies
Table 5.7	Optimal solutions obtained by GEK and OK based parallel algorithm system
Table 5.8	Quantitative comparison of performance of the GEK and OK based parallel algorithm system in optimization process

LIST OF FIGURES

Figure 1.1	Characteristics of engineering optimization problems
Figure 1.2	The typical process of surrogate-based optimization method (e.g. vehicle shape optimization)
Figure 2.1	Visualization of 6×2 LHD
Figure 2.2	Flowchart of MESE algorithm
Figure 2.3	The variation trend of spatial correlation between sample points under different hyper-parameter values (one-dimensional correlation function)
Figure 2.4	Comparison of fitting accuracy between gradient-enhanced kriging (GEK) and ordinary kriging (OK). (a) Real two-dimensional Griewank function, (b) OK fitting, (c) GEK fitting
Figure 2.5	Comparison of searching new points by EI and PEI criterion, respectively. (a) The first iteration, (b) The second iteration, (c) The third iteration, (d) The forth iteration.
Figure 2.6	Framework of the kriging-based EGO algorithm systems
Figure 2.7	Framework of GEK based parallel EGO algorithm system
Figure 3.1	The trend of ϕ_p value quartiles as the number of sampling times increases
Figure 3.2	Rankings of different space-filling quality level with different sample sizes for each benchmark test function
Figure 4.1	Average convergence curves of the OK-based parallel EGO algorithm on all tested problems
Figure 4.2	Average convergence curves of the GEK-based parallel EGO algorithm on all tested problems
Figure 4.3	Composition of the wall-clock time cost within a single optimization cycle for practical problems
Figure 4.4	The process for searching multi-points using the PEI criterion within a single optimization cycle
Figure 4.5	Population diversity curves of the OK-based parallel EGO algorithm on all tested problems
Figure 4.6	Population diversity curves of the GEK-based parallel EGO algorithm on all tested problems

Figure 4.7	Exploration and exploitation curves of the OK-based parallel EGO algorithm on all tested problems
Figure 4.8 problems	Exploration and exploitation curves of the GEK-based parallel EGO algorithm on all tested problems
Figure 5.1	Comparison of changes in EI and PEI functions during the process of point infills
Figure 5.2	Wilcoxon signed-rank test for MGBDE, PSO and DE optimizer with other optimizers, respectively
Figure 5.3	Friedman test between each optimizer for all tested functions
Figure 5.4	Average convergence curves of GEK-based parallel EGO algorithm with different optimizers on all tested functions
Figure 5.5	Wilcoxon signed-rank test between OK and GEK based parallel EGO system
Figure 5.6	Average convergence curves of comparative systems on benchmark tested function
Figure 5.7	Baseline model of NACA0012 airfoil and definition of design variables
Figure 5.8	Validation of the C_p value of airfoil surface with previous studies
Figure 5.9	Convergence curves of Cl/Cd ratio with GEK and OK based parallel EGO algorithm system, respectively
Figure 5.10	Surface pressure coefficient distribution of the baseline airfoil model and the optimized airfoil model using GEK and OK based parallel algorithm system, respectively
Figure 5.11	Pressure contour of the baseline airfoil model and the optimized airfoil model using GEK and OK based parallel algorithm system, respectively
Figure 5.12	Baseline vehicle shape, and deformed 10 vehicle shapes based on definition of design variables
Figure 5.13	Validation of the predicted C_d values of the neural network model and numerical simulation experiment, respectively
Figure 5.14	Validation of the obtained gradient information of finite difference and automatic differentiation technique, respectively
Figure 5.15	Convergence curves of C_d values with GEK and OK based parallel EGO algorithm system, respectively
Figure 5.16	Wall-clock time cost components of each optimization cycle

NOMENCLATURE

Roman Symbols

n_p	Number of sample points
n_v	Dimension of design variables
X_i	Vector of spatial coordinates
x_i	Spatial coordinates
d_{ij}	Distance between two random sample points
p	Parameter of calculating space-filling quality level
t	Parameter of calculating distance between two random sample points
$\hat{y}$	Predicted function value
F	Regression matrix
R	Correlation matrix
r	Correlation vector
w	Linear system
s^2	Mean squared error
y	Vector of function value
Q	Correlation matrix
$R^{+,-}$	Sum of rankings
Re	Reynolds number
v	Freestream velocity
S	Reference area

Greek Symbols

$\emptyset_p$	Space-filling quality level
$\hat{\mu}$	Mean term
$\hat{\epsilon}$	Error term
ψ	Correlation function
θ	Hyper-parameter of kriging
$\hat{\sigma}^2$	Predicted variance
$\Phi(x)$	Standard normal cumulative distribution function
$\phi(x)$	Probability density function

ρ Fluid density

Abbreviation

EGO	Efficient Global Optimization, a global optimization algorithm
PEI	Pseudo Expected Improvement, a point-infilling criterion
GEK	Gradient-enhanced Kriging, a surrogate model
OK	Ordinary Kriging, a surrogate model
SBO	Surrogate-based Optimization, an optimization technology
EI	Expected Improvement, a point-infilling criterion
q-EI	q-point Expected Improvement, a point-infilling criterion
KB	Kriging Believer, a point-infilling criterion
CL	Constant Liar, a point-infilling criterion
IF	Influence Functions, a series of correlation functions
GS-SQP	Golden Search Sequential Quadratic Programming, a gradient based optimization algorithm
DE	Differential Evolution, an evolutionary optimization algorithm
PSO	Particle Swarm Optimization, a population based optimization algorithm
EAGDE	Enhanced Adaptive Guided Differential Evolution, an evolutionary optimization algorithm
CJADE	Chaotic local search-based JADE, an evolutionary optimization algorithm
MGBDE	Modified Gaussian Bare-bones Differential Evolution, an evolutionary optimization algorithm
GTO	Artificial Gorilla Troop Optimizer, a nature-inspired optimization algorithm
LHD	Latin Hypercube Designs, a sampling method
Vaeopy	Vehicular aerodynamic engineering optimization in Python, an optimization toolbox in Python
CFD	Computational Fluid Dynamics, a numerical simulation technology
Cl	Lift Coefficient, a dimensionless parameter that describes the forces acting on an object in fluid dynamics
Cd	Drag Coefficient, a dimensionless parameter that describes the forces acting on an object in fluid dynamics
DFM	Design for Manufacturing, a morphing technology in ANSA software
SIMPLE	Semi-implicit Method of the Pressure-linked Equations, a numerical solution algorithm
Cp	Pressure Coefficient, a dimensionless parameter that describes the pressure distribution in fluid dynamics
OLHD	Optimal Latin Hypercube Designs, a sampling method
CUBE	Complex Unified Building cubE method, a scalable framework for large-scale industrial simulations

CHAPTER 1. INTRODUCTION

1.1 Surrogate-based Optimization

Nearly all engineering optimization problems are computationally expensive and involve complex black-box functions [1]. Such problems allow for the observation of input data and output results, while the specific mathematical relationship between the two remains unclear (Figure 1.1). Such as finite element analysis and computational fluid dynamics. With the growing demands of modern industries for improved product quality and production efficiency, the pursuit of highly efficient optimization techniques has become a shared objective among researchers.

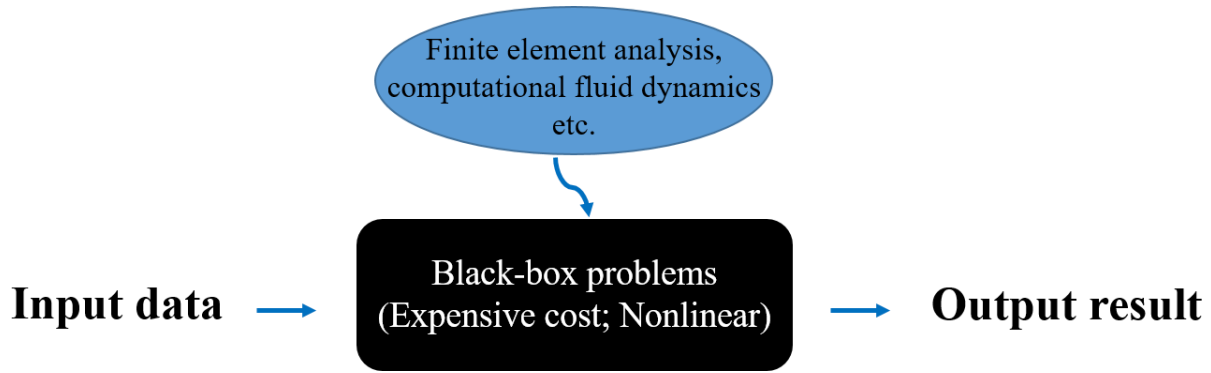


Figure 1.1. Characteristics of engineering optimization problems.

To overcome the high computational cost and inefficiency associated with directly applying traditional methods to engineering optimization problems, surrogate-based optimization (SBO) methods have gained increasing attention over the past two decades [2]. Figure 1.2 shows the typical process of SBO in vehicle shape optimization: sampling, deformation, experimentation, surrogate model construction, and search optimal solution. SBO provides a cost-effective way to predict function values and is widely used in engineering applications [3], such as gas turbine blade shape optimization [4], airfoil optimization [5, 6], helicopter rotor blade design [7], automotive aerodynamic performance optimization [8, 9], aerodynamic optimization of streamlined bridge deck [10] and design optimization for battery thermal management system [11].

1.2 Previous Investigations

In the early stages, engineers relied solely on empirical knowledge and theoretical research to improve optimization targets, a process that required extensive and costly experimental support [12]. Advances in computing technology, especially computational fluid dynamics and computer-aided design,

have greatly improved the efficiency of engineering optimization. However, the associated computational resource demands remain prohibitively high [13].

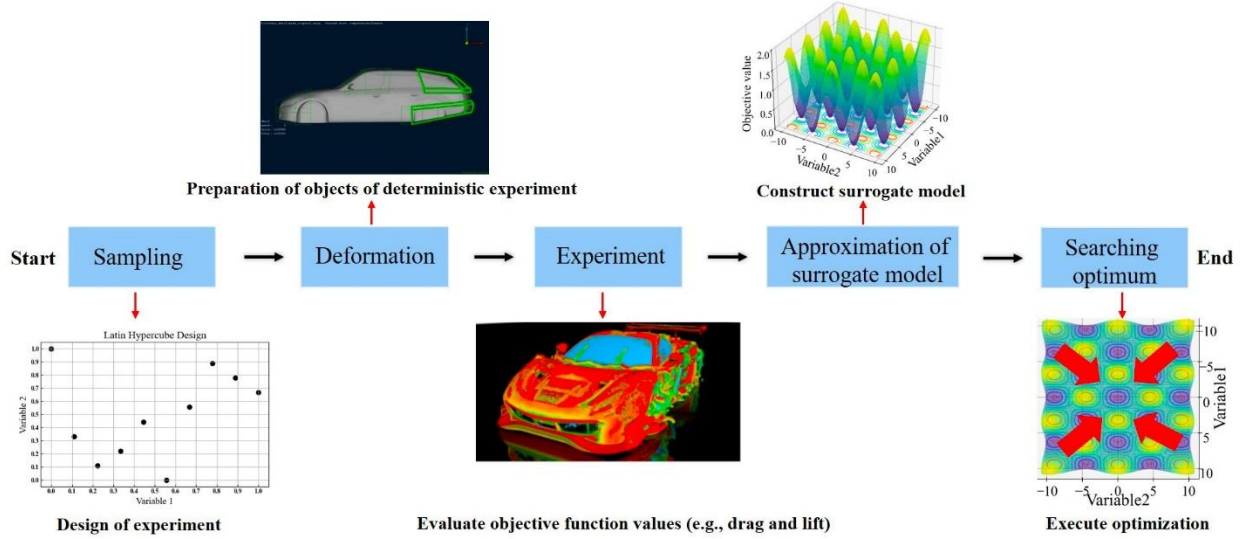


Figure 1.2. The typical process of surrogate-based optimization method (e.g. vehicle shape optimization).

With the rapid development of intelligent computing, engineering optimization is gradually overcoming its reliance on extensive human and material resources. Various optimization algorithms, including bio-inspired methods like differential evolution and genetic algorithms, as well as population-based techniques like particle swarm optimization, have significantly improved engineering optimization [14–16]. However, traditional optimization algorithms are prone to becoming trapped in local optima. To address this issue, recent years have witnessed the emergence of enhanced and novel optimization algorithms [17–22]. Such advanced optimizers have effectively mitigated the limitations of traditional algorithms in avoiding entrapment in local optima. However, applying these algorithms directly often requires thousands or even tens of thousands of objective function evaluations [23], making them impractical for product design. To address the high computational costs and inefficiency associated with direct application of optimizers, SBO methods have emerged as one of the most effective alternative approaches.

Among traditional surrogate models, kriging is widely used in engineering for its ability to predict unknown points and quantify uncertainty [24, 25]. However, the accuracy of the ordinary kriging (OK) model, constructed with a limited number of samples, may not meet the requirements of practical engineering optimization problems. The OK model, as an approximation of the true function, deviates in

regions with sparse samples. Limited samples also constrain the model's training, leading to suboptimal solutions and a higher risk of missing the global optimum.

Two key approaches address the existing limitations. On the one hand, Forrester et al. [26] proposed extended regression kriging with re-interpolation, which automatically filters noisy data introduced by experiments and yields a higher-fidelity surrogate model. To extract more efficient information from the existing sample set, Laurenceau et al. [27] used function values at sample points to build a surrogate model and incorporated additional information, such as gradient values, to aid the construction process. The collaborative approach has demonstrated effectiveness in practical engineering optimization problems [28-32]. Additionally, Han et al. [33, 34] proposed kriging-based variable-fidelity surrogate models to improve the efficiency and accuracy of kriging-based optimization method. However, relying solely on this approach often reaches the fitting limit of the sample points. This makes it challenging to address highly nonlinear and high-dimensional optimization problems. On the other hand, kriging is well-suited for developing a point infill criterion, as it predicts unknown points and quantifies the uncertainty of those predictions. Building on this concept, Jones et al. [35] proposed the efficient global optimization (EGO) algorithm, which utilizes the expected improvement (EI) criterion to identify new sample points with the highest potential for improvement. This approach achieves better efficiency and accuracy than previous methods and has been widely applied in practical engineering optimization problems [36, 37]. However, during each point infill cycle, only one potentially improving sample point is added to the training sample set. Consequently, external computational resources are not fully utilized, limiting the improvement in optimization efficiency.

To address this issue, parallel point infill criteria have been developed. Initially, Ginsbourger et al. [38] proposed the q-point expected improvement (q-EI) criterion, which selects multiple points based on a criterion in each cycle. Despite enhancing optimization efficiency, the method remains time-consuming. Further improvements include the kriging believer (KB) and constant liar (CL) strategies. The KB criterion relies on the analytically known EI and optimization in d dimensions, which is computationally affordable. However, over-relying on the predictor outside the observed data range may result in convergence to suboptimal regions. The CL strategy considers the current minimum value to be a “liar” and updates the kriging model to give the next candidate point. Although the computational cost of the CL strategy is much lower than that of the q-EI criterion, reliance on fabricated information from the ‘liar’ leads to exploration in poorer regions, causing the optimization to become trapped in local optima. In recent years, Zhan et al. [39] proposed a parallel point infill criterion, called pseudo expected improvement (PEI), based on the observed behavior of the EI function. The approach simulates the changes in the EI function caused by adding a new sample point to the training set, without updating the

kriging model. As a result, the parallel EGO algorithm with the PEI criterion demonstrates superior performance and efficiency compared to other parallel EGO algorithms. Zhan et al. proposed a parallel EGO algorithm based on the OK surrogate model. However, its optimization efficiency can be further improved by enhancing the accuracy of the surrogate model. The PEI function is constructed by multiplying the original EI function with multiple influence functions (IFs), corresponding to the number of parallel sample points. As the number of parallel sample points increases, the reliability of the PEI function declines significantly due to the limited accuracy of the OK model with a finite number of training samples, especially in high-dimensional optimization problems. Increasing the number of training samples can improve model accuracy but may lead to overfitting [40]. Wang et al. [41] attempted to enhance optimization efficiency by integrating the regression kriging model with parallel EGO. However, since regression kriging primarily smooths the fitting trend by filtering numerical noise, its contribution to improving optimization efficiency remains limited. Existing parallel EGO studies mainly use only the spatial positions of training samples while ignoring gradient information, which reflects changes in spatial position. Therefore, integrating gradient-enhanced kriging (GEK) with parallel EGO may significantly improve optimization efficiency and enable the handling of high-dimensional engineering optimization problems, making it a promising direction for further investigation.

1.3 Objective of the Present Research

To further enhance the optimization potential and efficiency of the parallel EGO algorithm and achieve the greatest possible improvements within a reasonable computational cost for engineering optimization problems, this research proposed an improved GEK-based parallel EGO algorithm system. This system integrates gradient-enhanced techniques, a parallel point infill criterion, and powerful optimizer engines. Furthermore, to ensure the performance advantages of the proposed improved optimization system in practical engineering applications, this study systematically investigates three critical aspects affecting the system's overall optimization performance: the matching mechanism of search engines within the system; the selection of space-filling properties for initial sample points during initialization; the determination of point-filling quantities in parallel optimization loops.

1.4 Overview of the Present Research

The organization of present research is as follows: Chapter 2 introduces the methodologies of kriging-based efficient global optimization; Chapter 3 constructs and validates the proposed improved optimization system; Chapter 4 investigates the influence of the space-filling quality of initial sample points on the performance of the proposed improved optimization system; Chapter 5 investigates the

influence of the parallel point infills strategy on the performance of the proposed improved optimization system, while Chapter 6 concludes the research and outlines potential directions for future research.

CHAPTER 2. METHODOLOGIES OF SURROGATE-BASED OPTIMIZATION

2.1 Design of Experiments

In practical engineering optimization, Design of Experiments (DoE) is a systematic and efficient methodology used to plan, conduct, and analyze experiments, aiming to identify key influencing factors, optimize process parameters, or improve product performance.

2.1.1 Latin hypercube design

For costly engineering optimization problems, conducting numerous experimental evaluations to obtain the desired solution is often infeasible. Therefore, effectively utilizing a finite number of evaluation points to approximate the properties of the entire design space emerges as a practical strategy to minimize experimental costs. Among various DoE methods, Latin hypercube design (LHD) is widely utilized for determining the locations of evaluation points due to its non-collapsing features and orthogonality [42, 43]. A dataset with n_p points and n_v variables is often represented as an $n_p \times n_v$ matrix $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_{n_p}]^T$, where each row $\mathbf{X}_i = [x_{i1}, x_{i2}, \dots, x_{in_v}]$ represents a sample point and each column represents a variable. LHD divides each column into n_p equal levels and ensures that one point per level. Random permutations are then applied to obtain an LHD sample points. Figure 2.1 clearly illustrates a size of 6×2 LHD sampling. However, due to the presence of these random permutations, ensuring the spatial uniformity of the LHD cannot be guaranteed.

The space-filling quality of the sample points is considered importantly to describe the entire design space in literature. The ϕ_p criterion is a widely adopted criteria for assessing the space-filling quality of sample points [44]. It is defined as follows:

$$\phi_p = \left[\sum_{i=1}^{n_p-1} \sum_{j=i+1}^{n_p} d_{ij}^{-p} \right]^{1/p} \quad (1)$$

Where d_{ij} represents the distance between any two random sample points. The expression reveals that maximizing the distance between any two random sample points is equivalent to minimizing the ϕ_p value of space-filling quality. The formula for d_{ij} is as follows:

$$d_{ij} = \left[\sum_{k=1}^{n_v} |x_{ik} - x_{jk}|^t \right]^{1/t} \quad (2)$$

Where x_{ik} and x_{jk} represent the k th components of the i and j points, respectively. Typically, $p = 50$ and $t = 1$ are suitable settings.

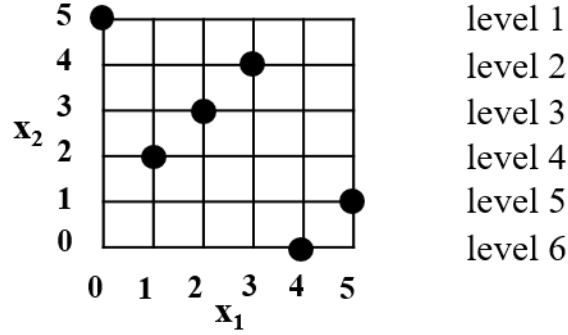


Figure 2.1. Visualization of 6×2 LHD.

2.1.1.2 Optimal Latin hypercube design

To enhance the uniformity of evaluation points distribution and capture more comprehensive information about the design space, researchers have proposed various optimization algorithms aimed at improving the space-filling quality of LHD [45-49]. The modified enhanced stochastic evolutionary (MESE) algorithm, proposed by Wang et al., is currently one of the most effective algorithms for optimizing the performance of conventional random LHD samples. Figure 2.2 illustrates the optimization workflow of the MESE algorithm. The MESE algorithm is an improvement of a standard ESE algorithm. It contains two loops (inner and outer loops). The main task of the inner loop is to randomly select J distinct elements from each column and exchange their positions to generate a new sample point design. By comparing the ϕ_p value of new optimal design with the current best design, the sample point design is updated accordingly. The primary task of the outer loop is to adaptively update the hyper-parameters of the optimization algorithm based on the current state of the optimization process. However, it is important to note that additional optimization process in DOE can increase time in costs, raising the question of whether this increased time investment yields significant improvements in the overall efficiency of the optimization process. The sampling of the design space determines the distribution of valuable information acquisition across the entire design space, directly influencing the exploration direction of the optimization algorithm and ultimately impacting the efficiency of the overall optimization process.

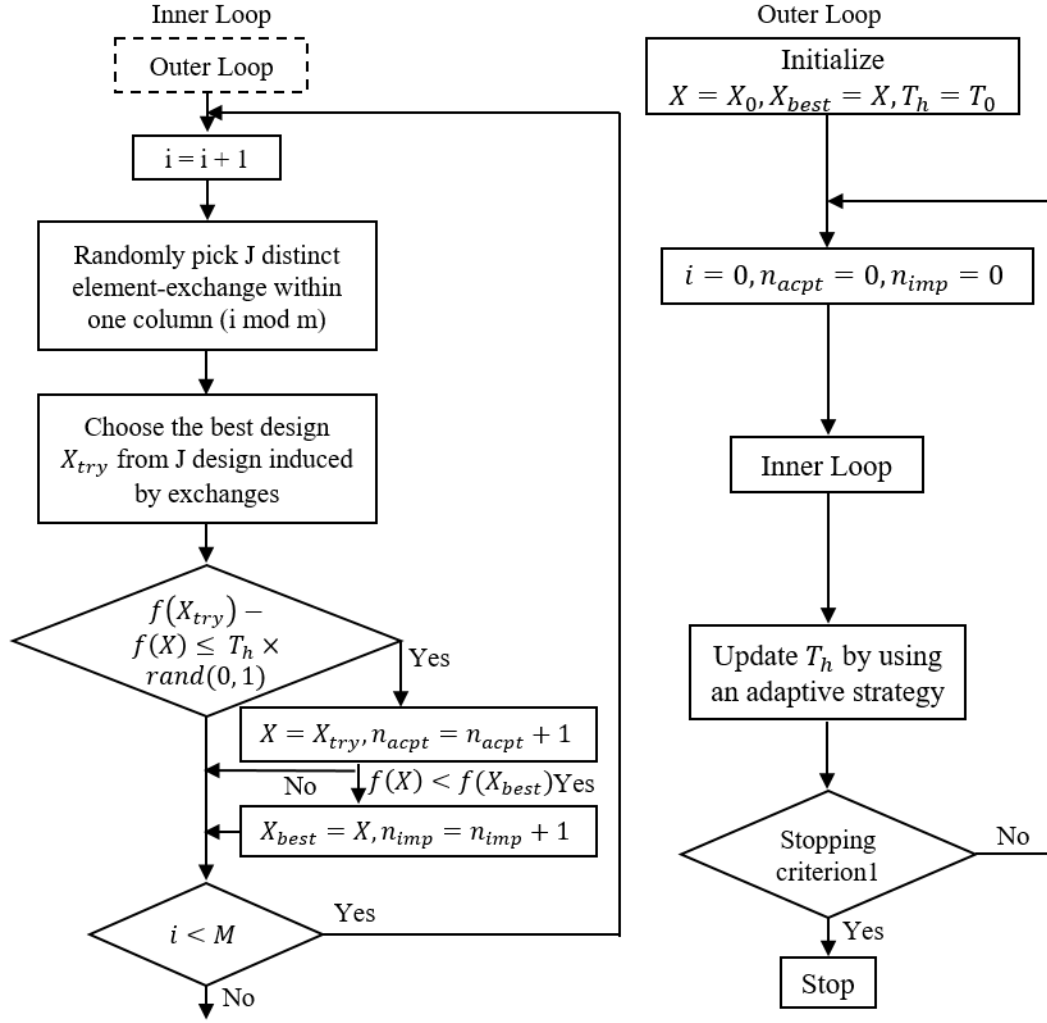


Figure 2.2. Flowchart of MESE algorithm [49].

2.2 Kriging surrogate models

After evaluating the objective function values for all the initially sampled points, the training set for fitting the initial surrogate model is obtained. The type of surrogate model used in this study is the kriging surrogate model. Kriging was first introduced by Sacks et al. [25] as a surrogate for expensive computational models. Among traditional surrogate models, kriging is widely used in engineering for its ability to predict unknown points and quantify uncertainty.

2.2.1 Ordinary kriging

Given n initial sampled points, each with n dimensions. the response of the (ordinary kriging) OK model can typically be expressed as follows:

$$\hat{y}(x) = \hat{\mu} + \hat{\epsilon}(x) \quad (3)$$

Here, $\hat{\mu}$ represents the mean term, which is determined by solving a weighted least squares problem:

$$\hat{\mu} = (\mathbf{F}^T \mathbf{R}^{-1} \mathbf{F})^{-1} \mathbf{F}^T \mathbf{R}^{-1} \mathbf{y} \quad (4)$$

Here, $\mathbf{F}$ is the regression matrix representing the zero-order basis function. In OK model, the polynomial vector of the basis function can be expressed as follows:

$$\mathbf{F} = \mathbf{1} = \underbrace{[1, \dots, 1]}_n^T \quad (5)$$

The correlation matrix $\mathbf{R}$ characterizes the spatial correlation among all training points:

$$\mathbf{R} = \begin{bmatrix} \psi(x^1, x^1) & \dots & \psi(x^1, x^n) \\ \vdots & \ddots & \vdots \\ \psi(x^n, x^1) & \dots & \psi(x^n, x^n) \end{bmatrix} \quad (6)$$

Here, the Gaussian function is selected as the modeling function for spatial correlation:

$$\psi(x^i, x^j) = \exp \left[\sum_{k=1}^n -\theta_k (\|x_k^i - x_k^j\|)^2 \right] \quad (7)$$

The θ_k is a critical hyper-parameter in kriging, and its value directly influences the quality of the kriging fit. As seen from the expression, θ_k scales the influence of the k_{th} design variable on the correlation between training points. Figure 2.3 illustrates the variation in correlation with respect to the distance between training points under different hyper-parameter values. The determination of hyper-parameters requires optimization using the maximum likelihood estimation (MLE) method to achieve the optimal fitting trend for the training points.

The error term $\hat{\epsilon}(x)$ in Eq. (3) is treated as a realization of a stochastic process and can be expressed as follows:

$$\hat{\epsilon}(x) = \mathbf{w}^T \mathbf{r}(x) \quad (8)$$

Here, the correlation vector $\mathbf{r}$ describes the spatial correlation between the training points and the prediction point:

$$\mathbf{r} = [\psi(x^{n+1}, x^1), \dots, \psi(x^{n+1}, x^n)] \quad (9)$$

The determination of $\mathbf{w}$ requires solving a system of linear equations:

$$\mathbf{w} = \mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\hat{\mu}) \quad (10)$$

The final kriging predictor can be expressed as follows:

$$\hat{y}(x^{n+1}) = \hat{\mu} + \mathbf{w}^T \mathbf{r} \quad (11)$$

The mean square error of the prediction made by the kriging predictor is given by:

$$s^2(x^{n+1}) = \hat{\sigma}^2 \left[1 - \mathbf{r}^T \mathbf{R}^{-1} \mathbf{r} + \frac{(1 - \mathbf{F}^T \mathbf{R}^{-1} \mathbf{r})^2}{\mathbf{F}^T \mathbf{R}^{-1} \mathbf{F}} \right] \quad (12)$$

Here, $\hat{\sigma}^2$ represents the predicted system variance:

$$\hat{\sigma}^2 = \frac{1}{n} (\mathbf{y} - \mathbf{F}\hat{\mu})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{F}\hat{\mu}) \quad (13)$$

The objective function values at the unknown points can be predicted by solving the linear equation systems (4), (10), (12), and (13).

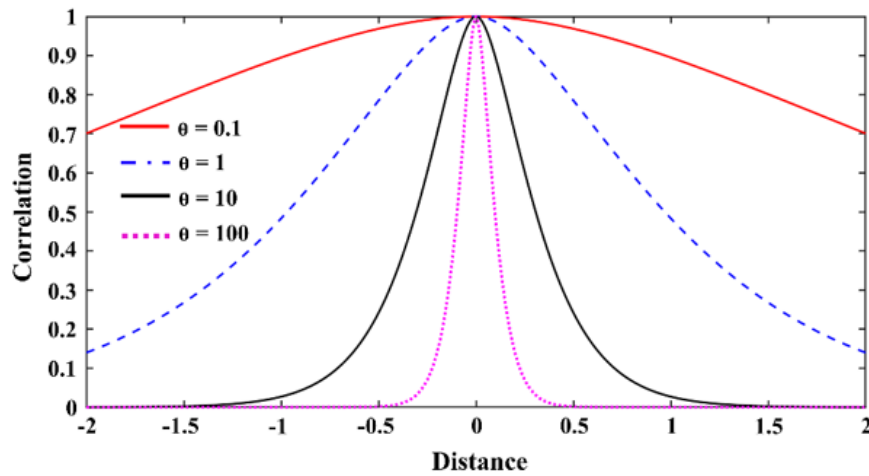


Figure 2.3. The variation trend of spatial correlation between sample points under different hyper-parameter values (one-dimensional correlation function).

2.2.2 Gradient-enhanced kriging

If the objective function values and sensitivity information of the corresponding points are available, incorporating gradient information into the construction of the kriging model can significantly improve the accuracy of fitting. Figure 2.4 presents an example of fitting the two-dimensional Griewank function using both OK and gradient-enhanced kriging (GEK), with the same number of training points. Clearly, the latter achieves higher fitting accuracy. GEK model is an extension of OK model that assumes a set of p initial sampled points, each with n dimensions. The objective value at an unknown point is predicted as follows:

$$\hat{y}(x_{p+1}) = \hat{\mu} + \mathbf{r}^T \mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\hat{\mu}) \quad (14)$$

where $\hat{\mu}$ represents the mean term, which is determined by solving the following weighted least-squares problem:

$$\hat{\mu} = (\mathbf{F}^T \mathbf{R}^{-1} \mathbf{F})^{-1} \mathbf{F}^T \mathbf{R}^{-1} \mathbf{y} \quad (15)$$

in which $\mathbf{F}$ is the regression matrix representing the zeroth-order basis functions. In GEK model, the polynomial vector of the basis function is expanded to include the derivatives of the regression basis functions. In the case of zeroth-order basis functions, a vector consist of 1s and 0s is given by:

$$\mathbf{F} = [\mathbf{1} \ \mathbf{0}] = \left[\underbrace{1, \dots, 1}_p, \underbrace{0, \dots, 0}_{p \times n} \right]^T \quad (16)$$

The correlation vector $\mathbf{r}$ represents the correlation between the sample points and the unknown prediction point, whereas the correlation matrix $\mathbf{R}$ represents the correlation between the sample points. As GEK model incorporates gradient information, the corresponding correlation vector and correlation matrix include gradient terms:

$$\mathbf{r} = \begin{bmatrix} \mathbf{q}^1 \\ \mathbf{q}^2 \end{bmatrix} \quad (17)$$

Here, $\mathbf{q}^1$ represents the correlation vector used in the OK model and $\mathbf{q}^2$ contains the first-order derivatives of $\mathbf{r}$:

$$\begin{cases} \mathbf{q}^1 = \begin{bmatrix} \psi(x^{n+1}, x^1) \\ \vdots \\ \psi(x^{n+1}, x^p) \end{bmatrix} \\ \mathbf{q}^2 = \begin{bmatrix} \frac{\partial \psi(x^{n+1}, x^1)}{\partial x^1} \\ \vdots \\ \frac{\partial \psi(x^{n+1}, x^p)}{\partial x^p} \end{bmatrix} \end{cases} \quad (18)$$

The Gaussian correlation function is selected:

$$\psi(x^i, x^j) = \exp \left[\sum_{k=1}^n -\theta_k (\|x_k^i - x_k^j\|)^2 \right] \quad (19)$$

where θ_k represents the hyper-parameter that controls the correlation variation along the k th dimension. It determines how the correlation between sample points changes with distance. Similarly, the correlation matrix is extended to include gradient terms:

$$\mathbf{R} = \begin{bmatrix} \mathbf{Q}^{1,1} & \mathbf{Q}^{1,2} \\ (\mathbf{Q}^{1,2})^T & \mathbf{Q}^{2,2} \end{bmatrix} \quad (20)$$

Here, $\mathbf{Q}^{1,1}$ represents the correlation matrix used in OK:

$$\mathbf{Q}^{1,1} = \begin{bmatrix} \psi(x^1, x^1) & \dots & \psi(x^1, x^p) \\ \vdots & \ddots & \vdots \\ \psi(x^p, x^1) & \dots & \psi(x^p, x^p) \end{bmatrix} \quad (21)$$

$\mathbf{Q}^{1,2}$ contains the first-order derivatives of $\mathbf{R}$:

$$\mathbf{Q}^{1,2} = \begin{bmatrix} \frac{\partial \psi(x^1, x^1)}{\partial x^1} & \dots & \frac{\partial \psi(x^1, x^p)}{\partial x^p} \\ \vdots & \ddots & \vdots \\ \frac{\partial \psi(x^p, x^1)}{\partial x^1} & \dots & \frac{\partial \psi(x^p, x^p)}{\partial x^p} \end{bmatrix} \quad (22)$$

and $\mathbf{Q}^{2,2}$ contains the second-order derivatives of $\mathbf{R}$:

$$\mathbf{Q}^{2,2} = \begin{bmatrix} \frac{\partial^2 \psi(x^1, x^1)}{\partial x^1 \partial x^1} & \dots & \frac{\partial^2 \psi(x^1, x^p)}{\partial x^1 \partial x^p} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \psi(x^p, x^1)}{\partial x^p \partial x^1} & \dots & \frac{\partial^2 \psi(x^p, x^p)}{\partial x^p \partial x^p} \end{bmatrix} \quad (23)$$

The correlation vector and matrix clearly show that GEK model incorporates gradient information, expanding their dimensions from $p \times 1$ and $p \times p$ to $p \times (p + 1)$ and $p \times (p + 1) \times p \times (p + 1)$, respectively. This expansion allows GEK model to better approximate steep or highly nonlinear regions and extract more useful information from limited training samples, leading to a more accurate representation of the design space and improved prediction accuracy. However, the increased correlation vector and matrix size also raise computational costs. For low-dimensional optimization problems, the cost of GEK model fitting is negligible compared to the function evaluation cost in actual optimization, giving it a clear advantage over OK model. As dimensionality increases, this advantage diminishes. The specific cost-growth trend with increasing dimensions requires further experimental analysis.

In OK model, the vector $\mathbf{y}$ contains the objective function values of all initial sample points, while in GEK model, the vector also includes the derivatives of the objective function with respect to the design variables:

$$\mathbf{y} = \begin{bmatrix} f(x^1) \\ \vdots \\ f(x^p) \\ \frac{\partial f(x^1)}{\partial x^1} \\ \vdots \\ \frac{\partial f(x^p)}{\partial x^p} \end{bmatrix} \quad (24)$$

The mean square error of the GEK model prediction can be expressed as:

$$s^2(x^{p+1}) = \hat{\sigma}^2 \left[1 - \mathbf{r}^T \mathbf{R}^{-1} \mathbf{r} + \frac{(1 - \mathbf{F}^T \mathbf{R}^{-1} \mathbf{r})^2}{\mathbf{F}^T \mathbf{R}^{-1} \mathbf{F}} \right] \quad (25)$$

The variance of the prediction system is given by:

$$\hat{\sigma}^2 = \frac{1}{n(1+p)} (\mathbf{y} - \mathbf{F}\boldsymbol{\mu})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{F}\boldsymbol{\mu}) \quad (26)$$

For detailed derivations regarding GEK model prediction, see Ollar et al. [50].

The objective function values at the unknown points can be predicted by solving the linear equation systems (10), (15), (25), and (26).

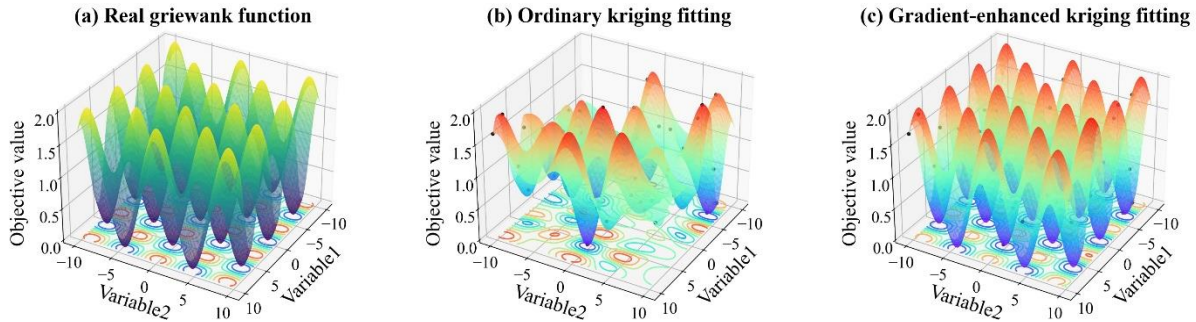


Figure 2.4. Comparison of fitting accuracy between gradient-enhanced kriging (GEK) and ordinary kriging (Ok). (a) Real two-dimensional Griewank function, (b) OK fitting, (c) GEK fitting.

2.3 Point Infills Criteriaions

After obtaining the initial kriging surrogate model from the initial training sample set, if the accuracy of the current surrogate model meets the predetermined requirements—meaning the optimal solution derived from the model satisfies the design needs—the entire optimization process terminates. However, for most engineering optimization problems, the quality of the optimal solution obtained from a surrogate model trained on a limited sample set often fails to meet the desired target. Therefore, to efficiently search for a high-quality optimal solution, it is essential to incorporate an infill criterion for the surrogate model.

2.3.1 Expected improvement

The kriging surrogate model provides predictions of the objective function values at unknown points and offers estimates of prediction uncertainty. Thus, it is appropriate to complement the kriging surrogate model with a point infill criterion. The process of predicting the objective function values at unknown points using kriging can be regarded as implementing a stochastic process that conforms to a normal distribution with a mean of $\hat{y}(x)$ and a variance of $s^2(x)$. Hence, a point infill criterion that accesses the potential expected improvement (EI) at unknown points is expressed as [35]:

$$EI(x) = (y_{min} - \hat{y}(x))\Phi\left(\frac{y_{min} - \hat{y}(x)}{s(x)}\right) + s(x)\phi\left(\frac{y_{min} - \hat{y}(x)}{s(x)}\right) \quad (27)$$

where $\Phi(x)$ and $\phi(x)$ represent the standard normal cumulative distribution function and probability density function, respectively. The EI function consists of two components: the first component exploits local optima when a point's predicted function value is lower than the current best, while the second promotes exploration of the design space when the prediction error is high. Therefore, the EI function itself maintains a balance between local exploitation and global exploration. Additionally, the EI function is equal to zero at the sample points, ensuring that sample points cannot be selected as potential improvement points. However, a significant limitation of the EI function is that it can only identify one candidate point within a single optimization cycle. This prevents it from fully utilizing external parallel computing resources to reduce the wall-clock time cost during the optimization process.

2.3.2 Pseudo expected improvement

To overcome the main limitation of EI function, the pseudo expected improvement (PEI) criterion proposes using an additional influence function to simulate the impact of the updated points on the behavior of the original EI function, thereby eliminating the costly evaluation of the update points and the kriging model update within a single optimization cycle. The expression for the PEI criterion is given by [39]:

$$PEI(x, q - 1) = EI(x) \times IF(x, x^{n+1}) \times \dots \times IF(x, x^{n+q-1}) \quad (28)$$

Where q represents the number of candidate points to be identified within a single optimization cycle. The IF function is used to simulate the impact of the updated points on the initial EI function and can be computed using the following expression:

$$IF(x, x^{n+1}) = 1 - Corr[Y(x), Y(x^{n+1})] \quad (29)$$

Here, x^{n+1} denotes the position of the candidate point, and $Corr[.]$ is the spatial correlation function used in the construction of kriging. The value of the PEI function at the update points is zero, and as the distance between the unknown point and the update point increases, the value of the PEI function gradually increases to one. This means that the EI function is significantly affected in the vicinity of the update point, while areas far from the update point are not noticeably impacted. This characteristic ensures that the variation of the PEI function across the entire search space closely approximates the influence of the update points on the initial EI function.

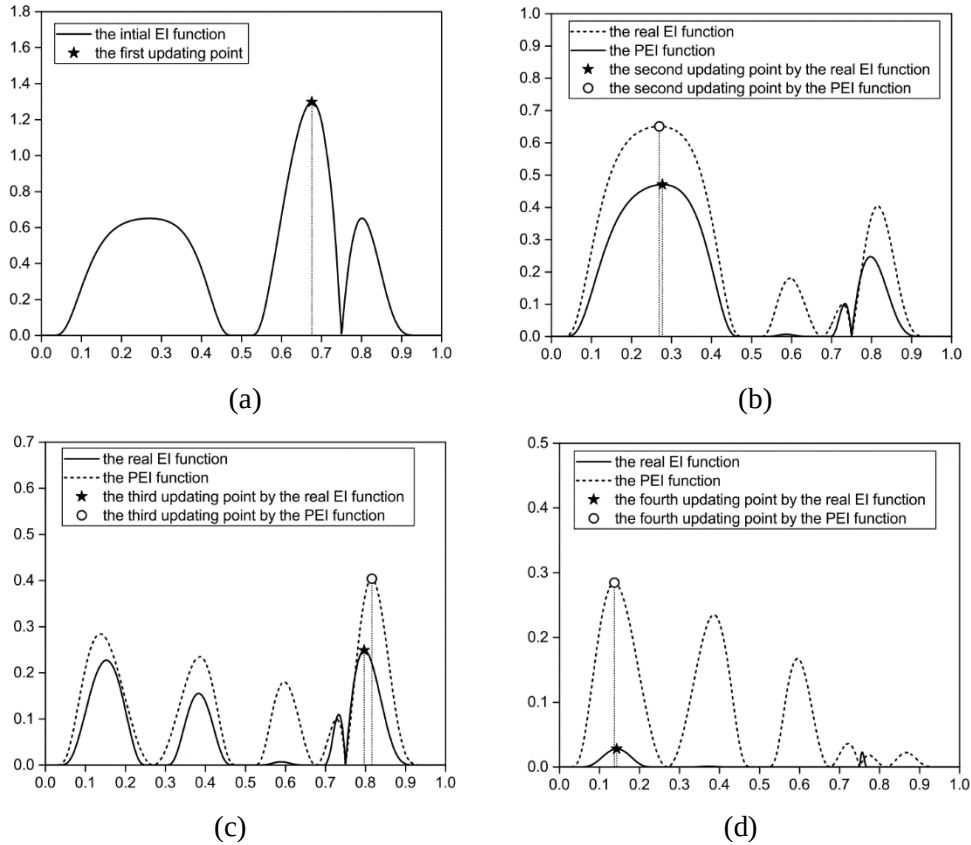


Figure 2.5. Comparison of searching new points by EI and PEI criterion, respectively. (a) The first iteration, (b) The second iteration, (c) The third iteration, (d) The forth iteration.

Figure 2.5 provides a case study to visually demonstrate the working principle of the PEI function. Figure 2.5a shows the initial visualization of the conventional EI function, where the optimal solution (maximum value) can be observed around 0.7. Therefore, the optimizer selects this point as a potential new point for improvement. The original non-parallel EGO algorithm first calls external computational resources to evaluate the objective function value at this point, incorporates the newly obtained sample point into the initial training dataset, and refits the Kriging model. The updated true EI function, derived from the new kriging model, is shown in Figure 5.2b. In contrast, the parallel EGO algorithm employs an influence function to simulate changes in the EI function without updating the Kriging model. After the update, we observe that the PEI function closely approximates the true EI function. This process iterates until all new candidate points in an optimization cycle are identified, after which all newly acquired points are collectively submitted to external computational resources for objective function evaluation. Consequently, this method significantly reduces the frequent invocation of external computational resources and improves the overall efficiency of the optimization process.

2.4 Kriging-based Efficient Global Optimization Algorithm Systems

The combination of the kriging surrogate model with the expected improvement sampling criterion constitutes an efficient global optimization (EGO) algorithm. Figure 2.6 presents the framework of the EGO algorithm systems based on the kriging surrogate models. First, a design of experiment method is used to generate the initial sample points for evaluation. Then, the computationally expensive experiments are conducted to evaluate the objective values of each sample point, forming the initial training dataset for fitting the kriging model. Next, an external optimizer is employed to optimize the hyper-parameters of the kriging model, thereby constructing the surrogate model. Subsequently, based on the current kriging model and the point infills criterion, the external optimizer is utilized to identify promising candidate point. The objective value of this candidate point is then re-evaluated. If the optimization of the objective values meets the predefined requirements, the optimization process terminates. Otherwise, the kriging model is updated, and the iterative optimization continues until the stopping criterion of the optimization process is satisfied.

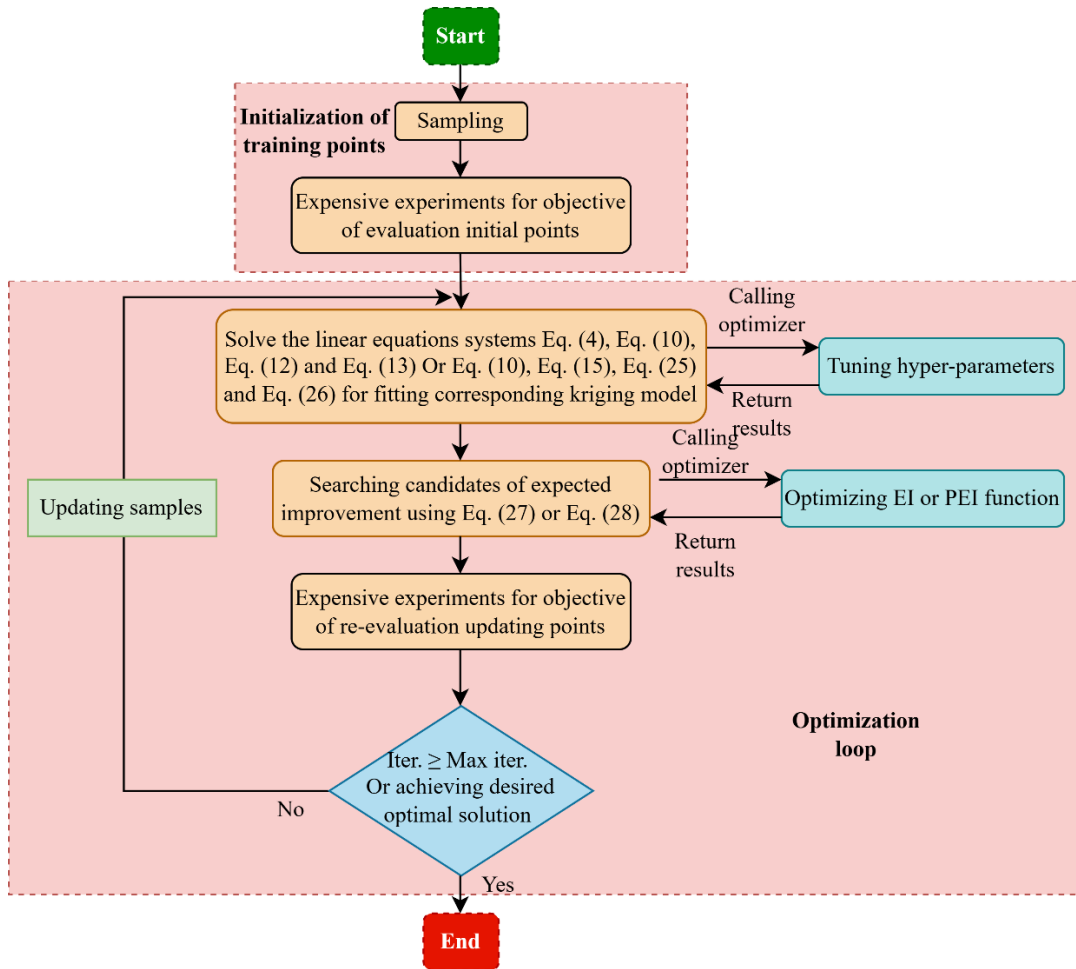


Figure 2.6. Framework of the kriging-based EGO algorithm systems.

2.5 Gradient-enhanced Kriging Based Parallel Efficient Global Optimization Algorithm System

To address complex engineering optimization problems and further improve optimization efficiency through making full use of external computational resources, an improved system, namely, GEK based parallel EGO, is proposed. Specifically, to explain the idea, a relatively high approximation quality of surrogate model can be easily obtained through the effective utilization of additional gradient information. In addition, the updating points identified by the PEI criterion provide more effective gradient information for surrogate model training, which helps reduce the total number of cycles required for inserting points. The methodology enables parallel implementations, facilitating the gradual updating of the surrogate model and guiding the optimization process toward true global convergence.

Figure 2.7 shows the framework of GEK based parallel EGO algorithm system. The algorithmic system primarily consists of two components: initializing training samples and performing optimization cycles. For initialization, an appropriate design of experiment is used to generate initial sample points, which are then evaluated through deterministic experiments to obtain objective function values. The resulting sample set is used for surrogate model training. During the optimization process, an external optimizer is used to optimize the hyper-parameters of the surrogate model to achieve the best possible fit. The external optimizer is then employed to optimize the parallel point-filling criterion (PEI function) closely related to the current surrogate model, identifying the locations of multiple updating sample points. The objective function values at the updating sample points are re-evaluated. The new sample set is incorporated into the existing sample set, forming a new training sample set. The process continues until the desired optimal solution is found. Table 2.1 presents the pseudocode for the GEK based parallel EGO algorithm.

As the algorithmic system involves the optimization of hyper-parameters for the kriging surrogate model and the parallel point-filling criterion (PEI function) for the updating sample points, the optimizer serves as a critical search engine in this process. Therefore, the optimization performance of the optimizer directly impacts the optimization performance of the algorithmic system. The following sections will discuss how the optimizer impacts the algorithmic system's performance, examine the effects of different optimizers, and identify the most robust optimizer for the system.

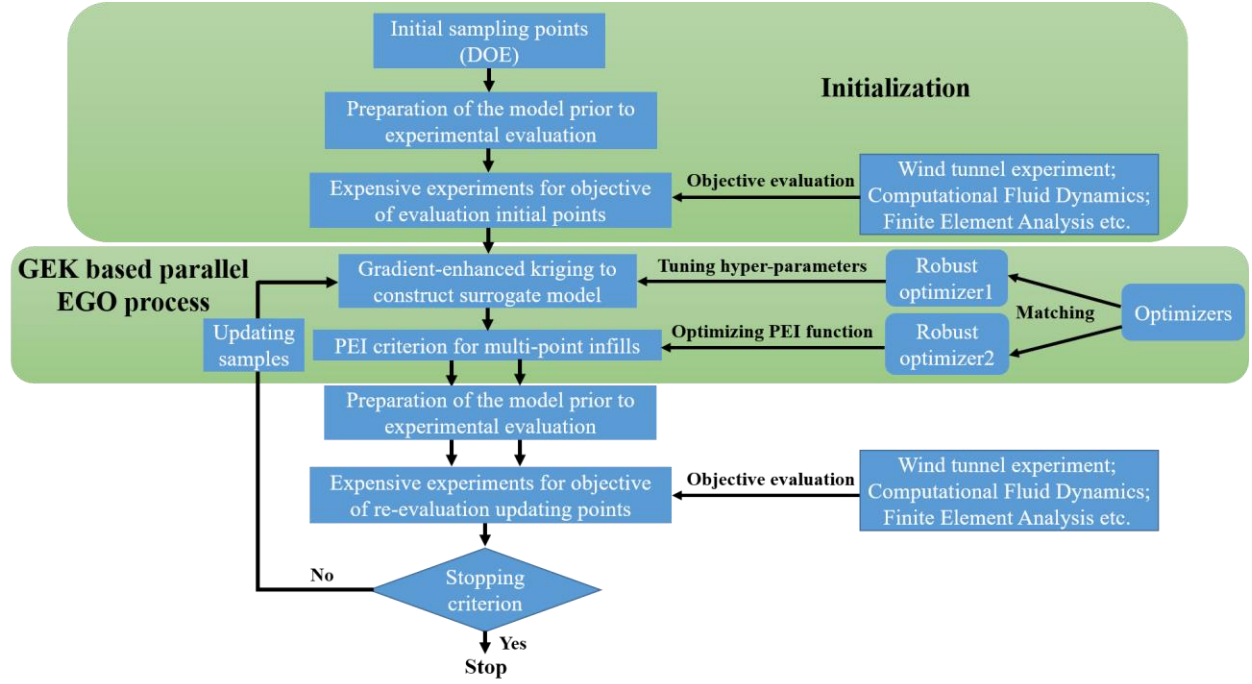


Figure 2.7. Framework of GEK based parallel EGO algorithm system.

Table 2.1. Pseudocode of GEK based parallel EGO algorithm.

Algorithm 1 The GEK based parallel EGO
Input: X (initial sample points), f (objective function), S (kriging model), q (number of parallel infills), C (stopping criterion)
Output: x _{opt} (optimized solution)
1: D ← InitializeDataset(X, f) 2: S ← TrainGEKSurrogateModel(D) 3: while not C do 4: PEI ← ConstructPEI(S) 5: X _{new} ← OptimizePEI(PEI, q) 6: Y _{new} ← EvaluateFunction(f, X _{new}) 7: D ← UpdateDataset(D, X _{new} , Y _{new}) 8: S ← RetrainGEKSurrogateModel(D) 9: end while 10: x _{opt} ← SelectBestSolution(D)

CHAPTER 3. INVESTIGATION OF THE INFLUENCE OF THE SPACE-FILLING QUALITY OF INITIAL SAMPLE POINTS ON THE PERFORMANCE OF THE PROPOSED OPTIMIZATION ALGORITHM SYSTEM

In this Chapter, we first classify the space-filling quality properties of evaluation points obtained through LHD and OLHD methods. Then conduct a preliminary investigation into the influence of various space-filling qualities of evaluation points on the performance of GEK based parallel EGO algorithm system.

3.1 Classification of Space-filling Quality

To investigate the influence of space-filling quality of sample points on the convergence performance of the proposed optimization algorithm system, its first necessary to classify the space-filling quality of sample points obtained through LHD and OLHD methods into reasonable levels. The classification is based on the differing ϕ_p values (Eq. (1)) of each group of sample points. Although LHD sampling exhibits specific randomness, statistical analysis reveals that using LHD sampling, regardless of the sample size, the quartile statistics and the maximum and minimum values of the ϕ_p values for the same sample size of LHD sampling tend to stabilize as the number of sampling repetitions increases. As shown in Figure 3.1, an optimization problem with a dimension (D) of 5 is selected here, including four LHD sizes ($2 \times D$, $4 \times D$, $6 \times D$, $8 \times D$, $10 \times D$). Therefore, based on this characteristic, the space-filling quality of LHD can be classified into four levels using the quartile values as the critical values. For the classification of OLHD sample points, the space-filling quality of the OLHD sample points is categorized based on the convergence state of the sample points optimization. Specifically, OLHD sample points are divided into two levels: one representing a completed convergence state (100 iterations in Figure 2.2) and the other representing an incomplete convergence state (10 iterations in Figure 2.2). As shown in Table 3.1, the specific classification of space-filling quality level is presented based on the results from 50,000 repeated samplings.

Table 3.1. Classification of space-filling quality for LHD and OLHD.

Level	Interval of ϕ_p value				
	2×5	4×5	6×5	8×5	10×5
OLHD-completed	0.5829~0.6319	0.7752~0.8326	0.9000~0.9400	0.9824~1.0309	1.0677~1.1108
OLHD-incomplete	0.6319~0.8412	0.8326~1.3588	0.9400~1.3569	1.0309~1.5626	1.1108~1.5826
LHD-Level1	0.8412~0.9200	1.3588~1.4623	1.3569~1.8378	1.5626~2.1669	1.5826~2.4500
LHD-Level2	0.9200~1.0140	1.4623~1.6060	1.8378~2.0727	2.1669~2.4376	2.4500~2.7222
LHD-Level3	1.0140~1.1407	1.6060~1.9000	2.0727~2.4167	2.4376~2.7875	2.7222~3.2667
LHD-Level4	1.1407~	1.9000~	2.4167~	2.7875~	3.2667~

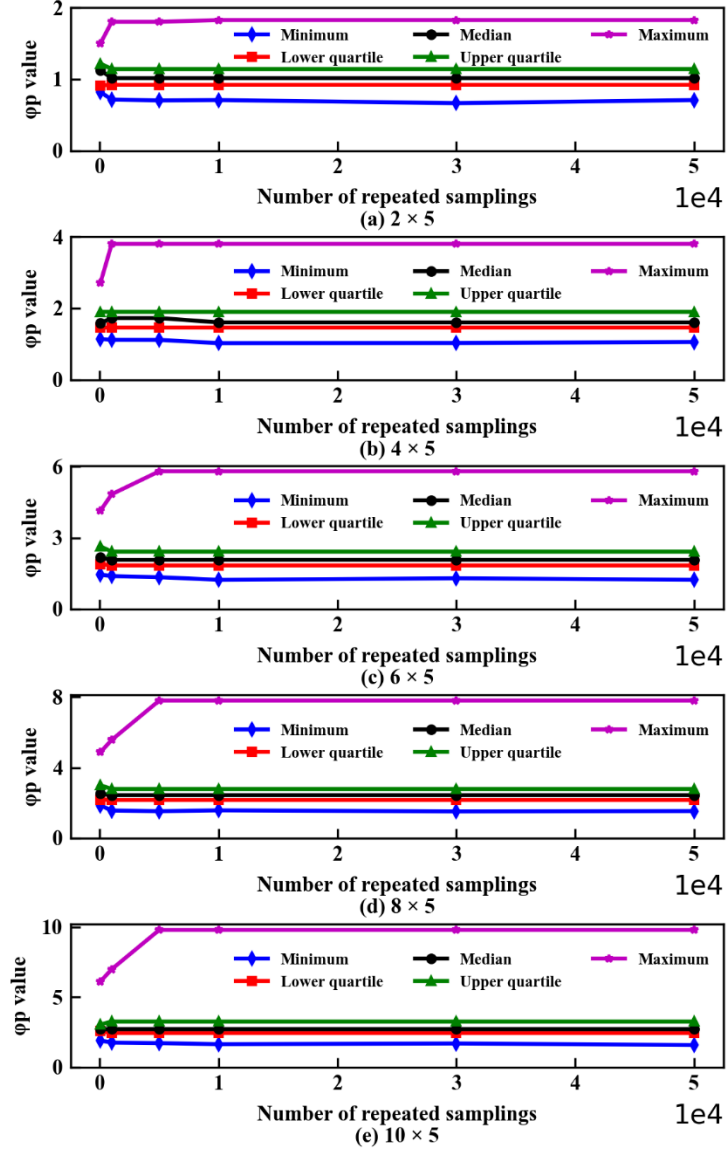


Figure 3.1. The trend of ϕ_p value quartiles as the number of sampling times increases.

3.2 Experimental Setting

To investigate the influence of space-filling quality of sample points on the performance of the proposed optimization algorithm system, 5 multimodal benchmark test functions with 5 dimensions (f_{10} , f_{11} , f_{12} , f_{14} , f_{22} in Table 5.4) from the CEC2013 test suite were selected as the evaluated functions [54]. During the optimization process, the parameter settings of the optimization algorithm system were kept consistent with those described in Chapter 5. Additionally, to ensure a fair comparison, 50 group of sample points were randomly selected from the 50,000 repeated samplings for each space-filling quality level. The optimization process was then repeatedly run 50 times for each benchmark test function.

3.3 Results and Discussion

Nonparametric statistical test [58] is used to analyze the significant differences between the comparative results. The Wilcoxon signed-rank test is employed to examine significant differences in the convergence results between two groups result on each test function, with a significance level set at 0.05. If the calculated p -value is less than 0.05, there is a significant difference in performance between the two groups result. Conversely, there is no significant difference in performance between two groups result. the Friedman test was used to calculate the mean rankings of each space-filling quality level for all test functions.

Table 3.2 presents the Wilcoxon signed-rank test results for the sample size of 10×5 compared to four other sample sizes under each space-filling quality level and each benchmark function. The symbol “=” indicates that there is no significant difference in performance between two compared sample sizes on the optimization system. The symbols “+” and “-” indicate that the first sample size shows better or worse performance on the optimization system compared with the second sample size, respectively. Furthermore, although p -values between 0.05 and 0.10 for the significance level cannot reflect significant difference in the performance of the two compared sample sizes on the optimization system, the symbols “= ($p < 0.1$)” and ““= ($p < 0.1$)” are still included in the table to show that the first sample size on the optimization system potentially outperforms or underperforms the second sample size, respectively. The comparison between the two sample sizes of 10×5 and 2×5 clearly demonstrates that the former shows significantly better convergence performance than the latter, as evidenced by the predominance of “+” symbols and “=” symbols in the table. However, as the sample size increases, the effective information of the design space brought by the samples has a diminishing impact on improving the convergence performance of the current optimization system, eventually stabilizing at a plateau.

Table 3.2. Wilcoxon signed-rank test between sample size of 10×5 and other sample size, respectively, under each space-filling quality level.

Benchmark function	Space-filling quality (2 × 5)					
	OLHD-completed	OLHD-incomplete	LHD-Level1	LHD-Level2	LHD-Level3	LHD-Level4
<i>f</i> 10	=	=	=	=	=	=
<i>f</i> 11	= (p/< 0.1)	+	=	=	+	=
<i>f</i> 12	+	=	+	=	+	+
<i>f</i> 14	+	+	+	+	+	=
<i>f</i> 22	+	=	=	=	+	= (p/< 0.1)
	Space-filling quality (4 × 5)					
	OLHD-completed	OLHD-incomplete	LHD-Level1	LHD-Level2	LHD-Level3	LHD-Level4
<i>f</i> 10	=	–	=	=	=	=
<i>f</i> 11	=	= (p/< 0.1)	=	=	= (p/< 0.1)	=
<i>f</i> 12	=	=	=	=	=	+
<i>f</i> 14	=	=	=	=	=	=
<i>f</i> 22	+	=	=	=	=	+
	Space-filling quality (6 × 5)					
	OLHD-completed	OLHD-incomplete	LHD-Level1	LHD-Level2	LHD-Level3	LHD-Level4
<i>f</i> 10	=	=	=	=	=	=
<i>f</i> 11	=	+	=	=	=	=
<i>f</i> 12	=	+	=	=	=	= (p/< 0.1)
<i>f</i> 14	=	+	=	= (p/< 0.1)	= (p/< 0.1)	=
<i>f</i> 22	+	+	=	=	=	=
	Space-filling quality (8 × 5)					
	OLHD-completed	OLHD-incomplete	LHD-Level1	LHD-Level2	LHD-Level3	LHD-Level4
<i>f</i> 10	=	=	=	=	=	=
<i>f</i> 11	=	=	=	=	=	=
<i>f</i> 12	=	=	= (p/< 0.1)	=	=	=
<i>f</i> 14	=	=	=	=	=	=
<i>f</i> 22	=	=	=	=	=	=

To comprehensively analyze the influence of different space-filling quality levels on the convergence performance of the current optimization system under the same sample size, Figure 3.2 shows the results of the Friedman ranking test on each sample size and benchmark function. Based on the ranking results, while the number of sample points significantly affects the convergence performance of the optimization system, the impact of different space-filling qualities of sample point designs, given the same number of sample points, is less pronounced in terms of convergence performance. However, the statistical results indicate that the OLHD-completed level, which has the highest space-filling quality and requires the most computational resources to ensure uniform distribution of sample points, does not necessarily lead to optimal convergence performance in the current optimization system. Specifically, the OLHD-incomplete level, which has slightly lower space-filling quality, significantly reduces computation time during

optimization and demonstrates superior convergence performance across most sample sizes and test functions (with lower ranks indicating better performance), except for its poorer performance at a sample size of 6×5 . Additionally, the various space-filling quality levels of the standard LHD sample point design, which requires the least computational resources but has a higher degree of randomness, also exhibit greater variability in their impact on the convergence performance of the optimization system.

To specifically evaluate which level of space-filling quality of sample points can more potential promote the current optimization system to converge to higher quality solutions within a limited number of iterations under each sample size, Table 3.3 presents the mean rankings of different space-filling quality levels with each sample size on all benchmark tested functions. It can be observed that the space-filling quality level of OLHD-incomplete achieved higher mean rankings with sample sizes of 2×5 , 4×5 , 8×5 and 10×5 on all tested functions, except with the sample size of 6×5 . In summary, based on the current test results and statistical analysis, the OLHD-incomplete sample point design, which has a good balance of space-filling quality and appropriate randomness, appears to better balance between local exploitation and global exploration within the current optimization system. This balance, in turn, facilitates the system's ability to converge more quickly to higher-quality solutions within a limited number of iterations.

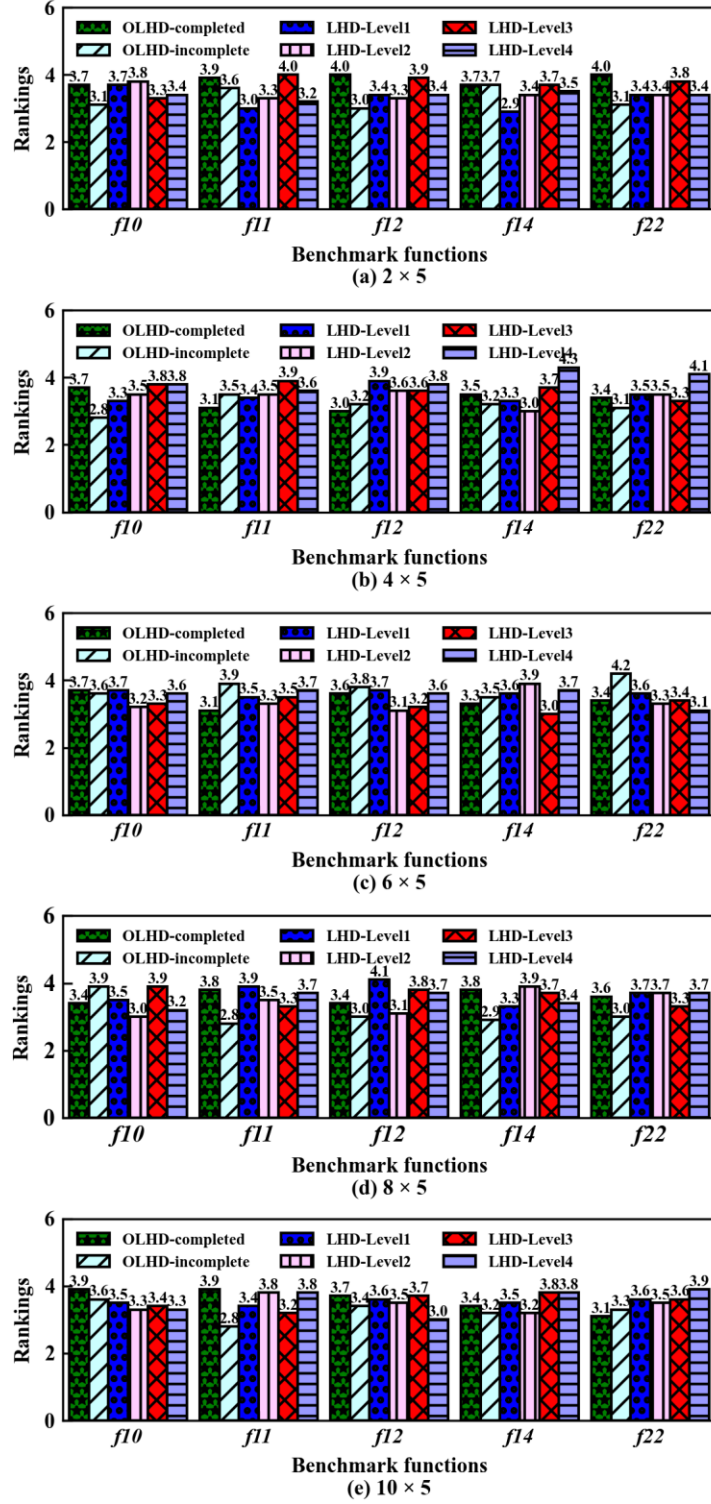


Figure 3.2. Rankings of different space-filling quality level with different sample sizes for each benchmark test function.

Table 3.3. Mean rankings of space-filling quality levels with each sample size for all benchmark test functions.

Samples size	Ranking	Space-filling quality	Mean ranking	<i>p</i> value
2×5	1	LHD-Level1	2.20	0.0671
	2	OLHD-incomplete	2.40	
	3	LHD-Level4	3.00	
	4	LHD-Level2	3.40	
	5	LHD-Level3	4.80	
	6	OLHD-completed	5.20	
4×5	1	OLHD-incomplete	1.80	0.0252
	2	OLHD-completed	2.60	
	3	LHD-Level2	3.00	
	4	LHD-Level1	3.60	
	5	LHD-Level3	4.40	
	6	LHD-Level4	5.60	
6×5	1	LHD-Level2	2.40	0.1422
	2	LHD-Level3	2.60	
	3	OLHD-completed	2.80	
	4	LHD-Level4	3.60	
	5	LHD-Level1	4.60	
	6	OLHD-incomplete	5.00	
8×5	1	OLHD-incomplete	1.80	0.2284
	2	LHD-Level2	3.20	
	3	LHD-Level4	3.60	
	4	LHD-Level3	3.80	
	4	OLHD-completed	3.80	
	6	LHD-Level1	4.80	
10×5	1	OLHD-incomplete	2.20	0.4053
	2	LHD-Level2	2.60	
	3	LHD-Level1	3.80	
	4	LHD-Level4	4.00	
	5	LHD-Level3	4.20	
	5	OLHD-completed	4.20	

CHAPTER 4. INVESTIGATION OF THE INFLUENCE OF THE PARALLEL POINT INFILLS STRATEGY ON THE PERFORMANCE OF THE PROPOSED OPTIMIZATION ALGORITHM SYSTEM

This Chapter aims to enhance the practical applicability and further development of the parallel EGO algorithm based on the PEI criterion. A comparative analysis was conducted on nine analytical functions with different characteristics to investigate the impact of the number of candidate points on the performance of the PEI-based parallel EGO algorithm using both the OK and GEK models. A systematic and in-depth evaluation was performed from multiple perspectives, including global optimization capability, convergence speed, optimization efficiency in achieving the desired convergence criteria, algorithmic diversity, and the balance between global exploration and local exploitation.

4.1 Experimental Setting

To comprehensively investigate the impact of the number of infill points on the performance of the kriging-based parallel EGO algorithm system, a comparative experimental study was conducted under different levels of infill points. The optimization performance of the parallel EGO algorithm system based on both the OK model and the GEK model was systematically evaluated. Table 4.1 presents nine benchmark test functions, selected from a benchmark test function suit specifically designed for evaluating the performance of optimization algorithms [54]. The tested functions all have a dimensionality of five. To reduce the impact of randomness in the initial points and stochastic parameters in the optimization process, the tested experiments were repeated 50 times using 50 sets of 40 initial points generated by OLHD method [49]. In each case, the parallel EGO algorithm based on OK and GEK was allowed to perform an additional 240 and 120 evaluations, respectively, on the tested function. In each optimization cycle, the parallel EGO algorithm generates 2, 4, 6, 8, and 10 candidate points, respectively. This means that the parallel EGO algorithm based on OK model has a maximum of 120 optimization cycles, while the parallel EGO algorithm based on GEK model has a maximum of 60 optimization cycles.

In the numerical simulation tests, a Python toolbox, named VAEOPY (Vehicular Aerodynamic Engineering Optimization in Python) [56], was used to fit both OK and GEK models and to perform optimization using the parallel EGO algorithm. The corresponding parameter settings are as follows:

- Regression function: *regpoly0*
- Correlation function: *corrGauss*.
- Feasible domain of hyper-parameters: $[10^{-5}, 10^5]$
- Optimizer for tuning hyper-parameters: GS-SQP, maximum number of iterations: 1000

- Optimizer for searching candidate points: MGBDE, population size: 50, maximum number of iterations: 500

Table 4.1. Details of benchmark test functions.

Benchmark test functions	Design spaces	Optimum
$F1(x) = \sum_{i=1}^{D-1} (1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2$	$x \in [-5, 10]^D$	0
$F2(x) = 1 + \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right)$	$x \in [-50, 50]^D$	0
$F3(x) = 1 + \sum_{i=1}^D \frac{z_i^2}{4000} - \prod_{i=1}^D \cos\left(\frac{z_i}{\sqrt{i}}\right) - 500, z = A^{100} M_1 \frac{600(x - o)}{100}$	$x \in [-100, 100]^D$	-500
$F4(x) = \sum_{i=1}^D (z_i^2 - 10 \cos(2\pi z_i) + 10) - 400, z = A^{10} T_{asy}^{0.2}(T_{osz}(\frac{5.12(x - o)}{100}))$	$x \in [-100, 100]^D$	-400
$F5(x) = \sum_{i=1}^D (z_i^2 - 10 \cos(2\pi z_i) + 10) - 300,$ $z = M_1 A^{10} M_2 T_{asy}^{0.2}(T_{osz}(M_1 \frac{5.12(x - o)}{100}))$	$x \in [-100, 100]^D$	-300
$F6(x) = 418.9829 \times D - \sum_{i=1}^D g(z_i) - 100,$ $z = A^{10} \left(\frac{1000(x - o)}{100} \right) + 4.209687462275036e + 002,$ $g(z_i) = \begin{cases} z_i \sin(z_i ^{\frac{1}{2}}), & \text{if } z_i \leq 500 \\ (500 - \text{mod}(z_i, 500)) \sin(\sqrt{ 500 - \text{mod}(z_i, 500) }) - \frac{(z_i - 500)^2}{10000D}, & \text{if } z_i > 500 \\ (\text{mod}(z_i , 500) - 500) \sin(\sqrt{ \text{mod}(z_i , 500) - 500 }) - \frac{(z_i + 500)^2}{10000D}, & \text{if } z_i < -500 \end{cases}$	$x \in [-100, 100]^D$	-100
$F7(x) = 418.9829 \times D - \sum_{i=1}^D g(z_i) + 100,$ $z = A^{10} M_1 \left(\frac{1000(x - o)}{100} \right) + 4.209687462275036e + 002,$ $g(z_i) = \begin{cases} z_i \sin(z_i ^{\frac{1}{2}}), & \text{if } z_i \leq 500 \\ (500 - \text{mod}(z_i, 500)) \sin(\sqrt{ 500 - \text{mod}(z_i, 500) }) - \frac{(z_i - 500)^2}{10000D}, & \text{if } z_i > 500 \\ (\text{mod}(z_i , 500) - 500) \sin(\sqrt{ \text{mod}(z_i , 500) - 500 }) - \frac{(z_i + 500)^2}{10000D}, & \text{if } z_i < -500 \end{cases}$	$x \in [-100, 100]^D$	100
$F8(x) = g_1(g_2(z_1, z_2)) + g_1(g_2(z_2, z_3)) + \dots + g_1(g_2(z_{D-1}, z_D)) + g_1(g_2(z_D, z_1)) + 500,$ $g_1(x) = 1 + \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right), g_2(x) = \sum_{i=1}^{D-1} (1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2,$ $z = M_1 \frac{5(x - o)}{100} + 1$	$x \in [-100, 100]^D$	500
$F9(x) = \sum_{i=1}^n \{\omega_i [\lambda_i g_i(x) + \text{bias}_i]\} + 800, n = 3, \omega_i = \frac{w_i}{\sum_{i=1}^n w_i},$ $w_i = \frac{1}{\sqrt{\sum_{j=1}^D (x_j - o_{ij})^2}} \exp\left(-\frac{\sum_{j=1}^D (x_j - o_{ij})^2}{2D\sigma_i^2}\right), \sigma = [20, 20, 20], \lambda = [1, 1, 1],$ $\text{bias} = [0, 100, 200], g_{1-3}: \text{Schwefel's function}$	$x \in [-100, 100]^D$	800

4.2 Results and Discussion

4.2.1 Influence of the parallel point infills strategy on the solution quality

A comparative analysis was conducted to examine the impact of different point-filling quantities on the optimization solution accuracy of the parallel EGO algorithm based on OK model and GEK model within a limited total number of evaluations. Table 4.2 presents the statistical results of the optimized solutions for all test cases, along with the Friedman ranking across the test cases. The results indicate that, for the case based on OK model, filling 2 update points per optimization cycle achieved the most accurate converged solutions for test functions F1, F4, F6, F8, and F9. Additionally, for F5, it also obtained the second-highest ranked approximate optimal solution in terms of accuracy. In contrast, as the number of filled points increased to 10, the algorithm yielded the least accurate converged solutions for almost all test functions, except for F3 and F6. However, overall, as the number of filled points increased from 2 to 10, the change in the accuracy of the converged solutions did not follow a strictly linear decline; instead, there were fluctuations within certain intervals. Meanwhile, it was observed that the differences in the accuracy of the converged solutions obtained with different numbers of filled points were relatively significant. This suggests that the stability of the optimization performance of the parallel EGO algorithm based on OK model deteriorates as the number of filled points increases.

For the case based on GEK model, the optimization performance advantage of the strategy with 2 filled points has decreased. It only achieved the best convergence accuracy on test functions F1, F3, and F5, while performing relatively poorly on several other test functions. Particularly, the convergence accuracy obtained on F2, F6, and F7 was the worst. Although the strategy with a point filling number of 10 still performs poorly on most test functions compared to other filling strategies, the overall accuracy of its convergence solutions has improved, with a reduction in the differences. Additionally, except for F1 and F3, the variation in the accuracy of the convergence solutions does not exhibit a linear decrease as the point filling number increases. It is worth noting that, based on the statistical decision metric p -value, the difference in the accuracy of the converged solutions has significantly reduced, compared to the case based on OK model, except for F1 and F3. This indicates that the stability of the optimization performance of the parallel EGO algorithm based on GEK model has been better ensured.

Table 4.2. Statistical results of the converged solutions obtained by the parallel EGO algorithms based on OK and GEK models under different point-filling strategies.

Systems	Test problems	Metrics	2 points	4 points	6 points	8 points	10 points
OK-based	F1	Mean	4.5089e-01	5.6691e-01	6.0459e-01	6.9066e-01	7.7906e-01
		Std	7.8648e-01	6.8798e-01	4.5595e-01	6.8258e-01	5.6624e-01
		Rank ($p=0.0019$)	2.00	3.03	3.30	3.10	3.57
Mean		4.4899e-02	9.1790e-02	1.2199e-01	1.4146e-01	2.7191e-01	
Std		3.7027e-02	7.0604e-02	1.1719e-01	9.5511e-02	3.5650e-01	
Rank ($p<0.0001$)		1.87	2.57	3.03	3.53	4.00	
OK-based	F2	Mean	4.6415e-01	4.9670e-01	4.5509e-01	4.6978e-01	5.0964e-01
		Std	1.1528e-01	1.1181e-01	8.5571e-02	1.1512e-01	1.0584e-01
		Rank ($p=0.2998$)	2.97	3.20	2.57	2.87	3.40
Mean		5.4790e-01	5.3746e-01	5.5005e-01	5.3116e-01	5.6354e-01	
Std		1.2902e-01	1.3038e-01	1.1066e-01	1.0940e-01	1.3511e-01	
Rank ($p=0.3578$)		3.03	2.97	2.90	2.63	3.47	
OK-based	F3	Mean	-	-	-	-	-
		Std	4.9934e+02	4.9936e+02	4.9940e+02	4.9936e+02	4.9938e+02
		Rank ($p=0.6387$)	3.27	3.03	2.73	3.17	2.80
Mean		-	-	-	-	-	
Std		4.9518e+02	4.9483e+02	4.9400e+02	4.9388e+02	4.9359e+02	
Rank ($p=0.0316$)		2.43	2.60	3.20	3.20	3.57	
OK-based	F4	Mean	-	-	-	-	-
		Std	3.9797e+02	3.9754e+02	3.9730e+02	3.9694e+02	3.9674e+02
		Rank ($p=0.0023$)	2.23	2.87	2.83	3.23	3.83
Mean		-	-	-	-	-	
Std		3.9818e+02	3.9840e+02	3.9817e+02	3.9807e+02	3.9806e+02	
Rank ($p=0.0316$)		3.20	2.20	2.97	3.30	3.33	
OK-based	F5	Mean	-	-	-	-	-
		Std	2.9684e+02	2.9566e+02	2.9585e+02	2.9730e+02	2.9666e+02
		Rank ($p=0.2073$)	2.67	3.10	3.47	2.63	3.13
Mean		-	-	-	-	-	
Std		2.9765e+02	2.9735e+02	2.9726e+02	2.9749e+02	2.9765e+02	
Rank ($p=0.3173$)		2.63	3.13	3.40	3.10	2.73	

Table 4.2. Continuous.

Systems	Test problems	Metrics	2 points	4 points	6 points	8 points	10 points
OK-based	F6	Mean	-	-	-	-	-
			6.4438e+01	1.8168e+01	4.4596e+01	2.0781e+01	1.9443e+01
		Std	6.3824e+01	1.0328e+02	1.3318e+02	9.5127e+01	1.1667e+02
GEK-based	F6	Rank ($p=0.0086$)	2.50	3.40	2.37	3.53	3.20
		Mean	-	-	-	-	-
			8.7698e+01	9.3243e+01	8.8515e+01	8.0994e+01	9.4264e+01
OK-based	F7	Std	3.8849e+01	2.1060e+01	2.9392e+01	4.9746e+01	8.0437e+00
		Rank ($p=0.6868$)	3.30	2.80	3.13	2.80	2.97
		Mean	2.0104e+02	1.7902e+02	1.8019e+02	1.7962e+02	1.7694e+02
GEK-based	F7	Std	1.4379e+02	1.0624e+02	1.0799e+02	1.0126e+02	1.0803e+02
		Rank ($p=0.8136$)	3.13	2.73	2.97	2.97	3.20
		Mean	1.3460e+02	1.4083e+02	1.3083e+02	1.3919e+02	1.3518e+02
OK-based	F8	Std	6.3485e+01	6.9991e+01	5.5263e+01	6.9680e+01	6.0899e+01
		Rank ($p=0.9197$)	3.23	2.93	2.97	2.87	3.00
		Mean	5.0284e+02	5.0311e+02	5.0334e+02	5.0351e+02	5.0351e+02
GEK-based	F8	Std	6.6693e-01	8.2985e-01	9.7198e-01	9.6062e-01	1.3396e+00
		Rank ($p=0.0047$)	2.17	2.57	3.40	3.37	3.50
		Mean	5.0500e+02	5.0480e+02	5.0476e+02	5.0566e+02	5.0515e+02
OK-based	F9	Std	1.7700e+00	1.3614e+00	1.3429e+00	2.0348e+00	1.3429e+00
		Rank ($p=0.6828$)	2.90	2.63	2.83	3.33	3.20
		Mean	9.7651e+02	1.0291e+03	1.0683e+03	1.0344e+03	1.0929e+03
GEK-based	F9	Std	1.6718e+02	1.3887e+02	1.3285e+02	1.5219e+02	1.5950e+02
		Rank ($p=0.0071$)	2.20	2.83	3.47	2.97	3.53
		Mean	9.5022e+02	9.5390e+02	9.6924e+02	9.3789e+02	9.4671e+02
OK-based	F9	Std	8.0413e+01	8.7164e+01	1.0428e+02	7.2873e+01	8.2066e+01
		Rank ($p=0.8088$)	2.97	3.17	3.20	2.90	2.77
		Mean					

4.2.2 Influence of the parallel point infills strategy on the convergence speed

To investigate the impact of different point filling quantities on the convergence speed of the parallel EGO algorithm, Figures 4.1 and 4.2 present the average convergence curves of the parallel EGO algorithm based on OK and GEK models, respectively, under different point filling strategies.

From Figure 4.1, it can be observed that for the case based on OK model, the convergence speed varies significantly across different point filling quantities for almost all test functions, except for F3. A point filling quantity of 2 exhibits an absolute performance advantage in terms of convergence speed for test

functions F1, F4, F6, F8, and F9. This conclusion can be drawn from the fact that the average convergence curve for a point filling quantity of 2 remains consistently below the other average convergence curves throughout the optimization process. In contrast, the average convergence speed for a point filling quantity of 10 is relatively slower across most test functions. The average convergence speed for other point filling quantities exhibits a certain degree of fluctuation between these two cases. This result is generally consistent with the findings on the accuracy of the optimized solutions presented above. This indicates that the optimization performance of the parallel EGO algorithm based on OK model is most reliable when the point filling quantity is set to 2. However, as the point filling quantity gradually increases to 10, the stability of the parallel EGO algorithm's optimization performance deteriorates significantly.

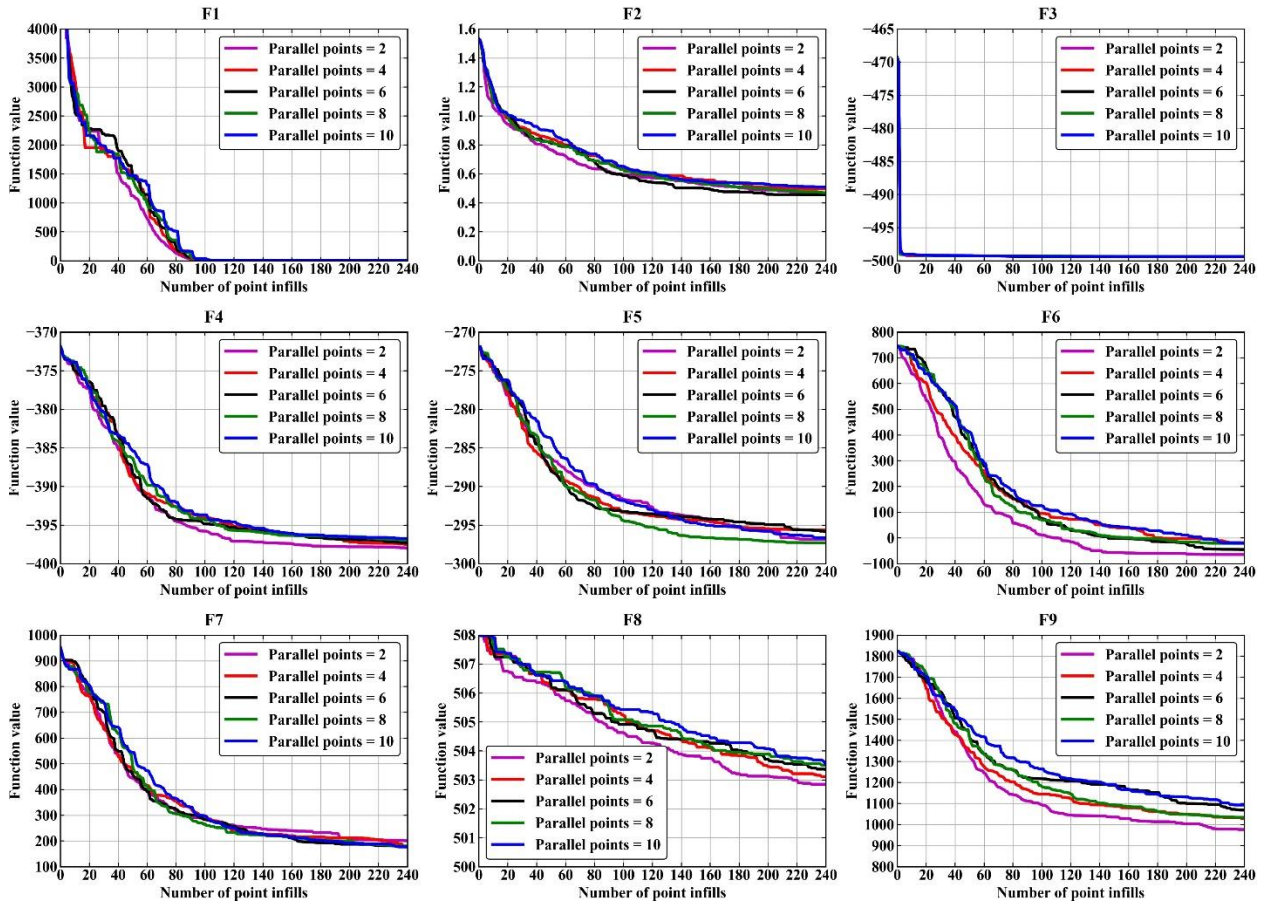


Figure 4.1. Average convergence curves of the OK-based parallel EGO algorithm on all tested problems.

From the average convergence curves in Figure 4.2, it can be observed that in the case of GEK model, the impact of different point filling quantities on the convergence speed of the parallel EGO algorithm differs only in the early stages of the optimization process for almost all test functions, except for F3 and F8. Moreover, during this stage, the impact on the average convergence speed exhibits a trend that is

essentially linearly correlated with the increase in point filling quantity. As the optimization process progresses, the differences in average convergence speed among different point filling quantities gradually diminish. The potential reason for this phenomenon is that, in the early stages of the optimization process, fewer point filling quantities lead to more frequent updates of the surrogate model. This provides more reliable design space information for the EGO algorithm's optimization. As the optimization process progresses and considering that GEK model has superior fitting performance compared to OK model, it can rapidly compensate for the loss of design space information due to fewer surrogate model updates in the early stages of the optimization process. This result also indicates that the convergence performance of the parallel EGO algorithm based on GEK model is more stable and reliable.

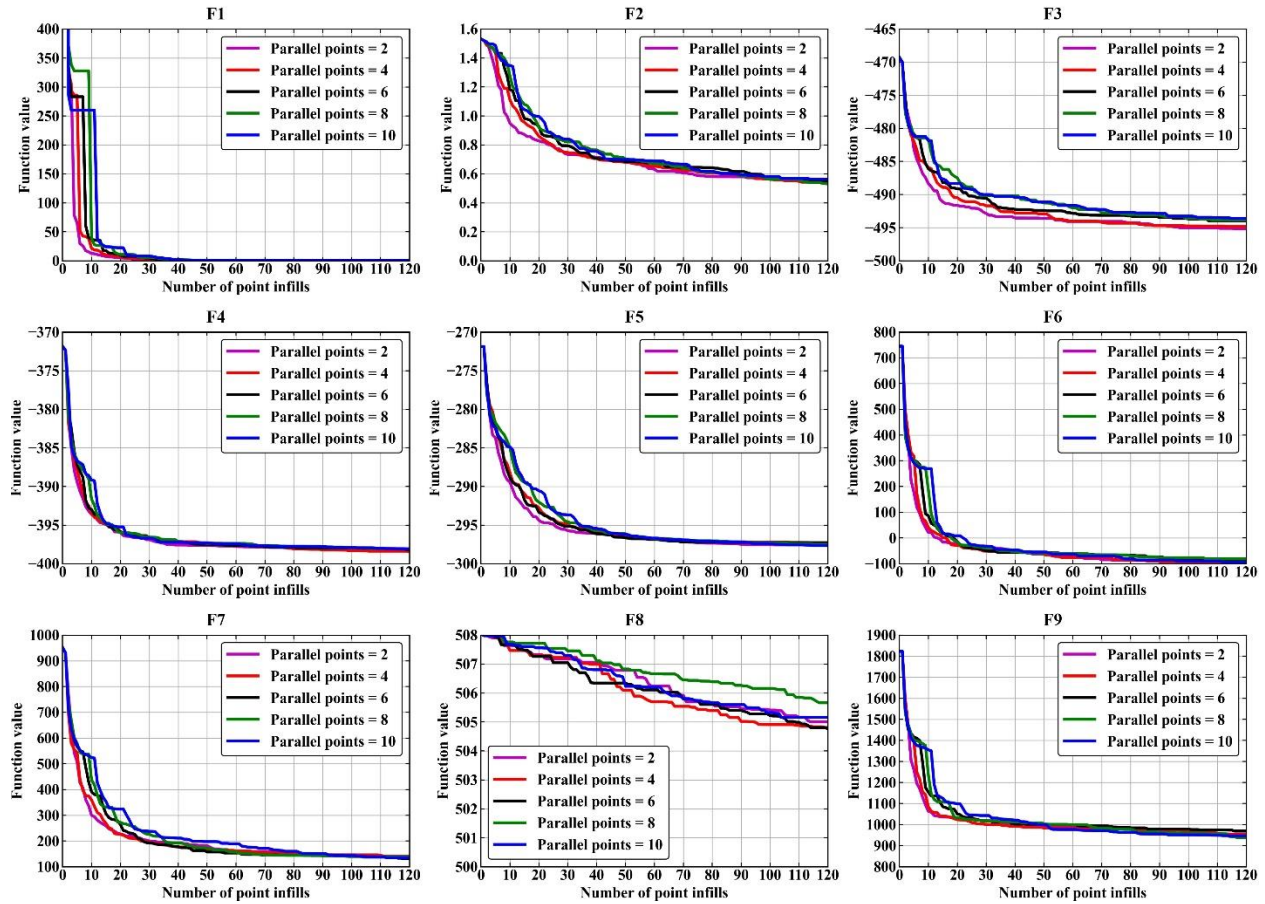


Figure 4.2. Average convergence curves of the GEK-based parallel EGO algorithm on all tested problems.

4.2.3 Influence of the parallel point infills strategy on the optimization efficiency

In practical performance optimization problems, the primary source of time consumption is the wall-clock time cost. In a single optimization cycle, the wall-clock time cost primarily consists of three

components: experimental model preparation, waiting gap, and experimental evaluation (Figure 4.3). An effective approach to significantly improving optimization efficiency is to achieve the desired optimization objective with as few optimization cycles as possible. Therefore, the multi-point infill criterion was proposed based on this principle.

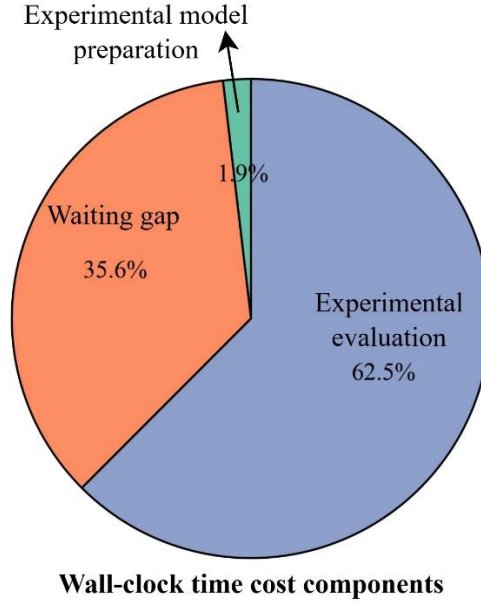


Figure 4.3. Composition of the wall-clock time cost within a single optimization cycle for practical problems.

To investigate the impact of different infill point quantities on the optimization efficiency of the parallel EGO algorithm, Table 4.3 presents the relevant statistics on the number of optimization cycles required to achieve the predefined convergence criteria for the parallel EGO algorithm based on both OK and GEK models. For the case based on OK model, the number of optimization cycles continuously decreases with an increasing number of infill points for almost all test functions, except for F3. This result is generally consistent with the findings presented in [39]. However, the reduction in the number of cycles required for the parallel EGO algorithm to reach the convergence criterion does not strictly follow a proportional relationship with the increase in the number of infill points. For example, on F1, taking the parallel EGO algorithm with an infill point count of 2 as the baseline, the parallel EGO algorithm with an infill point count of 4 accelerates the search by approximately $92.77/49.33 \approx 1.88$ times. However, for a strictly linear speedup, the expected factor should be 2. A possible explanation given in previous studies is that as the sequential selection process progresses within a single optimization cycle, an increasing number of IFs are multiplied by the initial EI criterion, making the PEI criterion increasingly unreliable. It is noteworthy that for F5, the parallel EGO algorithm with a point filling number of 4 achieved an acceleration factor of approximately $53.13/26.63 \approx 2.00$ times.

Table 4.3. The statistics of the number of cycles required to reach the expected stopping criterion under different point-filling strategies using parallel EGO algorithms based on OK and GEK, respectively.

Systems	Test problems	Stopping criterions	Number of cycles	2 points	4 points	6 points	8 points	10 points
OK-based	F1	Expected optimum < 1.0	Median	91.00	48.00	36.50	27.50	24.00
			Mean	92.77	49.33	35.00	26.83	22.23
			Std	16.15	7.49	4.70	4.03	2.36
GEK-based			Median	15.50	11.00	7.00	6.00	5.00
			Mean	15.97	11.47	8.07	6.90	6.17
			Std	4.14	4.30	2.22	1.70	1.93
OK-based	F2	Expected optimum < 0.7	Median	32.00	19.50	13.00	10.00	9.00
			Mean	39.10	23.43	13.13	11.17	10.23
			Std	29.74	16.36	7.32	6.89	5.59
GEK-based			Median	27.50	9.50	8.00	8.00	5.50
			Mean	26.93	13.80	10.23	8.43	7.20
			Std	19.17	9.35	5.78	3.47	3.19
OK-based	F3	Expected optimum < -490	Median	2.00	2.00	2.00	2.00	2.00
			Mean	1.93	1.93	1.93	1.93	1.93
			Std	0.25	0.25	0.25	0.25	0.25
GEK-based			Median	8.00	6.00	6.00	7.00	4.00
			Mean	11.20	7.57	7.53	7.23	5.33
			Std	10.50	5.63	5.04	4.59	3.09
OK-based	F4	Expected optimum < -395	Median	36.50	23.00	14.50	15.00	14.00
			Mean	40.83	29.03	19.70	16.23	14.53
			Std	18.88	16.17	10.81	7.59	6.32
GEK-based			Median	7.00	4.00	4.00	3.50	3.00
			Mean	9.60	6.50	5.17	4.63	4.20
			Std	8.19	4.88	3.74	3.06	2.45
OK-based	F5	Expected optimum < -294	Median	44.50	21.00	15.00	13.00	11.00
			Mean	53.13	26.63	19.37	13.83	13.47
			Std	29.76	16.28	11.57	5.22	5.93
GEK-based			Median	10.00	6.50	5.00	5.00	5.00
			Mean	13.13	8.13	5.93	5.40	4.77
			Std	9.96	4.81	3.44	2.33	1.76
OK-based	F6	Expected optimum < -50	Median	51.50	52.50	23.00	25.00	17.00
			Mean	58.50	42.60	24.17	21.20	16.90
			Std	34.35	19.00	11.22	8.91	6.69
GEK-based			Median	6.00	4.00	4.00	3.00	3.00
			Mean	14.10	8.07	7.03	6.03	4.97
			Std	14.14	7.23	5.94	4.41	2.81
OK-based	F7	Expected optimum < 200	Median	59.00	28.50	22.50	17.50	13.50
			Mean	69.33	36.00	24.20	19.03	15.53
			Std	42.81	19.93	12.49	9.89	7.33
GEK-based			Median	14.50	6.50	6.00	6.00	6.50
			Mean	23.80	12.60	8.90	7.27	7.03
			Std	20.94	10.54	6.35	4.29	3.52

Table 4.3. Continuous.

Systems	Test problems	Stopping criterions	Number of cycles	2 points	4 points	6 points	8 points	10 points
OK-based	F8	Expected optimum < 505	Median	41.50	26.00	15.00	12.00	13.50
			Mean	38.03	24.93	16.17	12.57	12.60
			Std	25.14	18.38	11.61	8.33	8.01
GEK-based			Median	60.00	20.00	19.00	15.00	12.00
			Mean	42.20	17.87	14.00	11.27	9.00
			Std	24.33	11.43	7.73	5.66	4.34
OK-based	F9	Expected optimum < 1100	Median	50.00	27.50	40.00	16.50	21.00
			Mean	61.43	35.20	29.83	19.00	18.07
			Std	39.14	20.69	12.97	9.44	6.84
GEK-based			Median	5.00	4.00	3.50	3.00	3.00
			Mean	14.27	8.27	7.27	5.53	4.70
			Std	17.96	9.17	6.68	4.18	2.85

To explain this phenomenon, Figure 4.4 illustrates the process of the PEI criterion searching for multiple potential improvement points within an optimization cycle. Strictly speaking, the fundamental principle of the PEI criterion remains a sequential process, rather than a true parallel point-filling criterion (its parallel concept is based on the parallelization of external computational resources, rather than the parallelization of the algorithm itself). The search for the first potential improvement point relies on the original EI criterion, while the search for subsequent points is sequentially adjusted by multiplying with the corresponding IFs. The search for later points depends on the search space information provided by earlier points, rather than each point independently having its own fixed search space information. This means that the search space information for multiple potential points is highly redundant, leading to the observation that optimization efficiency does not increase in a linear proportion with the number of filled points. Additionally, the EI criterion itself strikes a perfect balance between local exploitation and global exploration. However, in complex optimization problems, OK often fails to provide globally reliable design space information, causing the classical EGO algorithm to lean more towards local optimization. In certain special cases, the observed linear proportional improvement in optimization efficiency with the increase in the number of filled points may be due to the incorporation of IFs, which provides more algorithmic diversity to the classical EGO algorithm. This increases the probability of escaping local optima and facilitates the EGO algorithm in finding higher-quality solutions.

For the case based on GEK model, compared to the parallel EGO algorithm with OK model, the number of optimization cycles with the same number of point fillings significantly decreased in almost all test functions, except for F3 and F8. Because the fitting performance of GEK model has been significantly improved. As a result, the effective spatial information relied upon during the search for potential improvement points is more reliable, which facilitates the convergence of the parallel EGO algorithm.

However, the redundancy of the spatial information relied upon by multiple points in a single optimization cycle also increases, which means the bottleneck for achieving optimization efficiency improvements occurs more quickly than in the case of OK model.

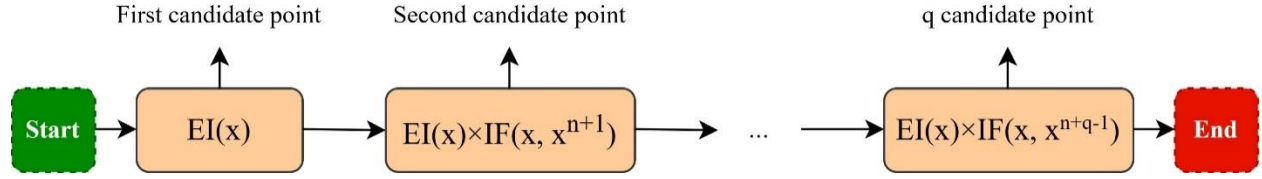


Figure 4.4. The process for searching multi-points using the PEI criterion within a single optimization cycle.

4.2.4 Influence of the parallel point infills strategy on the diversity of parallel EGO algorithm

When an optimization algorithm is prone to getting trapped in local optima, strategies can be employed to provide population diversity, helping the algorithm escape local optima and thereby increasing the probability of finding higher-quality solutions [66]. The classical EGO algorithm inherently achieves a perfect balance between local exploitation and global exploration; however, it is also prone to getting trapped in local optima. The introduction of the parallel EGO algorithm not only enhances the efficiency of the entire optimization process but also, by multiplying the corresponding number of IFs after original EI function, can be seen as increasing the diversity of the optimization. Therefore, the impact of different point filling quantities on the performance of the parallel EGO algorithm is analyzed from the perspective of algorithmic diversity. The calculation method for population diversity is as follows:

$$Pop_{div}(l) = \frac{1}{n} \sum_{i=1}^n \sqrt{\sum_{j=1}^D (\overline{pos}_j - pos_{i,j}(l))^2}, \overline{pos}_j = \frac{1}{n} \sum_{i=1}^n pos_{i,j}(l) \quad (30)$$

Where n and D represent the total population size and the number of design variables, respectively. $pos_{i,j}(l)$ represents the position of the i individual in the j dimension, and $\overline{pos}_j$ denotes the average position of all individuals in the j dimension. A low population diversity indicates that the search is exploring the design space near the local optimum. In contrast, a high population diversity suggests that the search is exploring regions of the design space farther from the local optimum.

Figures 4.5 and 4.6 present the average population diversity curves of the parallel EGO algorithm based on OK model and GEK model, respectively, under different point-filling quantities during optimization process. For the case based on OK model, the parallel EGO algorithm with a larger point-filling quantity generally exhibits greater population diversity in the early stages of optimization, indicating

that it explores a larger design space. As the optimization process progresses, population diversity gradually decreases. Meanwhile, the rate of decline in population diversity is relatively slower for the parallel EGO algorithm with a larger point-filling quantity. This also explains why its overall solution convergence speed is relatively slower (Figure 4.1). However, the advantage of having greater population diversity is that it can facilitate the parallel EGO algorithm in escaping local optima, thereby increasing the probability of discovering higher-quality optimal solutions. For example, in the case of test functions F2, F3, F5, and F7, higher-quality solutions were found when using a larger number of filled points.

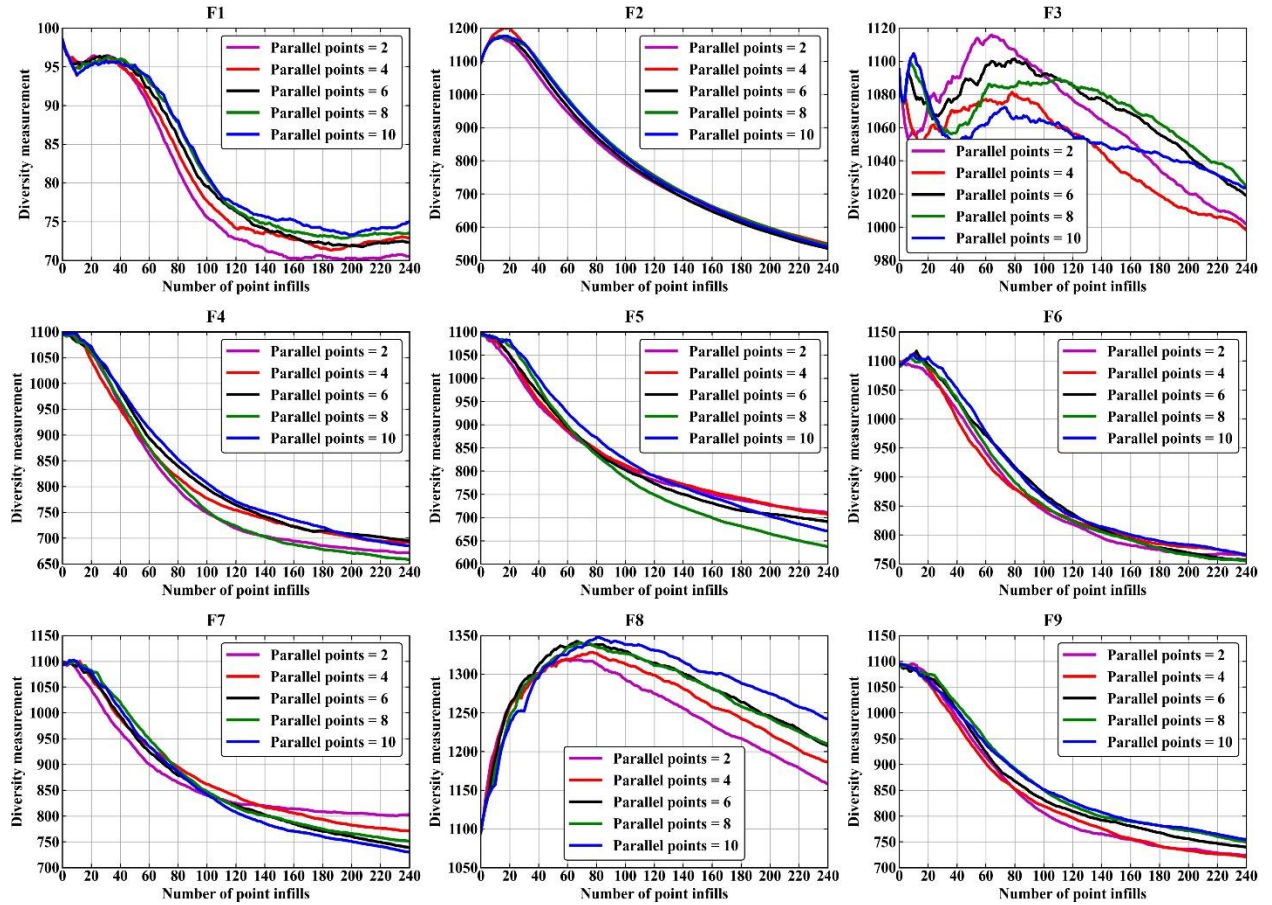


Figure 4.5. Population diversity curves of the OK-based parallel EGO algorithm on all tested problems.

For the case based on GEK model, overall, the impact of different point filling numbers on the population diversity of the parallel EGO algorithm is reduced, except for test functions F1, F3, and F8. Combining the average convergence curves for the corresponding test functions in Figure 4.2, it can also be observed that the differences in the impact of different point filling numbers on the convergence speed of the parallel EGO algorithm are minimal. This may be due to the fact that the spatial design information

provided by GEK to the parallel EGO algorithm is sufficiently reliable, which makes the search directions for potential improvement points more well-defined.

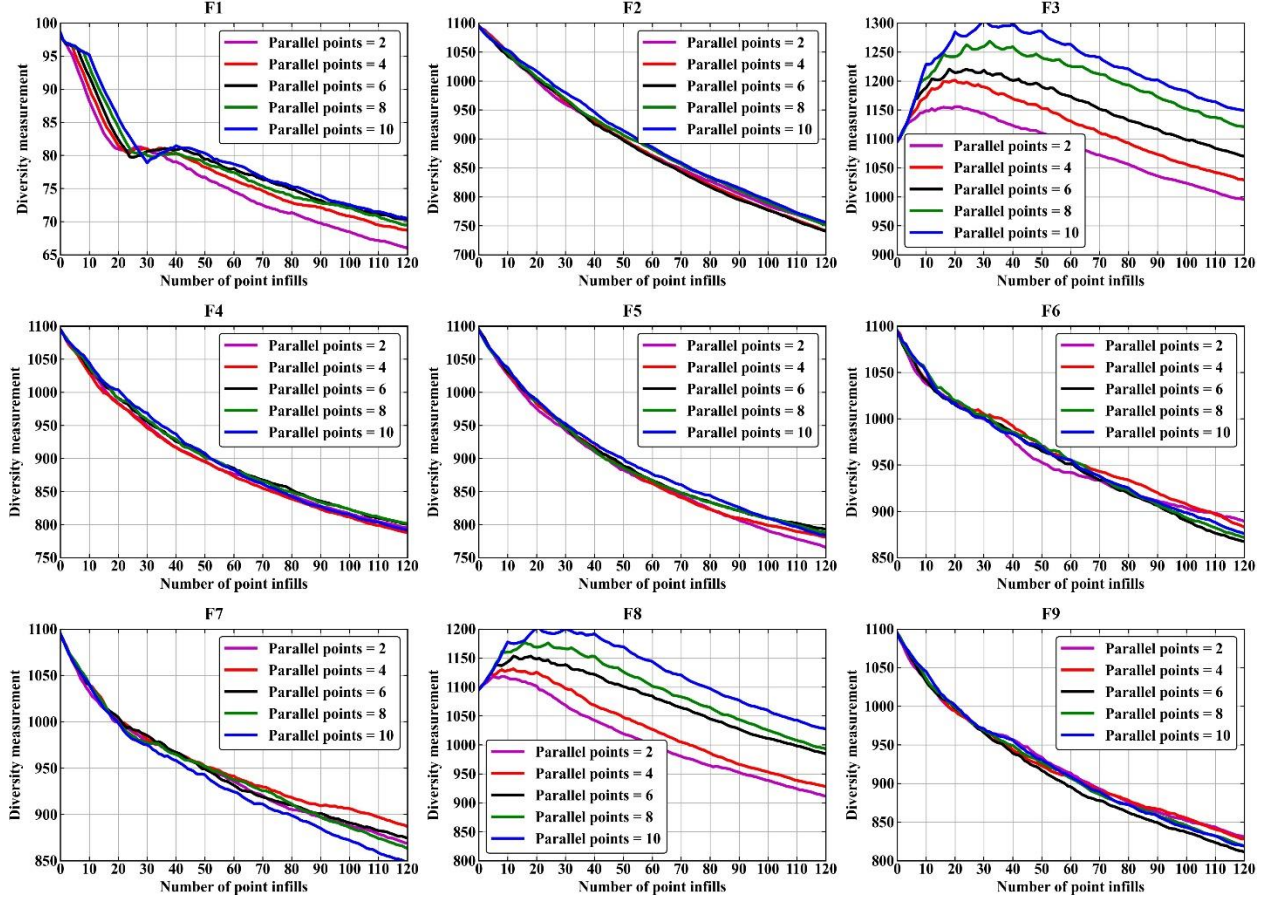


Figure 4.6. Population diversity curves of the GEK-based parallel EGO algorithm on all tested problems.

Almost all global optimization algorithms aim to achieve a reasonable balance between local exploitation and global exploration, and the EGO algorithm is no exception. To evaluate the impact of different point filling numbers on the balance between local exploitation and global exploration in the parallel EGO algorithm, the percentage change trends of exploitation-exploration throughout the optimization process were calculated using the following formula:

$$exploration(l) = \left(\frac{Pop_{div}(l)}{\max(Pop_{div}(l))} \right) \times 100\% \quad (31)$$

$$explitation(l) = \left(\frac{\|Pop_{div}(l) - \max(Pop_{div}(l))\|}{\max(Pop_{div}(l))} \right) \times 100\% \quad (32)$$

Here, Pop_{div} is defined in Eq. (30) and represents the diversity of the entire population.

Figures 4.7 and 4.8 show the trends of the exploration-exploitation ratio over the process of optimization for the parallel EGO algorithm based on OK and GEK models, respectively, under different point-filling quantities. Overall, it can be observed that the parallel EGO algorithm with different point-filling quantities maintains a high level of exploration during the early stages of optimization, allowing the algorithm to explore a broader design space for global optima. As the optimization progresses, the exploration rate gradually decreases while the exploitation rate increases, enabling the algorithm to continue improving the solution quality in the vicinity of local optima. For the case based on OK model, as shown in Figure 4.1, the point-filling quantity of 2 consistently achieves the best convergence speed on F1, F4, F6, F8, and F9. Therefore, with a point-filling quantity of 2, the parallel EGO algorithm better maintains a reasonable balance between local exploitation and global exploration. This conclusion can be drawn from the trend of the exploration-exploitation curves in Figure 4.7, as the exploration-exploitation curves for the case with 2 points remain smaller and larger than those for the other point configurations throughout almost the entire optimization process.

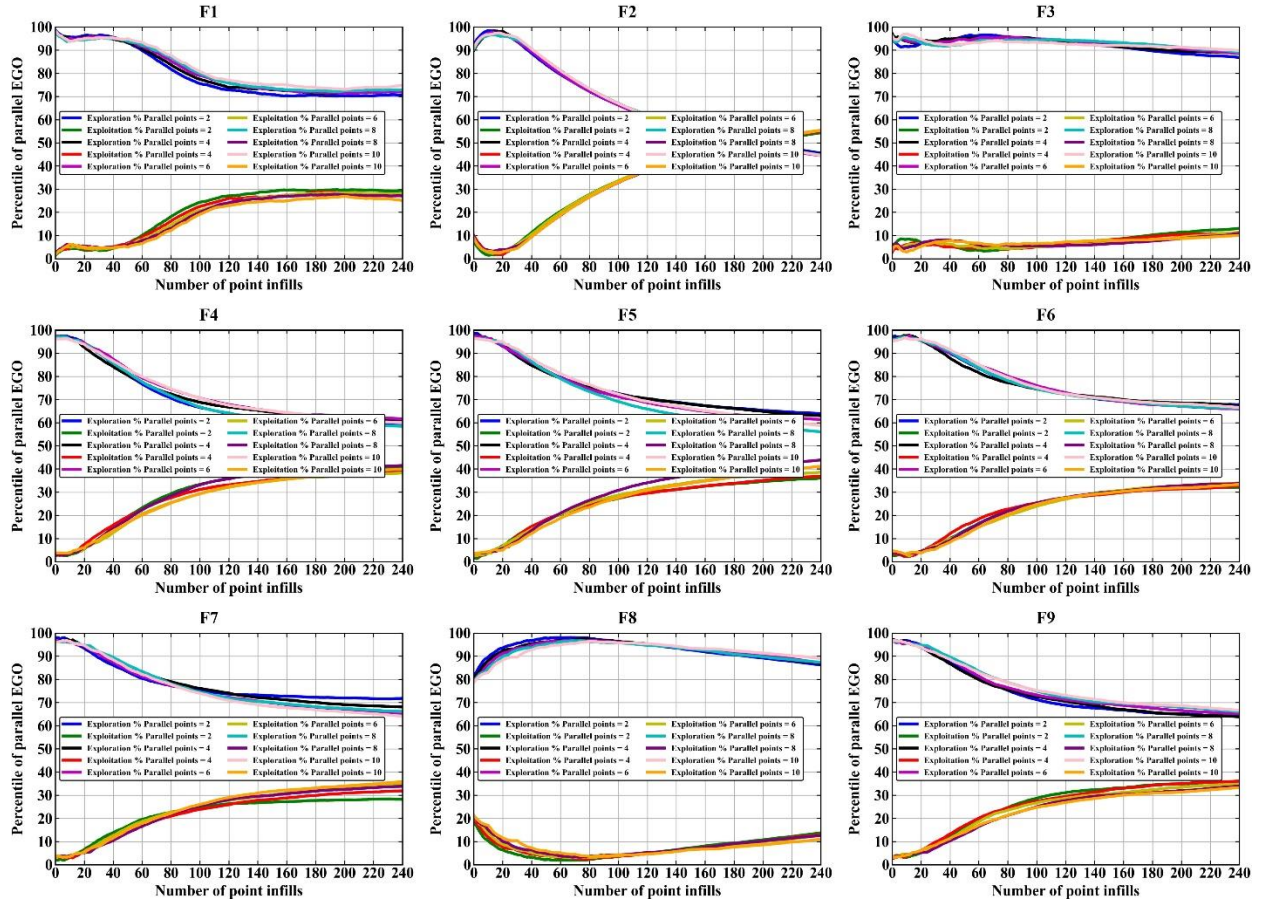


Figure 4.7. Exploration and exploitation curves of the OK-based parallel EGO algorithm on all tested problems.

For the case based on GEK model, as shown in Figure 6, except for F1, F3, and F8, where there is a significant difference in convergence speed between different point-filling quantities, the convergence speed for the other test functions only shows slight differences in the early stages of optimization. As the optimization iterations progress, the convergence speed remains nearly consistent. Therefore, as shown in Figure 4.8, the differences in the impact of different point-filling quantities on the balance between local exploitation and global exploration of the parallel EGO algorithm are negligible for these test functions. This indicates that GEK model provides greater reliability and stability to the parallel EGO algorithm.

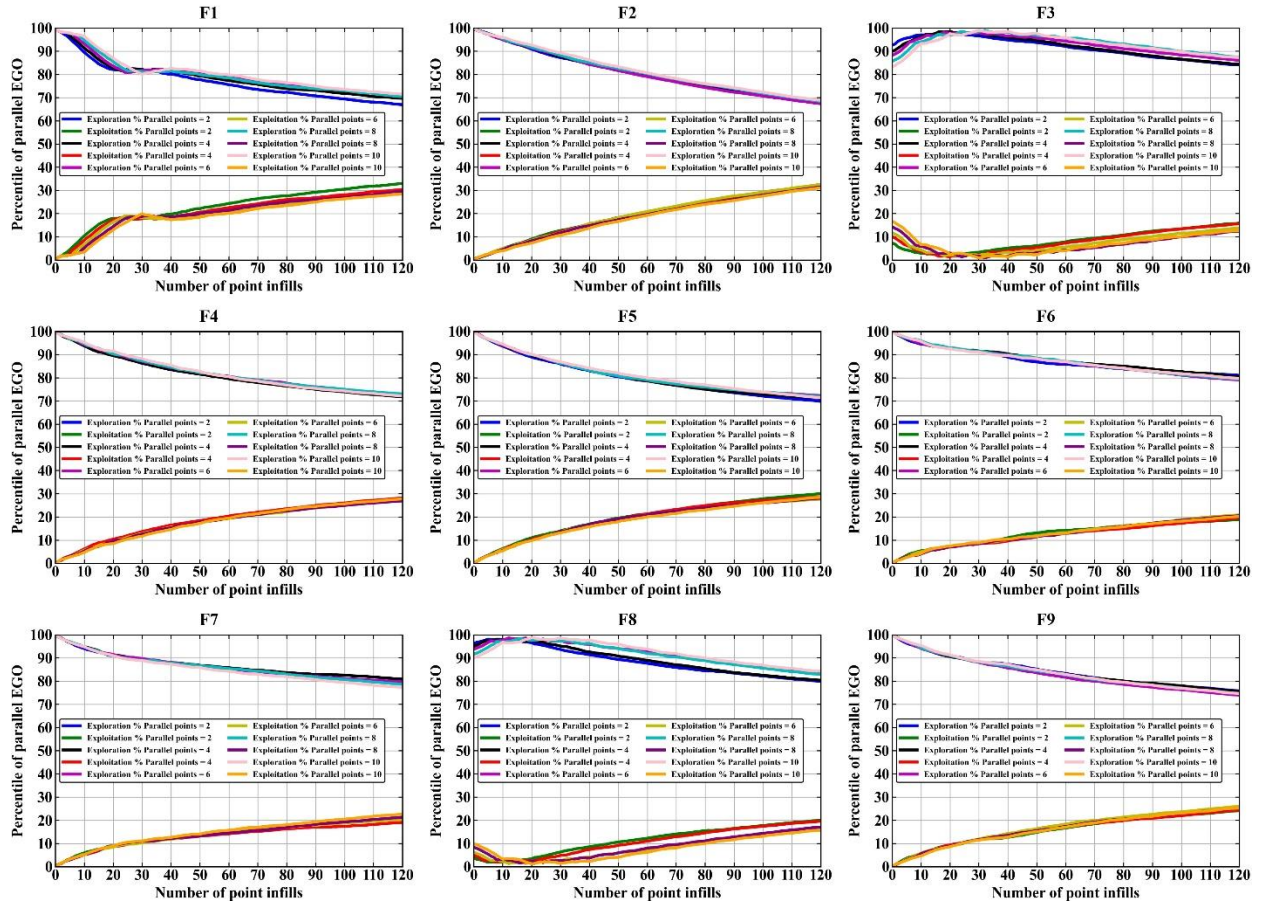


Figure 4.8. Exploration and exploitation curves of the GEK-based parallel EGO algorithm on all tested problems.

CHAPTER 5. CONSTRUCTION AND VALIDATION OF THE PROPOSED IMPROVED OPTIMIZATION ALGORITHM SYSTEM

5.1 Matching the Suitable Optimizer

To match the most suitable optimizer for the GEK based parallel EGO algorithm system, this section first elaborates on and analyzes the critical role of the optimizer within the entire optimization system, as well as how it impacts the overall optimization performance. Then, it performs a comprehensive investigation into the influence of different types of optimizers on the performance of the proposed system.

5.1.1 Key role of optimizer in current optimization system

Based on the theory described above, if q potential improvement points are to be selected in each point infill cycle, then $m \times (q + 1)$ optimization processes are sequentially performed for the GEK-based parallel EGO algorithm, where m represents the total number of cycles. Zhan et al. [39] found that, in large-scale and high-dimensional optimization problems, the resulting PEI function often exhibits complex and highly multimodal properties, making it very difficult for optimizers to identify globally optimal solutions on the PEI function. In the parallel EGO algorithm, during each point infill cycle, the first updating point is located using the original EI function. The remaining updating points are then determined by multiplying the corresponding IFs with the original EI function. The quality of the earlier updating points indirectly impacts the quality of the later updating points throughout the entire optimization process, ultimately affecting the quality of the solution. Zhan et al. [39] briefly studied the influence of the DE algorithm with different parameter settings on the performance of the OK-based parallel EGO algorithm. The results revealed that the performance of the optimizer has a significant impact on the convergence performance of the OK-based parallel EGO algorithm. Given the relatively low computational cost of evaluating the PEI function, it was also recommended to conduct as many evaluations as possible to ensure the global maximum of the PEI function is located. However, the investigation only explored the influence of the DE algorithm with different parameter settings on the convergence performance of the OK-based parallel EGO algorithm and the tested benchmark functions were relatively simple. Therefore, Wang et al. [41] systematically studied the influence of various optimizers and benchmark functions on the performance of OK based parallel EGO algorithm. The results demonstrated that optimizers with a higher central goal of exploration–exploitation promote faster convergence to high-quality solutions. Which further confirmed that the performance of the optimizer significantly influences the performance of OK-based parallel EGO algorithm. However, it is clearly see that the tuning hyper-parameters [50, 51] and the optimizing PEI function are two different optimization processes, thereby the optimization strategies may differ. Previous

research using the same optimizer to tune the hyper-parameters and optimize the PEI function treated the two optimization processes as a unified complete. Which will bring challenge for analyzing the specific mechanisms whereby various optimizers influences the convergence performance of the parallel EGO algorithm. Therefore, it is necessary to separate two optimization processes, keeping one optimizer fixed for one process while using different optimizers for the other process. Furthermore, as the GEK model incorporates into optimization system, its correlation vectors and matrices become more complex, leading to a significant increase in the difficulty of hyper-parameter tuning and PEI function optimizing. It is possible that previous conclusions are no longer applicable to the proposed new system.

To further explain the positive impact of the differences between EI and PEI on the convergence of the parallel EGO algorithm, the ordinary EGO and parallel EGO algorithms were used to perform point infills on the one-dimensional Griewank function within the feasible region $[-10, 10]$. The function has a global minimum of zero at point 0. The parallel EGO algorithm inserts two points in each cycle. Figure 5.1 illustrates the differences in the EI and PEI functions during the point infill process. Figure 5.1a shows the respective locations of the second point infills obtained by the ordinary EGO and parallel EGO algorithms. After updating the first point, certain differences arise between the EI and PEI functions due to the ordinary EGO algorithm updating the new EI function through the updated kriging model. In contrast, the parallel EGO algorithm does not update kriging model, but instead simulates the behavior changes of the EI function using the IFs. Therefore, the PEI function approximates the current EI function. Both methods select second point infills located a certain distance from the optimal value of the test function. Figure 5.1b shows the respective locations of the third point infills obtained by the ordinary EGO and parallel EGO algorithms. Evidently, the EGO algorithm with the PEI criterion finds a point infill location that is closer to the optimal solution than the point infill selected by the EGO algorithm with the EI criterion. Therefore, a reasonable difference between the PEI and EI functions facilitates faster convergence to high-quality solutions for the parallel EGO algorithm. It is worth noting that the performance of the optimizer in finding the maximum potential improvement points in the PEI function affects the approximation quality of the EI and PEI functions. Typically, an optimizer with higher global search capabilities ensures a better approximation of the PEI and EI functions. Conversely, an optimizer with insufficient global search capabilities may amplify the differences between the PEI and EI functions. Hence, it is worth exploring whether this elegant balance of approximation and differences between EI and PEI induced by the IFs and optimizer will be disrupted when the GEK model incorporates into the system.

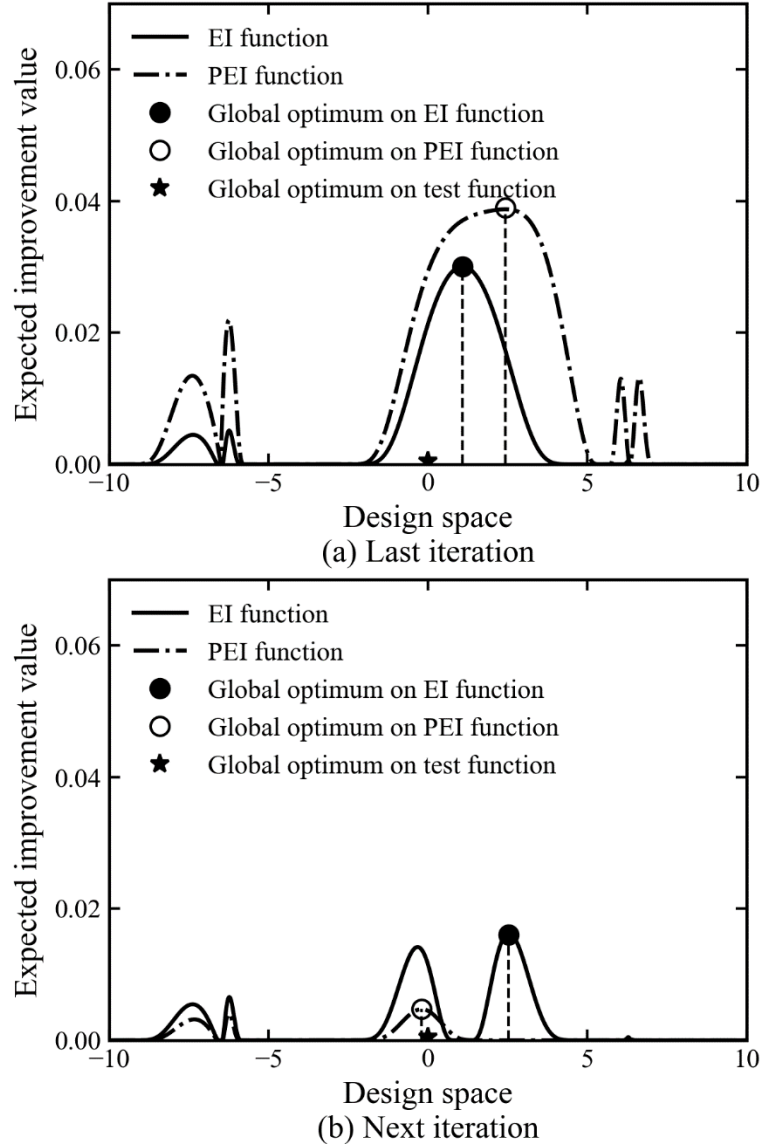


Figure 5.1. Comparison of changes in EI and PEI functions during the process of point infills.

5.1.2 Experimental setting

To explore the influence of different optimizers on convergence performance of GEK based parallel EGO system. The gradient-based GS-SQP optimizer is fixed to tune the hyper-parameters [50]. Different optimizers were selected to search for potential improvement points on the PEI function. Experiments were conducted on seven benchmark test functions (described in Section 5.2.1). The final results determine the most suitable optimizer for the proposed system and provide a reference for other investigators and developers to select appropriate optimizers.

The candidate optimizers include two traditional algorithms, DE [52] and PSO [53]; two recently enhanced DE-based algorithms, enhanced adaptive guided DE (EAGDE) [18] and chaotic local search-based JADE (CJADE) [19]; a hybrid optimizer, modified Gaussian bare-bones DE (MGBDE) [20], which demonstrated the best performance in Wang et al. [41]; and a novel algorithm inspired by the hunting behavior of gorilla groups, the artificial gorilla troop optimizer (GTO) [21]. The optimizers were selected for their distinct advantages, allowing an analysis of the influence of different aspects on the performance of the proposed system. DE, EAGDE, and CJADE focus on global exploration, but exhibit relatively slow convergence speeds. PSO accelerates the entire optimization process through individual learning and social cooperation. MGBDE emphasizes the balance between local exploitation and global exploration by combining the advantages of PSO and DE. The nature-inspired GTO enhances the global search capability by using various random mechanisms. In summary, the impact of the optimizers on the performance of GEK based parallel EGO system is investigated in terms of the convergence speed, balance between local exploitation and global exploration, and global search capability.

To guarantee a comprehensive evaluation of each optimizer's impact on the performance of the proposed system, the parameters setting for each optimizer were set to the optimal working states, determined through numerous tests. Table 5.1 provides the detailed parameter settings for each optimizer. A consistent population size of 50 and a maximum iteration count of 500 were applied across all optimizers. The tests focused on optimizing benchmark functions. Please refer to the 5.2.1 section for the selection of benchmark functions and parameter setting of the algorithm system. Additionally, before analyzing the impact of the different optimizers on the performance of the proposed system, a comparison was conducted to better understand the inherent performance of each optimizer. Three key performance metrics were evaluated: global search capability, algorithm complexity, and convergence speed through a series of algorithm tests. Table 5.2 presents the comparison results.

Table 5.1. Parameters setting for each optimizer.

Optimizer	Parameters	Value
GTO	P; β ; w	0.03; 3; 0.8
PSO	c1; c2; w	1.49618; 1.49618; 0.7298
DE	F; cr	0.8; 0.9
MGBDE	F; Mu_cr; Sigma_cr	0.5; 0.5; 0.1
EAGDE	F; p_rate; N _{min}	Rand(0.1, 0.9); 0.1; 12
CJADE	Mu_F; Mu_cr; ρ	0.5; 0.5; 0.988

Table 5.2. Performance comparison of different optimizers.

Ranking	Global search capability	Complexity	Convergence speed
1	CJADE	PSO	PSO
2	EAGDE	DE	MGBDE
3	MGBDE	MGBDE	EAGDE
4	GTO	EAGDE	CJADE
5	DE	CJADE	DE
6	PSO	GTO	GTO

5.1.3 Results and discussion

To clearly illustrate the impact of different optimizers on the performance of the proposed system, the results obtained from running the algorithm system with six distinct optimizers are compared. Figure 5.2 shows the results of the Wilcoxon signed-rank test, comparing the convergence levels of MGBDE, PSO, and DE against other optimizers across all tested functions, respectively. The symbol “=” indicates that there is no significant difference in performance between the two compared optimizers on the proposed system. Furthermore, although p -values between 0.05 and 0.10 for the significance level cannot reflect significant differences in the performance of the two compared optimizers on the proposed system, the symbols “ $\geq$ ” and “ $\leq$ ” are still included in the figure to show that the first optimizer potentially outperforms or underperforms the second optimizer, respectively, on the proposed system. It is clearly observed that there are no significant differences in performance between the PSO, DE, and MGBDE optimizers for the most of tested functions. For the f_{10} and $FMSW$ functions, PSO achieves significantly better performance than MGBDE and DE, respectively. Whereas for the function f_{12} , MGBDE significantly outperforms DE. Although most of the statistical results for PSO, DE, and MGBDE in Figure 6 are marked with the “=” symbol, indicating comparable performance, MGBDE demonstrates a slightly better performance than the other two optimizers. Evidence for this can be found in Figure 6a, where most of the bar charts for MGBDE are positioned above the x-axis. Furthermore, from an overall perspective, PSO, DE, and MGBDE demonstrate significantly better performance than EAGDE, CJADE, and GTO. The statistical results support this observation, as PSO, DE, and MGBDE are more frequently assigned the symbols +, $\geq$, and = in comparisons with EAGDE, CJADE, and GTO, respectively. Consequently, the results suggest that MGBDE, DE and PSO improve the convergence performance of the GEK based parallel EGO algorithm system within a limited number of samples.

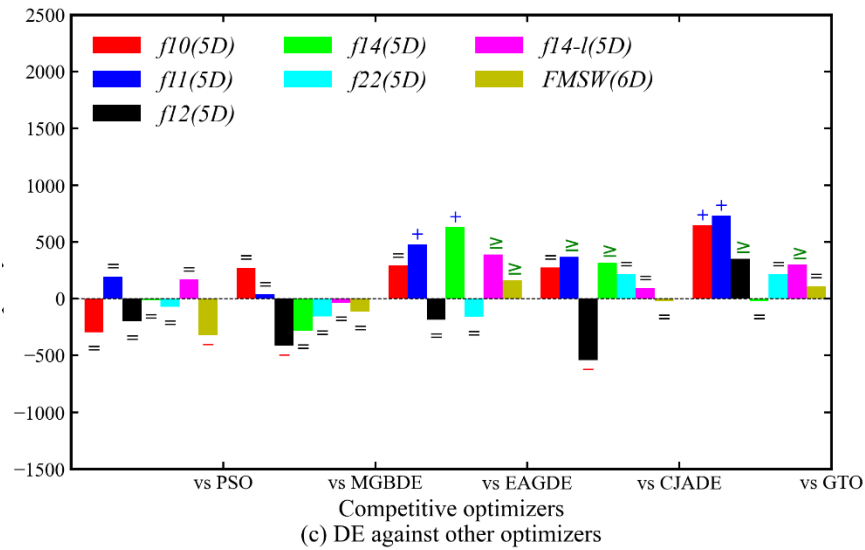
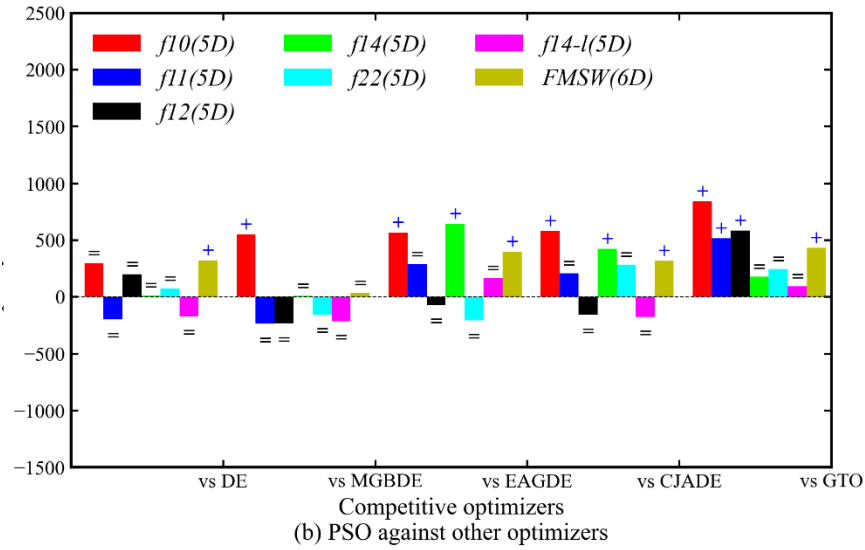
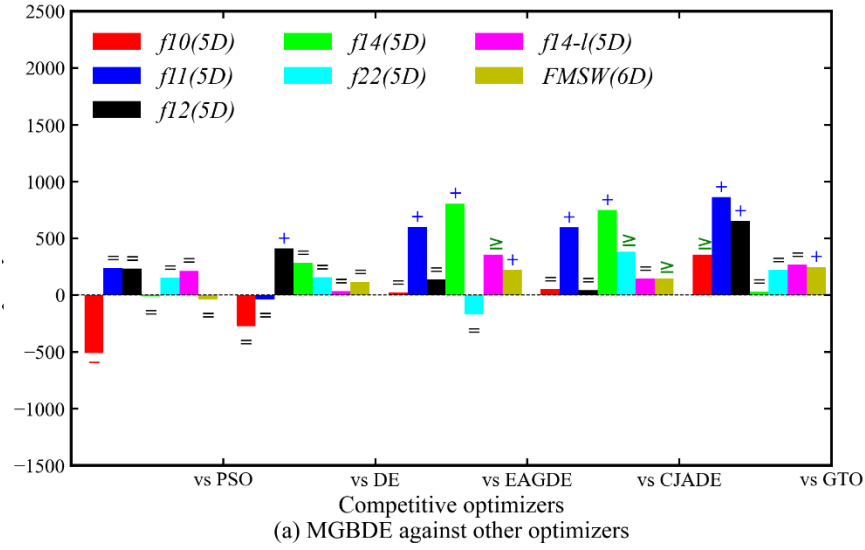


Figure 5.2. Wilcoxon signed-rank test for MGBDE, PSO and DE optimizer with other optimizers, respectively.

The rankings of each optimizer for each test function are more intuitively presented in Figure 5.3. MGBDE achieves the top ranking for test functions f11, f12, f14, and f14-l, and also performs well on f22 and FMSW, securing second place. Performance shows inferiority only on f10, lagging behind most of the other optimizers. The traditional PSO, known for its fast convergence speed, exhibits performance similar to that of MGBDE. In Figure 5.3, PSO demonstrates dominant performance on functions, and its ranking results indicate that this optimizer enables the proposed system to converge to high-quality solutions for most of the other test functions. Furthermore, the traditional DE, known for its strong global search capability, shows slightly inferior performance compared to MGBDE and PSO. However, DE outperforms the other three optimizers, as evidenced by the mean ranking results in Table 5.3. Surprisingly, the two DE variants, EAGDE and CJADE, which are known for their stronger global search capabilities than DE, do not provide any significant advantages in improving the convergence performance of the proposed system. This conclusion is further supported by the mean rankings in Table 5.3, where both EAGDE and CJADE are ranked fourth. The recently developed GTO performs reasonably well only on function f14, but its performance is among the weakest across the remaining test functions.

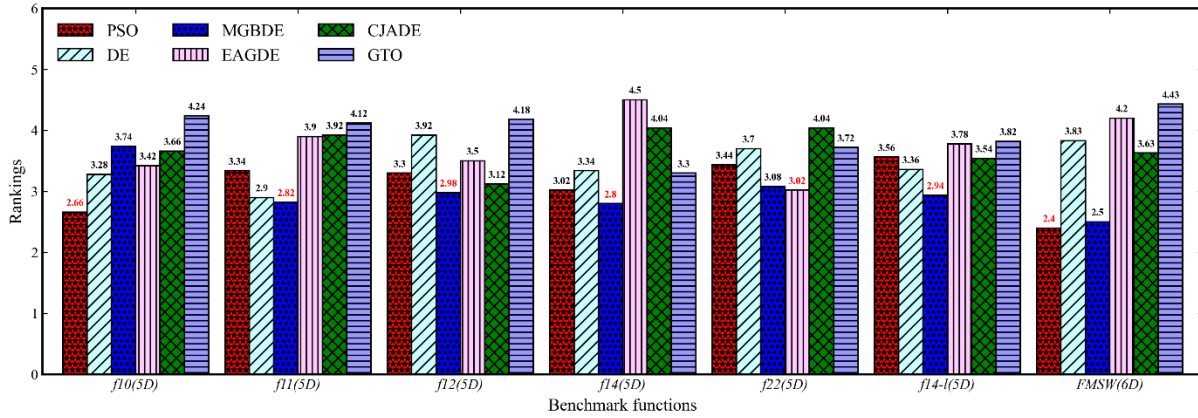


Figure 5.3. Friedman test between each optimizer for all tested functions.

Table 5.3. Mean rankings of optimizers for all tested functions.

Ranking	Optimizer	Mean ranking	<i>p</i> value
1	MGBDE	1.8571	0.0062
2	PSO	2.4286	
3	DE	3.2857	
4	EAGDE	4.0000	
4	CJADE	4.0000	
6	GTO	5.4286	

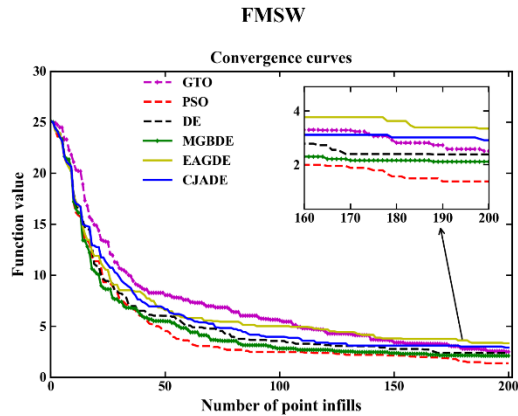
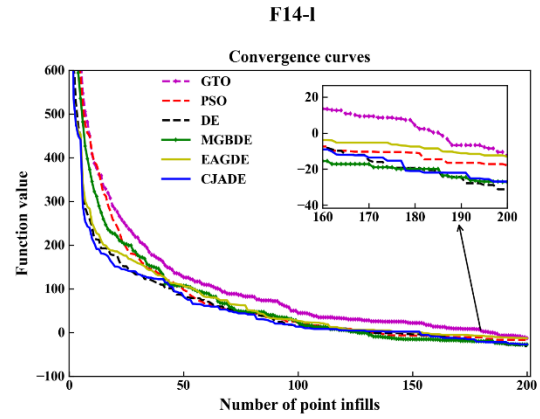
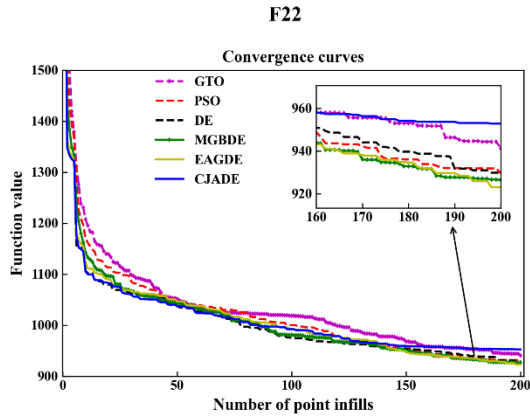
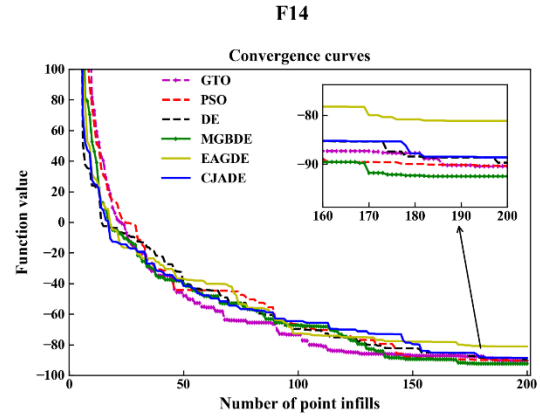
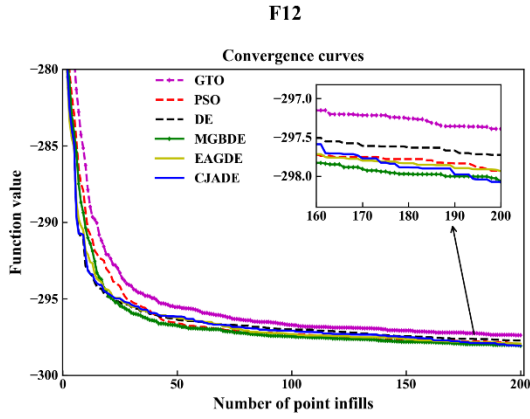
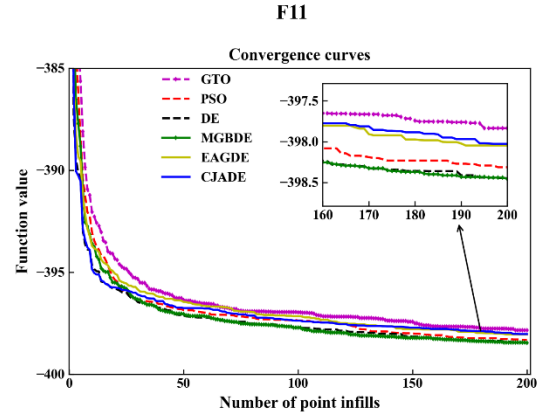
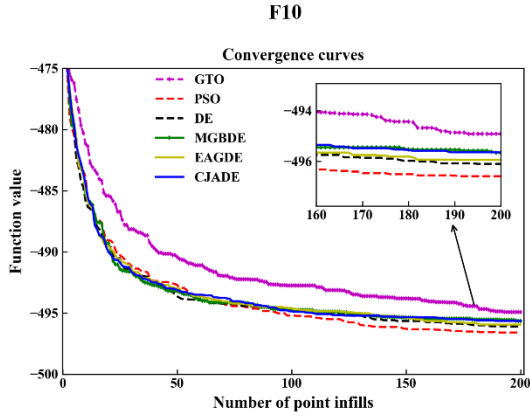


Figure 5.4. Average convergence curves of GEK-based parallel EGO algorithm with different optimizers on all tested functions.

To more intuitively observe the convergence trends of different optimizers throughout the infill optimization process of the GEK based parallel EGO algorithm and to further evaluate their overall performance, the average convergence curves of the GEK based parallel EGO algorithm with different optimizers over 50 runs for all test functions are presented in Figure 5.4. Since the differences in the average convergence curves during the initial few infill cycles are minimal across all test functions, the results from the earliest infill cycles are not visualized. For f10, it is evident that the GEK based parallel EGO algorithm with PSO and DE optimizers converges significantly faster than those with MGBDE, EAGDE, CJADE, and GTO within a limited number of infill points. This observation further corroborates the ranking results of the optimizers for f10 shown in Figure 5.3. For f11 and FMSW, the average convergence curves of MGBDE, DE, and PSO remain lower than those of EAGDE, CJADE, and GTO throughout almost the entire convergence process, indicating that the GEK based parallel EGO algorithm with MGBDE, DE, and PSO optimizers generally converges faster than those with EAGDE, CJADE, and GTO. For f12, MGBDE consistently ensures faster convergence of the GEK based parallel EGO algorithm throughout nearly the entire infill optimization process. Although the convergence curve of PSO gradually flattens in the later stages of infill, it exhibits convergence performance similar to MGBDE in the range of approximately 50 to 170 infill points. For f14, f22, and f14-l, while no single optimizer demonstrates an absolute advantage in convergence speed across the entire infill process, it can still be observed that the GEK based parallel EGO algorithm with the MGBDE optimizer exhibits a better convergence trend in the later stages compared to other optimizers. For PSO, although its average convergence curve shows slower convergence in the early and middle stages for f14 and f22, a noticeable decline occurs in the later stages. However, for the GEK based parallel EGO algorithm with EAGDE and CJADE optimizers, which possess stronger global search capabilities, their convergence speeds are faster than other optimizers in the early stages of infill for nearly all test functions. Yet, as the optimization progresses, their convergence rates gradually stabilize. The traditional DE optimizer exhibits a convergence trend similar to the two DE variant optimizers (EAGDE and CJADE) in the early stages. However, compared to the two DE variants, DE tends to achieve a declining trend more easily in the middle and later stages of infill. Finally, the GEK based parallel EGO algorithm with the recently developed emerging optimizer GTO exhibits the slowest convergence speed in almost all test functions, except for f14.

In summary, the hybrid optimizer MGBDE, which with a high central goal of exploration–exploitation, achieved superior convergence performance in the GEK based parallel EGO algorithm system by balancing the approximation and difference between the EI and PEI criterion. Additionally, although the PSO

optimizer focuses more on rapid convergence and cannot always guarantee high-quality solutions on the PEI function, it still provides valuable algorithmic diversity to the proposed system. The diversity increases the probability of escaping local optima, leading to PSO's performance being second only to MGBDE. Furthermore, although EAGDE and CJADE demonstrate strong global search capabilities and often find high-quality solutions on the PEI function, they did not significantly improve the convergence performance of the proposed system. A potential reason is that finding high-quality solutions on the PEI function does not ensure an absolute improvement, as the PEI function is merely an approximation of the EI function, rather than the true EI function. Since the EI function is also based on the current surrogate model, high-quality solutions found on the PEI function may not lead to an improvement in the optimization objective, making it difficult to escape local optima. This limitation ultimately results in a reduction in convergence performance. Similarly, the GTO optimizer, which also possesses strong global search capabilities, demonstrates poor convergence performance in the proposed system due to reasons similar to those affecting EAGDE and CJADE.

5.2 Performance Verification of the Proposed Optimization Algorithm System

To evaluate the performance of the proposed system in comparison with the traditional OK-based parallel EGO system, performance tests were conducted on both benchmark functions and two real-world optimization problems of aerodynamics.

5.2.1 Verification on benchmark test functions

To evaluate the convergence performance of the proposed system in comparison with the traditional OK-based parallel EGO system, the MGBDE optimizer, identified as the most suitable for the proposed system, is employed to optimize the PEI function. Additionally, a gradient-based optimization algorithm, GS-SQP, is employed to tune the hyper-parameters of the GEK model. Table 5.4 provides descriptions of five benchmark functions and one engineering optimization problem, selected respectively from CEC2013 [54] and CLEC2011 [55], for performance evaluation. To maintain a reasonable computational cost, the feasible regions of all test functions were scaled to smaller ranges compared to the original intervals of $[-100, 100]^5$ and $[-6.4, 6.35]^6$. To reduce the potential impact of initial sample selection and optimizer randomness on the results, and to ensure experimental fairness, each system was evaluated 50 times for each test function. The 50 groups of initial samples, each consisting of 10D samples, are obtained using Latin hypercube designs (LHD) [48]. For each test case, a total of 200 updating points were added to the GEK model through the parallel EGO algorithm, with four points added in each cycle.

An object-oriented Python toolbox named Vehicular Aerodynamic Engineering Optimization in Python (Vaeopy) [56] was utilized to construct the corresponding kriging model and implement the parallel EGO algorithm. The parameter settings for constructing the OK and GEK models are specified as follows:

- Regression function: *regpoly0*.
- Correlation function: *corrgauss*.
- Feasible region of hyper-parameters: $[10^{-5}, 10^5]$.

A nonparametric statistical test [57] is employed to analyze significant differences among the comparative results. The Wilcoxon signed-rank test examines significant differences in convergence results between the two systems for each test function, with the significance level set at 0.05. A p -value below 0.05 indicates a significant performance difference between the two systems; otherwise, the difference is not significant.

To clearly illustrate the performance advantages of the proposed system, the optimization results obtained using the two comparative systems across all benchmark functions are compared. Figure 5.5 presents the Wilcoxon signed-rank test results, comparing the convergence levels of the proposed system

with the traditional system across all selected test functions. The symbols “+” and “-” indicate that the first system outperforms or underperforms the second system, respectively. R^+ and R^- represent the sum of the rankings in the Wilcoxon signed-rank test, indicating that the first group of results outperforms or underperforms the second group, respectively. The y-axis in Figure 5.5 represents the difference between the absolute values of R^+ and R^- , with a larger gap between R^+ and R^- indicating a greater performance difference between the two compared systems. From the statistical results, it is clear that the convergence level of the GEK-based parallel EGO system is significantly better than that of the OK-based parallel EGO system on almost all tested benchmark functions, except for f10. The results show that by effectively utilizing additional gradient information to construct a high-reliability surrogate model, the updating point can acquire more effective information from the high-precision surrogate model, ensuring the search for higher-quality solutions through the PEI function.

Table 5.4. Details of benchmark test functions.

Benchmark function	Property	Lower bound	Upper bound	optimum
Rotated Griewank's Function (f10)	Multi-modal; Rotated; Non-separable	[-50, 0, -60, 30, -50]	[0, 50, -10, 80, 0]	-500
Rastrigin's Function (f11)	Multi-modal; Separable; Asymmetrical	[-50, 0, -60, 30, -50]	[0, 50, -10, 80, 0]	-400
Rotated Rastrigin's Function (f12)	Multi-modal; Non-separable; Asymmetrical	[-50, 0, -60, 30, -50]	[0, 50, -10, 80, 0]	-300
Schwefel's Function (f14; f14-l)	Multi-modal; Rotated; Non-separable; Asymmetrical	[-50, 0, -60, 30, -50], [-80, -30, -90, 0, -80]	[0, 50, -10, 80, 0], [20, 70, 10, 100, 20]	-100
Composition Function (f22)	Multi-modal; Separable; Asymmetrical	[-50, 0, -60, 30, -50]	[0, 50, -10, 80, 0]	800
Frequency Modulated Sound Waves (FMSW)	Multi-Modal; Strong hierarchy	[0, 4, -2.5, 3.8, 1, 3.9]	[2, 6, -0.5, 5.8, 3, 5.9]	0

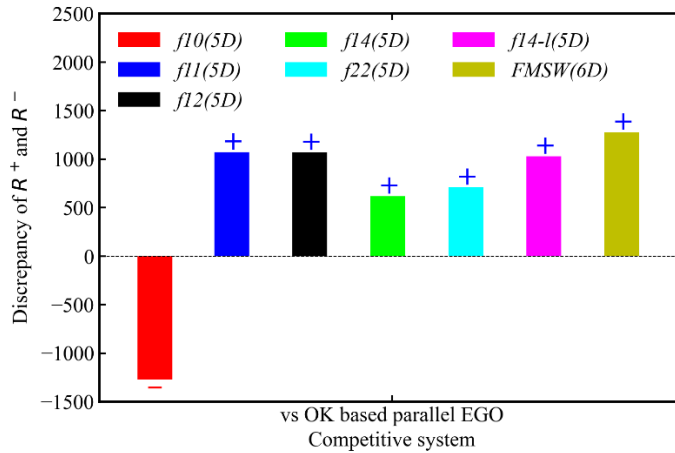


Figure 5.5. Wilcoxon signed-rank test between OK and GEK based parallel EGO system.

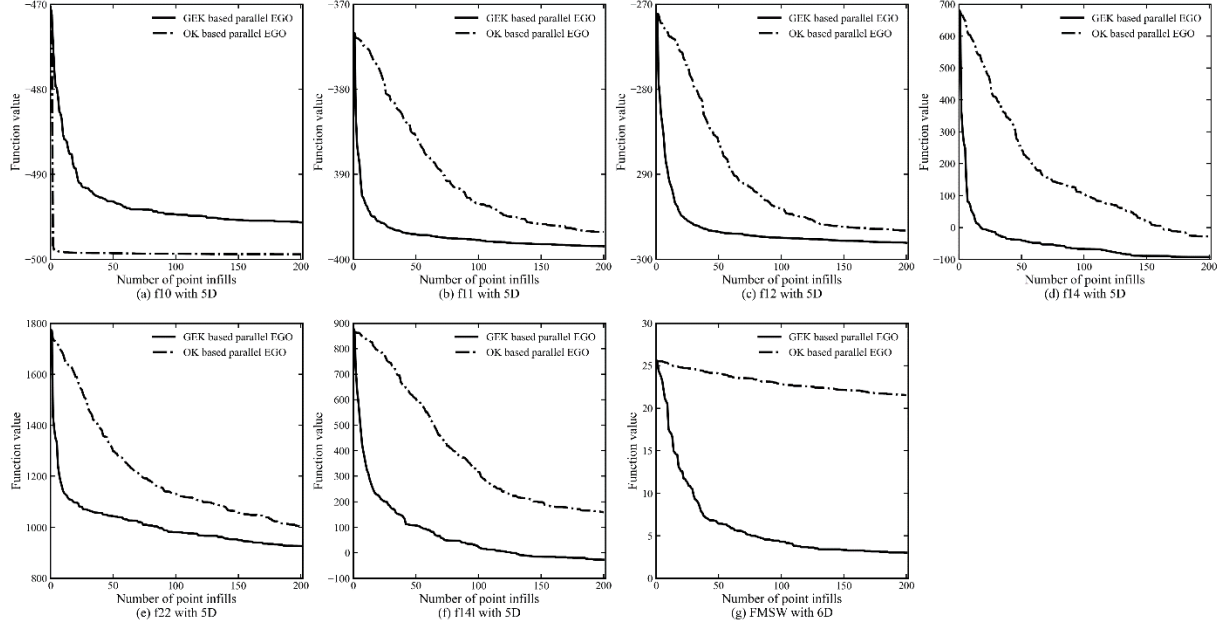


Figure 5.6. Average convergence curves of comparative systems on benchmark tested function.

To observe the convergence trend of the entire optimization process more intuitively and better highlight the difference in convergence speed between the two compared systems, the average convergence curves of different systems on the test functions are presented in Figure 5.6. The GEK-based parallel EGO system converged significantly faster than the OK-based parallel EGO system on nearly all tested functions, except for f10. Notably, on the FMSW optimization problem, the OK-based parallel EGO system achieved minimal improvement throughout the optimization process, while the GEK-based parallel EGO system effectively addressed this type of optimization problem.

5.2.2 Verification on airfoil aerodynamics optimization problem

The proposed GEK-based parallel EGO system is employed to address a real-world airfoil aerodynamics optimization problem, with its optimization performance compared to that of the OK-based parallel EGO system. To accomplish this, key details of the optimization problem are outlined, including the airfoil model, optimization objective, design variables, meshing strategy, and computational fluid dynamics (CFD) solver. The rationality of the numerical simulation was first validated, followed by a detailed discussion of the optimization performance of the proposed system in addressing the airfoil shape optimization problem.

The NACA0012, a classic symmetrical airfoil widely utilized in low Reynolds number flow and control studies [58], serves as the baseline model for shape optimization. In this study, two-dimensional

numerical simulations are conducted for airfoil shape optimization. The airfoil model has a relative thickness of 12%, with the maximum thickness positioned at 40% of the chord length from the leading edge. The chord length is set to 1 meter, as illustrated in Figure 5.7.

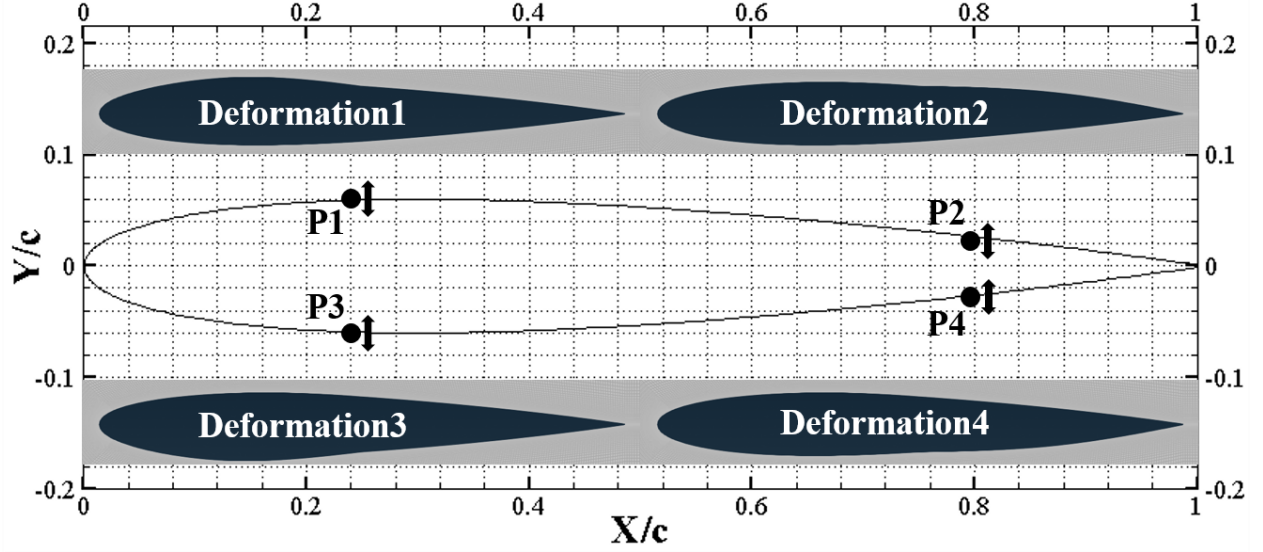


Figure 5.7. Baseline model of NACA0012 airfoil and definition of design variables.

This study selects the maximization of the lift-to-drag ratio (Cl/Cd) relative to the baseline airfoil model as the optimization objective. The lift coefficient (Cl) and drag coefficient (Cd) of the airfoil are calculated by the following formulas respectively:

$$Cl = \frac{L}{\frac{1}{2} \rho v^2 S} \quad (33)$$

$$Cd = \frac{D}{\frac{1}{2} \rho v^2 S} \quad (34)$$

Here, L and D represent the lift and drag forces generated by the object in the airflow, measured in Newtons (N). ρ denotes the fluid density (kg/m^3), v is the freestream velocity (m/s), and S refers to the object's reference area (m^2).

To enhance the aerodynamic performance of the airfoil while minimizing computational costs, the design variables are carefully selected to modify the shape of the airfoil model. Yu et al. [59] reported that the aerodynamic performance of an airfoil is predominantly affected by the pressure differences between its leading and trailing edges, as well as between the upper and lower surfaces. Therefore, four control

points were defined as design variables, located at 23.62%c and 76.38%c along both the upper and lower surfaces of the airfoil, measured from the leading edge.

To control the airfoil shape automatically, the Direct Morphing (DFM) module in ANSA v23 is utilized to perform shape deformation based on the original airfoil configuration. Figure 5.7 illustrates the definition of each design variable and the corresponding deformed models. The deformation process is managed by adjusting the positions of control points, represented by black solid circles, along the y-direction. Table 5.5 specifies the feasible range for each design variable. To maintain the integrity of the airfoil shape after deformation, appropriate upper and lower bounds are established for the design variables.

To obtain the Cl/Cd of the airfoil models before and after deformation, CFD solver Fluent in ANSYS 2021 R1, a commercial CFD software, running on the computer along with the $K - \omega - SST$ turbulence model was used to perform the numerical simulation. The computer is equipped with an i9-12900K processor, which operates at a base frequency of 3.20 GHz and has 16 cores. The details of the computational domain and mesh generation are as follows: the distance between the far-field boundary of the computational domain and the airfoil surface is set to 50c [60]. A C-H type structured grid is adopted as the mesh strategy. To capture the details of the flow field around the airfoil, the mesh has a growth rate of 1.1, and the first layer height is set to 0.015 mm to ensure that the y^+ value is controlled to be less than 1, which is necessary for the turbulence model used [61]. The total number of mesh elements is 106,752.

Table 5.5. Details of design variables.

Deformation position	Variable	Lower bound (mm)	Upper bound (mm)
23.62% (upper edge)	P1	-20	+20
76.38% (lower edge)	P2	-20	+20
23.62% (upper edge)	P3	-20	+20
76.38% (lower edge)	P4	-20	+20

Reasonable boundary conditions and solver settings are crucial for accurately solving the external flow field around the airfoil. In this study, the inlet velocity is set to 42.0245m/s ($Re = 3.0 \times 10^6$), and the incoming flow angle of attack is 10° . The outlet pressure is fixed at zero. The airfoil surface boundary is set to a no-slip condition. The simulation is conducted using the semi-implicit method of pressure-linked equations (SIMPLE) algorithm with GAME for pressure and "smoothSolver" for the transport parameters. The number of iterations is set to 500 for each simulation, and the convergence tolerance for each solution variable is set to 1×10^{-6} .

To verify the reliability of the numerical simulation method, the results obtained from the original model used in this study are compared with those from previous studies [62] (NASA Langley Turbulence Modeling Resource website). The comparative results are shown in Figure 5.8. It is evident that the variation trend of the pressure coefficient (C_p) on the airfoil surface in this study aligns closely with the results from previous studies. Additionally, Table 5.6 presents the C_l and C_d values obtained in this and previous studies. The computational error is within acceptable limits. In conclusion, the numerical simulation method employed in this study is reliable and suitable for subsequent optimization studies.

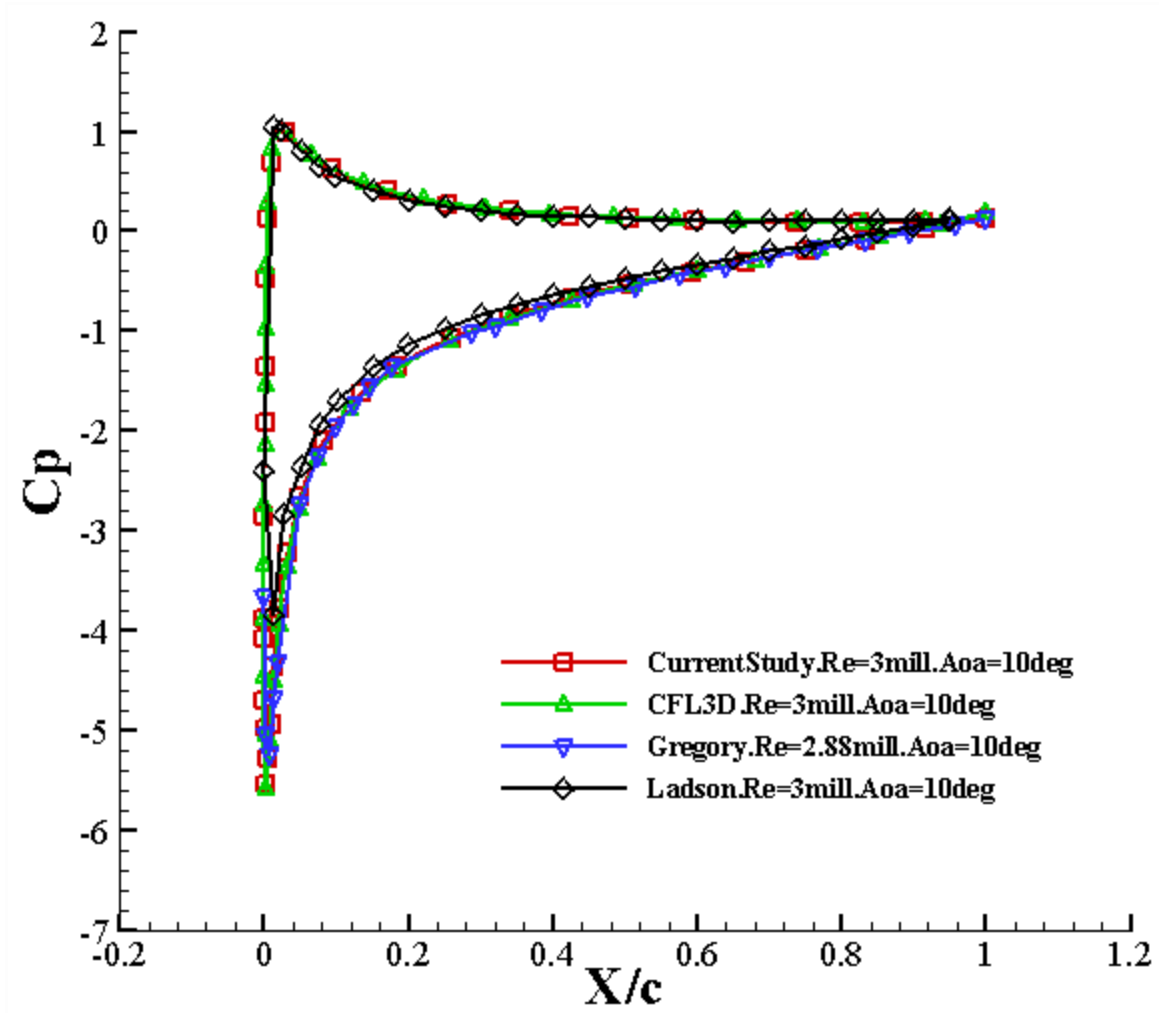


Figure 5.8. Validation of the C_p value of airfoil surface with previous studies [62].

Table 5.6. Validation of aerodynamic characteristics of Cl and Cd of airfoil with previous studies [62].

Data source	Cl	Cd
In this study	1.04994	0.01590
CFL3D	1.07781	0.01236
Gregory	0.98488	--
Ladson	1.07070	0.01201

To efficiently apply the proposed optimization system for airfoil shape optimization, a series of key processes must be performed sequentially. These processes include sampling, evaluation, construction of the GEK surrogate model, and the point infill criterion to drive global convergence during the optimization process. Five initial sample points were obtained using the LHD sampling method. Numerical simulations were conducted to obtain the Cl and Cd values for each sample point. The OK and GEK-based parallel EGO algorithm systems were repeatedly run until a stopping criterion was met, respectively. It is important to note that real-world optimization problems are often constrained by development time and high costs. Thus, achieving the maximum possible improvement within a limited time, rather than focusing solely on obtaining a global optimum, better aligns with the practical needs of optimization in real-world scenarios. In consideration of this, a total of 20 point infills (4 updating points per cycle) were selected to evaluate the performance of the proposed system. Another important aspect to note is that the Cl/Cd value of the original airfoil model was 66.03396, while the maximum Cl/Cd value in the five initial samples was 70.07862. Therefore, the actual improvement in the Cl/Cd value during the optimization process is measured based on the maximum objective value from the initial samples, meaning that a higher Cl/Cd value than 70.07862 must be achieved. Additionally, the required gradient information for constructing the GEK is obtained using the finite difference method. The differential step length is set to $1 \times 10^{-3}\text{m}$, which yielded the best results in the tests. The specific settings for constructing the OK and GEK models are the same as those described in section 5.2.1.

To comprehensively evaluate the performance advantages of the proposed optimization system, the improvement in the Cl/Cd ratio with different systems as the number of point infills increases is presented in Figure 5.9. The proposed system converges more efficiently than the traditional system. Notably, the former system identified a high-quality solution at the first updating point and achieved continuous improvement in the Cl/Cd ratio, increasing it from 70.07862 to 72.04189 throughout nearly the entire optimization process. In contrast, the latter system did not identify an improved solution during the first point infill cycle and showed only limited improvement over the first 12 updating points. Table 5.7 presents the details of the optimal solutions obtained by two compared systems. The final improvement in the Cl/Cd

ratio achieved by the proposed system and the traditional system was 2.80% and 2.39%, respectively, compared to the baseline airfoil model.

Table 5.7. Optimal solutions obtained by GEK and OK based parallel algorithm system.

System	P1 (mm)	P2 (mm)	P3 (mm)	P4 (mm)	Cl	Cd	Cl/Cd
GEK based parallel EGO	19.37	2.90	20.00	20.00	1.18653	0.01647	72.04189
OK based parallel EGO	16.21	-2.97	20.00	20.00	1.16959	0.01630	71.75399

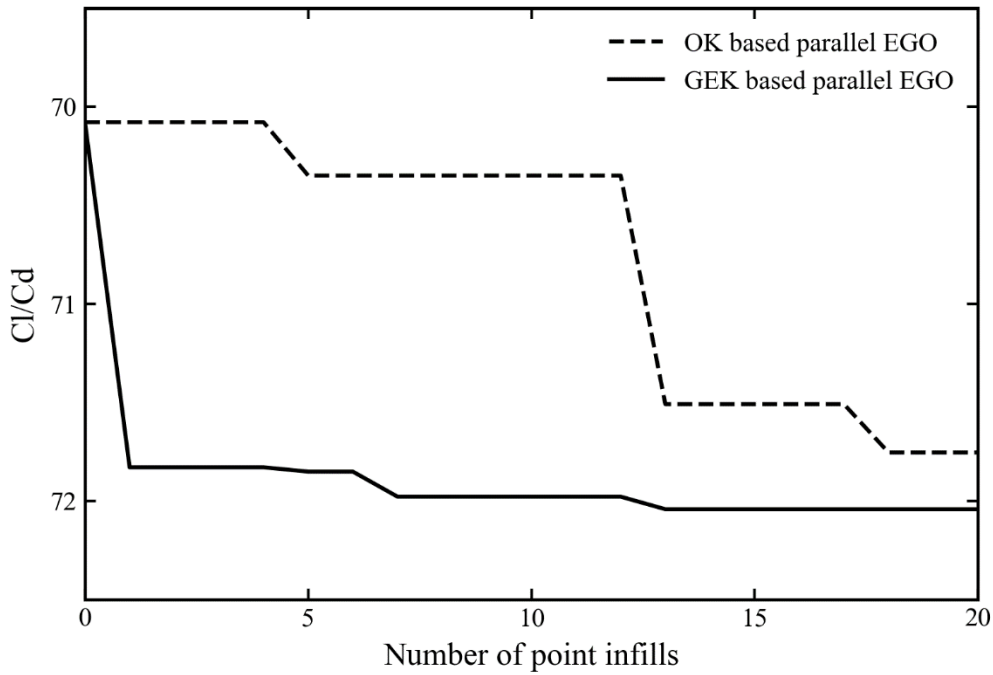


Figure 5.9. Convergence curves of Cl/Cd ratio with GEK and OK based parallel EGO algorithm system, respectively.

To better understand the mechanism behind the improvement in the Cl/Cd ratio following airfoil shape optimization, a comparative analysis of the pressure distribution on the airfoil surface before and after optimization was conducted. Figure 5.10 illustrates the surface pressure distribution for both the baseline and optimized airfoils using different optimization systems. The optimal solutions obtained by both systems

resulted in significant positive changes in the control variables P1, P3, and P4 along the y-direction. As a result, the pressure around these control points also underwent substantial changes compared to the baseline airfoil. The primary reason for the improvement in the Cl/Cd ratio is the enhanced pressure difference between the upper and lower surfaces of the airfoil, which significantly increases the lift generated by the airfoil. To more intuitively observe the flow field changes resulting from airfoil shape modifications during the optimization process, Figure 5.11 presents the corresponding pressure contours for the baseline airfoil model and the optimized airfoil models at the 1st, 7th, 13th, and 19th solutions using different optimization systems. The pressure contours clearly show that significant changes in the control points correspond to notable variations in the pressure distribution.

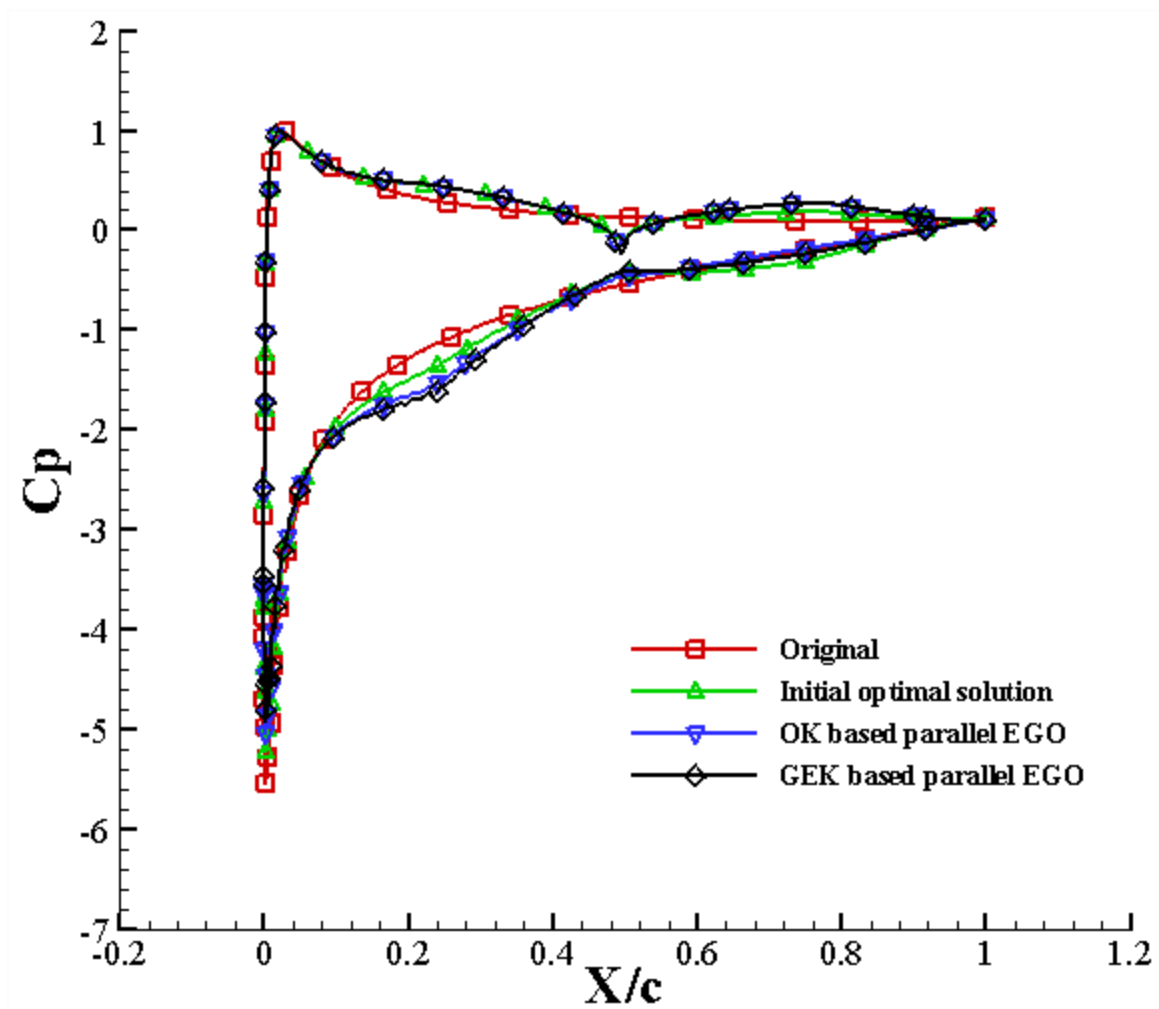


Figure 5.10. Surface pressure coefficient distribution of the baseline airfoil model and the optimized airfoil model using GEK and OK based parallel algorithm system, respectively.

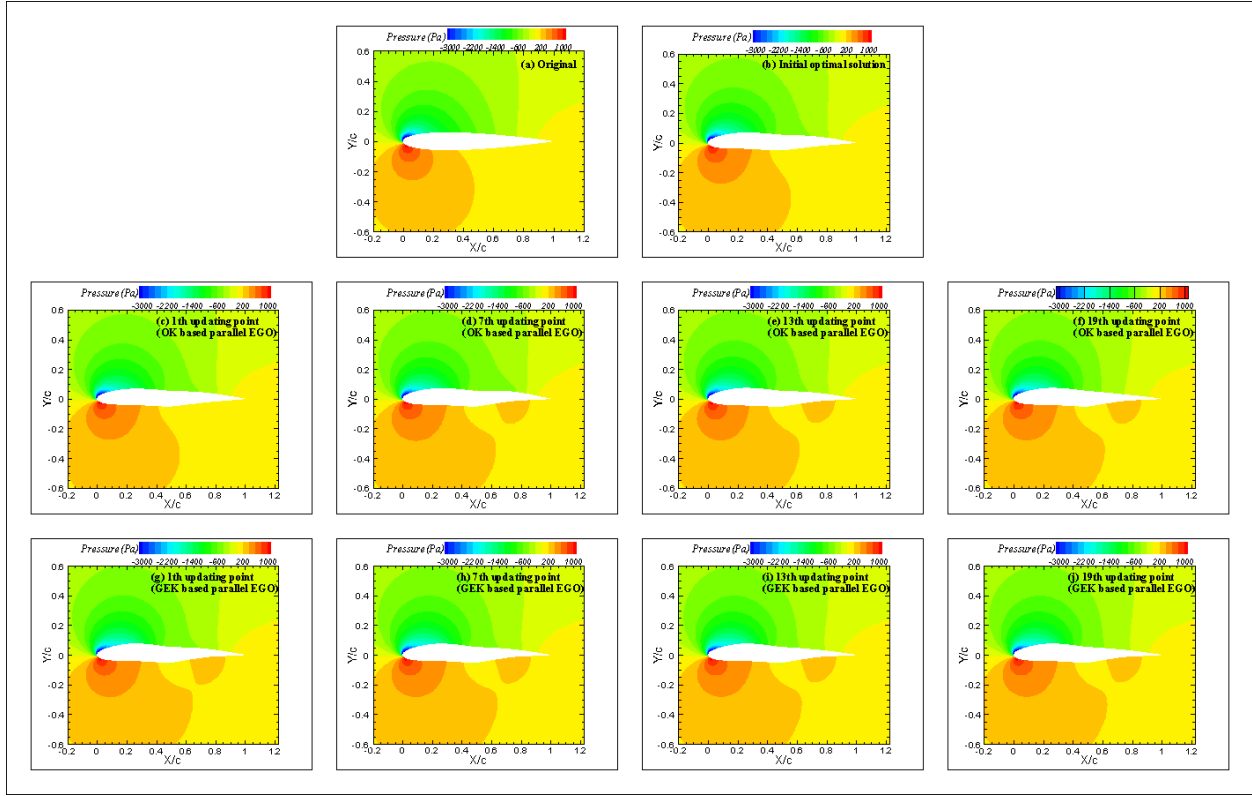


Figure 5.11. Pressure contour of the baseline airfoil model and the optimized airfoil model using GEK and OK based parallel algorithm system, respectively.

5.2.3 Verification on vehicle aerodynamics optimization problem

To demonstrate the performance advantages of the proposed algorithm system in more complex engineering optimization problems, this section presents a comparative investigation of another real-world optimization problem in vehicle aerodynamics.

To efficiently and rapidly validate the performance advantages of the proposed improved algorithm system in automotive aerodynamic optimization, a high-precision neural network model [63], trained on a large high-fidelity dataset, was employed to evaluate the objective. The dataset for the high-fidelity model was constructed using aerodynamic parameters, such as drag, lift, and changed drag, of a full-scale SUV vehicle shape. The baseline vehicle shape and the 10 deformed vehicle shapes, obtained through parametric deformation using the commercial pre-processing software "DEP MeshWorks," are shown in Figure 5.12. A total of 29 design variables were selected to control the vehicle shape. A total of 232 and 123 sample points were chosen from the design space using the OLHD method and the optimization algorithm, respectively. These objective values were efficiently and accurately evaluated using the 'Fugaku' supercomputer [64] and the 'CUBE' numerical simulation method [65]. In this case, the inlet velocity was

set to 27.78 m/s toward the vehicle model, and the Reynolds number was $Re = 5.83 \times 10^6$ based on the original vehicle length, inlet velocity, and fluid viscosity. The numerical simulation was conducted using the “Fractional-step method,” “Crank-Nicolson method,” and “Quick scheme.” For grid resolution, the minimum grid size was 7 mm, and the total number of grids was 201 million. The total numerical simulation time was 2 s, with the time step set to 2.5×10^{-5} s.

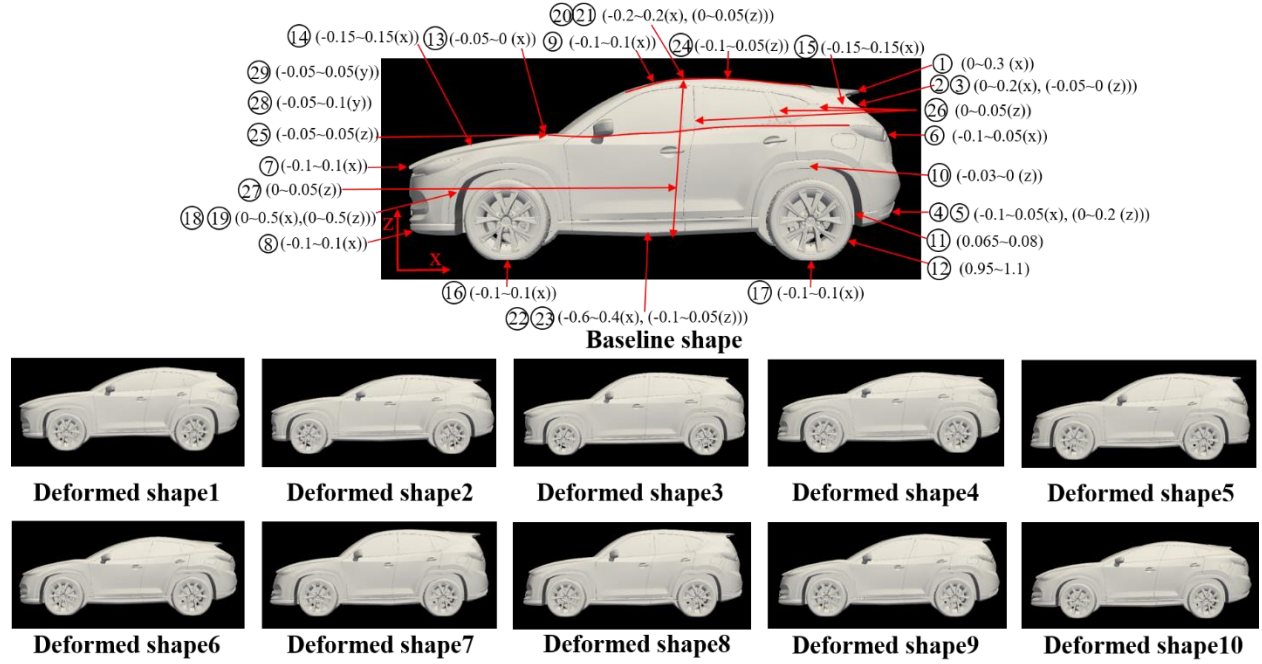


Figure 5.12. Baseline vehicle shape, and deformed 10 vehicle shapes based on definition of design variables.

A fully connected neural network model was trained using 355 high-fidelity original datasets, with the 'ReLU' activation function applied. Different training parameters were compared to determine the optimal configuration. Both the input layer and hidden layer were set to 30 nodes, which provided the best performance. The output layer consisted of 3 nodes, each representing a predicted objective value for the aerodynamic performance of the vehicle.

The predictive accuracy and reliability of the neural network model were validated using 10 vehicle models, as mentioned in the previous section. In this study, only the C_d (calculating by Eq. (34)) is considered as the optimization objective for validating the performance of the proposed algorithm system. Therefore, Figure 5.13 shows the validation of the predicted C_d values from the neural network model and the numerical simulation experiment. It can be observed that the predicted C_d values align well with the numerical simulation results. Both experimental and predicted C_d values were normalized. The average prediction error is 0.52%, which is within the acceptable range for practical engineering applications.

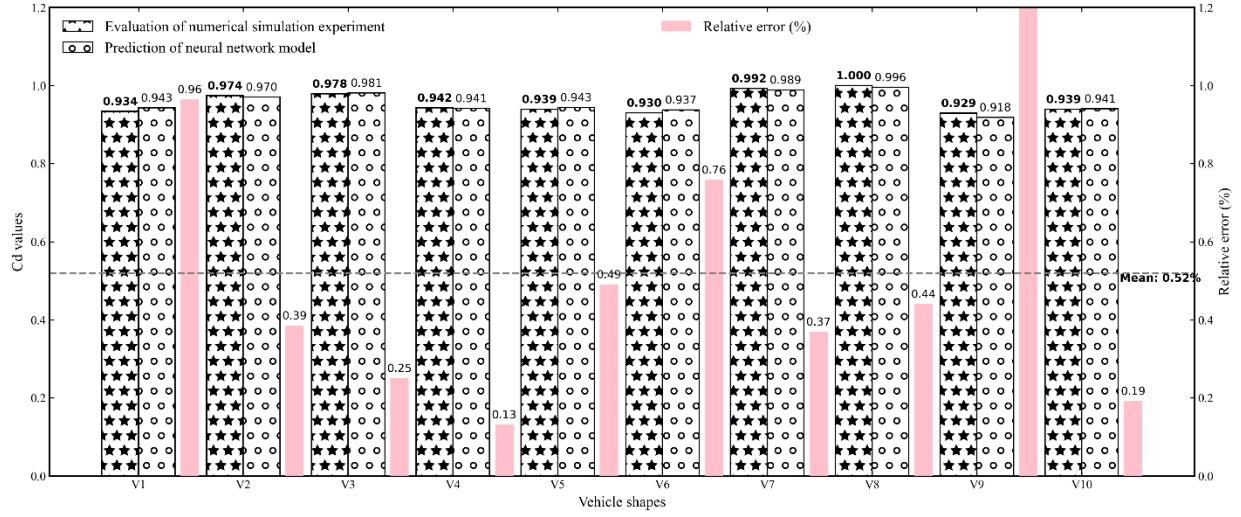


Figure 5.13. Validation of the predicted Cd values of the neural network model and numerical simulation experiment, respectively.

In the current study, only the Cd value is considered as the optimization objective for validating the performance of the proposed algorithm system. Therefore, when evaluating the objective values of the sample points using the constructed neural network model, the focus is solely on the Cd value. The LHD method was used to select 10 initial sample points from the design space, which were then used to construct OK and GEK surrogate models after obtaining the Cd value of each sample point. To obtain the gradient information required for constructing the GEK surrogate model, two methods were employed. The first method is the finite difference method (with a step size of 0.001). The second method leverages the training characteristics of the neural network model, enabling the efficient acquisition of gradient information for each dimension of each sample point through automatic differentiation techniques. Figure 5.14 presents the gradient information obtained using both methods. As shown, the gradient information derived from the two approaches is generally consistent. During the optimization process, each of the compared algorithm systems filled 60 new sample points, with 4 new sample points added in each iteration, following the parallel point infill criterion.

Figure 5.15 presents the normalized convergence curves of the parallel EGO optimization algorithm systems based on the OK and GEK surrogate models, respectively. From the figure, it can be observed that the parallel EGO algorithm system based on the OK surrogate model fails to find an improved solution during the early optimization stage due to the limited effective information provided. It only identifies the first improved solution after the 32nd sample is added, resulting in a relatively slow convergence speed. In contrast, the system based on the GEK surrogate model is able to identify an improved solution even in the early optimization stage, leveraging the limited information available. Moreover, as more sample points are

added, the GEK-based system rapidly and consistently converges towards the optimal objective. Notably, within the current limited number of sample points, the GEK-based system ultimately converges to higher-quality solutions, while the OK-based system converges to a lower-quality solution. Specifically, the final maximum drag reduction rate achieved is 12.97%, relative to the initial optimum.

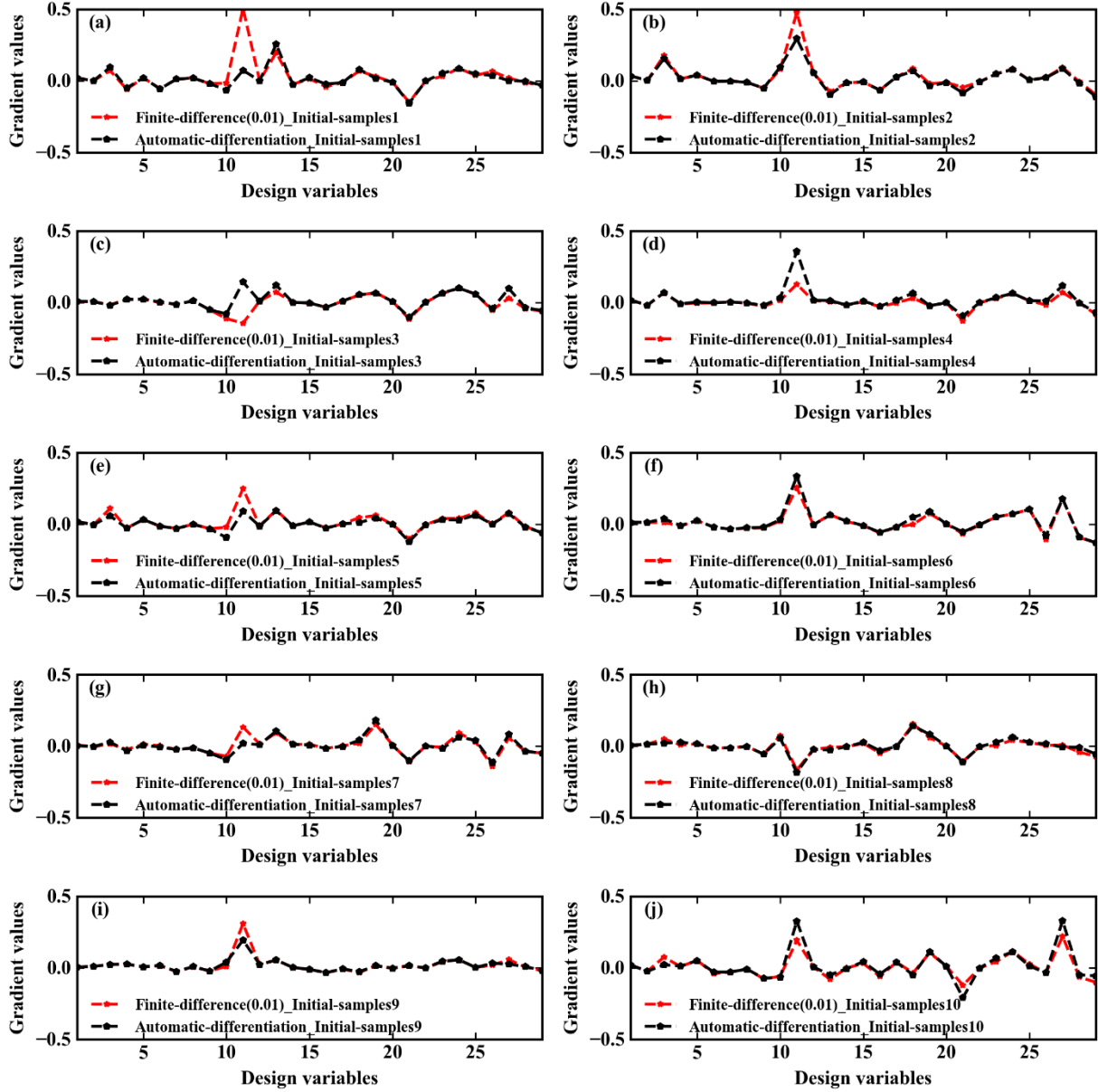


Figure 5.14. Validation of the obtained gradient information of finite difference and automatic differentiation technique, respectively.

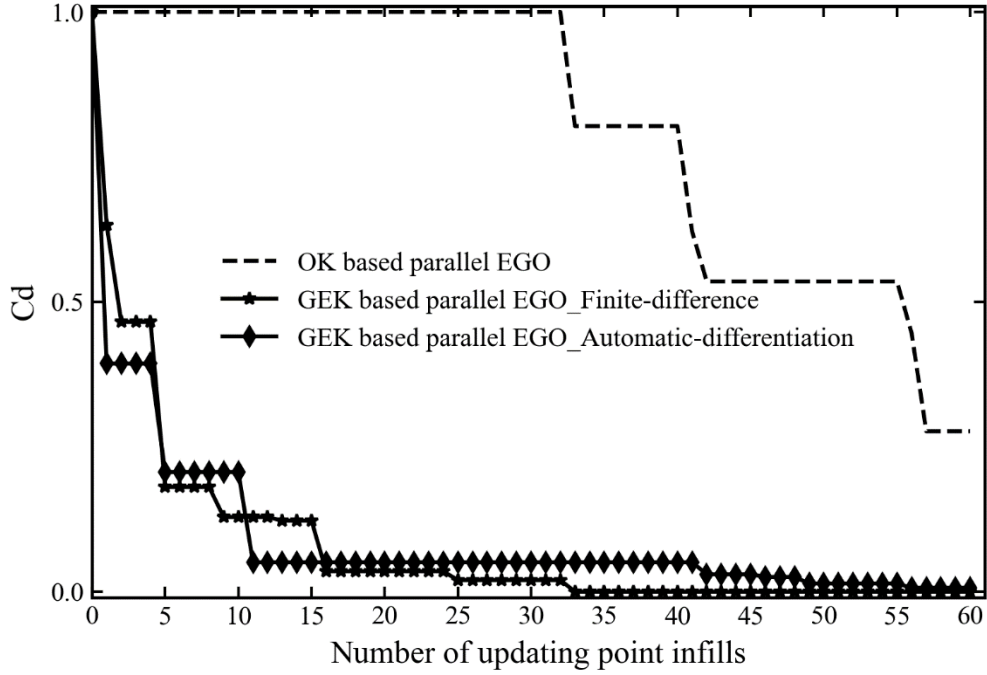


Figure 5.15. Convergence curves of Cd values with GEK and OK based parallel EGO algorithm system, respectively.

5.2.4 Quantitative analysis on real-world optimization problem of aerodynamics

To validate the performance advantages of the proposed improved parallel EGO optimization system in practical engineering optimization problems, experimental verification was conducted in sections 5.2.2 and 5.2.3. From the convergence curves in Figures 5.9 and 5.15, it was observed that the proposed method achieves faster convergence than the traditional parallel EGO optimization system. Additionally, under the same number of sample point infills, the proposed method identifies higher-quality approximate optimal solutions.

To further analyze its performance advantages, a quantitative comparison was conducted in terms of samples cost, optimization cycles, computational resource cost, and total wall-clock time cost required to reach a predefined optimization target. Firstly, it is essential to understand the main components of wall-clock time cost. In our case, the wall-clock time cost for a parallel infill process consists of three parts: mesh preparation, waiting gap, and simulation, as shown in Figure 5.16. It is worth noting that updating the surrogate model and searching for new sample points based on the infill criterion require significantly less time than other steps, making their wall-clock time cost negligible. Specifically, the wall-clock time cost

per parallel infill process for the airfoil and vehicle shape optimization problems is 0.33 h and 14.5 h, respectively. Table 5.8 shows a quantitative comparison of optimization costs between the proposed improved optimization system and the traditional parallel EGO optimization system. In the airfoil shape optimization problem, the traditional parallel EGO optimization system requires 18 samples infills, corresponding to 5 parallel optimization cycles, to reach the optimization target. In contrast, the GEK-based parallel EGO optimization system achieves the same target with only one samples infill and a single optimization cycle. Furthermore, the GEK-based system has a lower wall-clock time cost. For the vehicle shape optimization problem, the traditional parallel EGO optimization system requires 56 samples infills, equivalent to 14 optimization cycles, whereas the GEK-based system reaches the target with only 2 samples infills and a single optimization cycle. Additionally, the GEK-based system significantly reduces the wall-clock time cost compared to the traditional parallel EGO optimization system.

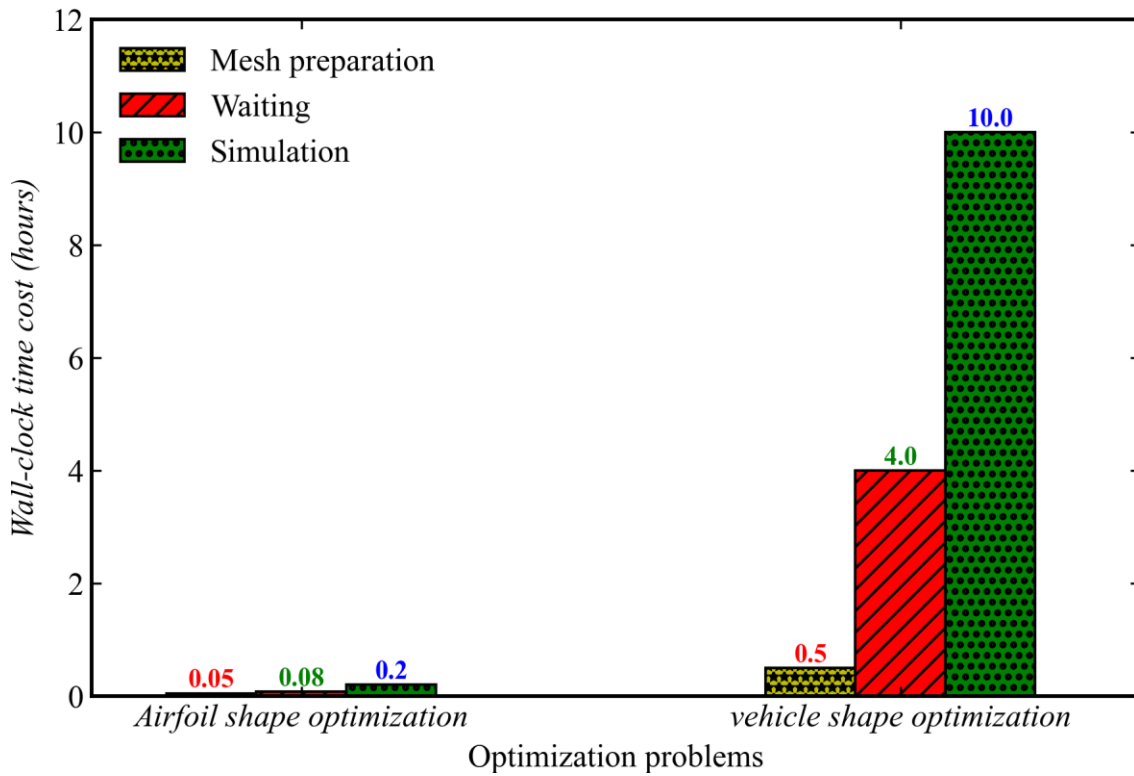


Figure 5.16. Wall-clock time cost components of each optimization cycle.

Table 5.8. Quantitative comparison of performance of the GEK and OK based parallel algorithm system in optimization process.

Optimized target	Improved aerodynamic performance (Case C: 2.4%)				Improved aerodynamic performance (Case D: 7.0%)			
Performance Indicators	Cost of samples	Execute optimization cycles	Cost of evaluation objective	Cost of Wall-clock time (h)	Cost of samples	Execute optimization cycles	Cost of evaluation objective	Cost of Wall-clock time (h)
OK based parallel EGO	18	5	18	1.65	56	14	56	203
GEK based parallel EGO	1	1	1	0.33	2	1	2	14.5

CHAPTER 6. CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions

In this study, to fully exploit the performance advantages of the parallel EGO algorithm, an improved algorithm system called the GEK-based parallel EGO algorithm system was proposed. This system is derived from the OK-based parallel EGO algorithm system by introducing a powerful surrogate model called GEK. It effectively utilizes gradient information to enhance the construction of a high-fidelity surrogate model, improving the reliability of the EGO algorithm in identifying potential improvement points based on the parallel infill criterion. First, optimization performance tests were conducted on benchmark functions to ensure the robustness of the proposed system and identify the most suitable optimizer. Based on this, performance comparisons and analyses were conducted between the proposed method and the traditional parallel EGO optimization system on benchmark functions and two aerodynamic shape optimization applications. Second, this study also investigated the impact of varying sample sizes and space-filling quality levels of sample points design on the optimization performance of GEK based parallel EGO algorithm system. By categorizing the space-filling quality of sample points obtained from LHD and OLHD sampling methods. Third, this study also conducted an in-depth investigation into how the optimization performance of the kriging-based parallel EGO algorithm with the PEI multi-point sampling criterion is affected by different numbers of points filled in a single optimization cycle. Finally, from the analysis and discussion of the results, the following conclusions were drawn:

- 1) The hybrid MGBDE optimizer, with its stronger balance between exploration and exploitation, achieved the best performance on the proposed algorithm system. Therefore, for real-world engineering optimization problems, it is recommended that the MGBDE optimizer be used as the core engine to search for potential improvement points.
- 2) The proposed improved system demonstrates significant improvement in convergence performance compared to the original system on nearly all tested benchmark functions and two real-world engineering optimization problems.
- 3) Quantitative analysis of airfoil and automotive shape optimization cases demonstrates that the proposed algorithm system significantly reduces wall-clock time compared to traditional methods, highlighting its potential for real-world engineering applications.
- 4) A gradual increase in sample size does not consistently improve the convergence performance of the current optimization system when the space-filling quality of the sample points remains unchanged.
- 5) Under the same sample size, the incomplete level of space-filling quality in OLHD effectively accelerates the convergence of the current optimization system, enabling it to reach higher-quality

solutions within a limited number of iterations.

- 6) The impact of different point-filling quantities on the optimization performance of the parallel EGO algorithm based on the OK model is significant. Filling 2 points per optimization cycle ensures the most stable optimization performance of the parallel EGO algorithm. However, as the number of filled points increases, the stability of the optimization performance gradually declines. In contrast, for the GEK model, the influence of different point-filling quantities on the optimization performance of the parallel EGO algorithm is only noticeable in the early stages of optimization. As the optimization process progresses, the performance differences gradually diminish.
- 7) For the parallel EGO algorithm based on the OK model, optimization efficiency improves as the point-filling quantity increases. However, the rate of improvement gradually slows with the increasing number of filled points. In the case of GEK, this slowdown in optimization efficiency becomes even more pronounced.
- 8) The point-filling quantity has a significant impact on the optimization diversity of the parallel EGO algorithm based on the OK model. In most cases, filling 2 points per optimization cycle ensures a reasonable level of optimization diversity and maintains a balance between local exploitation and global exploration. In contrast, for the GEK model, the influence of different point-filling quantities on the diversity of the parallel EGO algorithm is less pronounced.

6.2 Recommendations for Future Work

Notably, while the proposed system significantly improves optimization performance over the traditional parallel EGO system, certain limitations must be addressed for practical engineering applications. These limitations can be categorized into two types. The first arises when the optimization problem exhibits strong nonlinearity, leading to noise in numerical simulation results. This noise may cause overfitting in the GEK model, restricting its generalization ability. The second limitation stems from the additional computational cost required to obtain gradient information for the initial GEK model. When the number of design variables is high, obtaining gradients via traditional finite difference method can become computationally prohibitive. Overcoming challenge requires improving gradient computation efficiency. Additionally, future work will also focus on further exploring the influence of sample point properties on the optimization performance of the current optimization system across a broader range of sample dimensions, sample sizes, and different types of benchmark functions. Furthermore, further research will also focus on developing more efficient and stable parallel point-filling strategies to better meet the needs of practical engineering optimization problems. Finally, future research will also investigate the integration of deep learning-based surrogate models with hardware acceleration techniques to improve computational efficiency, with the ultimate goal of applying these advancements to real-world engineering optimization problems.

REFERENCES

- [1] MARIO Bunge: A General Black Box Theory. *Philosophy of Science*, 30(4), 346–358, 1963
- [2] ATHARV Bhosekar, MARIANTHI Ierapetritou: Advances in surrogate based modeling, feasibility analysis, and optimization: A review. *Computers and Chemical Engineering*, 108, 250-267, 2018
- [3] T. W. Simpson, J. D. Peplinski, P. N. Koch, J. K. Allen: Metamodels for Computer-based Engineering Design: Survey and recommendations. *Engineering with Computers*, 17, 129-150, 2001
- [4] VISHAL Raul, LEIFUR Leifsson: Surrogate-based aerodynamic shape optimization for delaying airfoil dynamic stall using Kriging regression and infill criteria. *Aerospace Science and Technology*, 111, 106555, 2021
- [5] BEI Liu, HUA Liang, ZhONG-HUA Han, GUANG Yang: Surrogate-based aerodynamic shape optimization of a sliding shear variable sweep wing over a wide Mach-number range with plasma constraint relaxation. *Structural and Multidisciplinary Optimization*, 66, 43, 2023
- [6] GORKEM Demir, RECEP M. Gorguluarslan, SELIN Aradag: Robust shape optimization under model uncertainty of an aircraft wing using proper orthogonal decomposition and inductive design exploration method. *Structural and Multidisciplinary Optimization*, 66, 93, 2023
- [7] ATTHAPHON Ariyarat, MASAHIKO Sugiura, YASUTADA Tanabe, MASAHIRO Kanazaki: Hybrid surrogate-model-based multi-fidelity efficient global optimization applied to helicopter blade design. *Engineering Optimization*, 50(6), 1016–1040, 2017
- [8] MAGNUS Urquhart, MAX Varney, SIMONE Sebben, MARTIN Passmore: Aerodynamic drag improvements on a square-back vehicle at yaw using a tapered cavity and asymmetric flaps. *International Journal of Heat and Fluid Flow*, 86, 108737, 2020
- [9] ZHAO He, XIAOHUI Xiong, BO Yang, HAIHONG Li: Aerodynamic optimisation of a high-speed train head shape using an advanced hybrid surrogate-based nonlinear model representation method. *Optimization and Engineering*, 23, 59-84, 2022
- [10] Séverin Tinmitondé, XUHUI He, LEI Yan: Single-objective aerodynamic optimization of a streamlined bridge deck subjected to shape modification using a polynomial emulator and genetic algorithm. *Structural and Multidisciplinary Optimization*, 65, 356, 2022
- [11] XIANLONG Zhou, WEILONG Guo, XIANGYU Shi, CHUNLING She, ZHOUYUAN Zheng, JIE Zhou, YUSONG Zhu: Machine learning assisted multi-objective design optimization for battery thermal management system. *Applied Thermal Engineering*, 253, 123826, 2024

- [12] W. H. Hucho, L. J. Janssen, H. J. Emmelmann: The Optimization of Body Details-A Method for Reducing the Aerodynamic Drag of Road Vehicles. SAE Technical Paper 760185, 1976
- [13] PRAMOD NARI Krishnani, DONGMEI Zhou: CFD analysis of drag reduction for a generic SUV. Proceeding of the ASME 2009 International Mechanical Engineering Congress & Exposition (IMECE2009), November 13-19, Lake Buena Vista, Florida, USA, 10170, 589-598, 2009
- [14] YI Liu, JIANG Chen, JINXIN Cheng, HANG Xiang: Aerodynamic Optimization of Transonic Rotor Using Radial Basis Function Based Deformation and Data-Driven Differential Evolution Optimizer. Aerospace, 9, 508, 2022
- [15] TAKUYA Kojima, NGUYEN ANH VU Doan, HIDEHARU Amano: A Genetic Algorithmic Approach for Optimizing Spatial Mapping of Coarse-Grained Reconfigurable Architectures. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 28(11), 2383-2396, 2020
- [16] ZHAO He, TANGHONG Liu, HUI Liu: Improved particle swarm optimization algorithms for aerodynamic shape optimization of high-speed train. Advances in Engineering Software, 173, 103242, 2022
- [17] ZAIN ANWAR Ali, ZHANGGANG Han, RANA JAVED Masood: Collective Motion and Self-Organization of a Swarm of UAVs: A Cluster-Based Architecture. Sensors, 21, 3820, 2021
- [18] ALI KHATER Mohamed, ALI WAGDY Mohamed: Real-parameter unconstrained optimization based on enhanced AGDE algorithm. In: Hassanien, A. (ed) Machine Learning Paradigms: Theory and Application. Studies in Computational Intelligence, Springer, Cham., 801, 2019
- [19] SHANGCE Gao, YANG Yu, YIRUI Wang, JIAHAI Wang, JIUIJUN Cheng, MENGCHU Zhou: Chaotic local search-based differential evolution algorithms for optimization. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 51(6), 3954-3967, 2021
- [20] HUI Wang, SHAHRYAR Rahnamayan, HUI Sun, MAHAMED G. H. Omran: Gaussian bare-bones differential evolution. IEEE Transactions on Cybernetics, 43(2), 634-647, 2013
- [21] BENYAMIN Abdollahzadeh, FARHAD SOLEIMANIAN Gharehchopogh, SEYEDALI Mirjalili: Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. International Journal of Intelligent Systems, 36, 5887-5958, 2021
- [22] MUHAMMAD Shafiq, ZAIN ANWAR Ali, AMBER Israr, EMAN H. Alkhamash, MYRIAM Hadjouni, JARI JUHANI Jussila: Convergence Analysis of Path Planning of Multi-UAVs Using Max-Min Ant Colony Optimization Approach. Sensors, 22, 5395, 2022

- [23] ADAM Slowik, HALINA Kwasnicka: Evolutionary algorithms and their applications to engineering problems. *Neural Computing and Applications*, 32, 12363–12379, 2020
- [24] G. GARY Wang, SONGQING Shan: Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical Design*, 129(4), 370-380, 2007
- [25] JEROME Sacks, WILLIMN J. Welch, TOBY J. Mitchell, HENRY P. Wynn: Design and analysis of computer experiments. *Statistical Science*, 4(4), 409-423, 1989
- [26] ALEXANDER I. J. Forrester, ANDY J. Keane, NEIL W. Bressloff: Design and analysis of “noisy” computer experiments. *AIAA Journal*, 44(10), 2331-2339, 2006
- [27] J. Laurenceau, P. Sagaut: Building efficient response surfaces of aerodynamic functions with kriging and cokriging. *AIAA Journal*, 46(2), 498-507, 2008
- [28] MOHAMEDo A. Bouhlel, JOAQUIM R. R. A. Martins: Gradient-enhanced kriging for high-dimensional problems. *Engineering with Computers*, 35, 157-173, 2019
- [29] LIMING Chen, HAobo Qiu, LIANG Gao, CHEN Jiang, ZAN Yang: Optimization of expensive black-box problems via Gradient-enhanced Kriging. *Computer Methods in Applied Mechanics and Engineering*, 362, 112861, 2020
- [30] LUC Laurent, RODOLPHE LE Riche, BRUNO Soulier, PIERRE-ALAIN Boucard: An Overview of Gradient-Enhanced Metamodels with Applications. *Archives of Computational Methods in Engineering*, 26, 61-106, 2019
- [31] GERARDO Raggi, IGNACIO FDEZ. Galván, CHRISTIAN L. Ritterhoff, MORGANE Vacher, ROLAND Lindh: Restricted-variance molecular geometry optimization based on gradient-enhanced kriging. *Journal of Chemical Theory and Computation*, 16(6), 3989-4001, 2020
- [32] FAN Cao, ZHILI Tang, CAICHENG Zhu, XIN Zhao: An Efficient Hybrid Multi-Objective Optimization Method Coupling Global Evolutionary and Local Gradient Searches for Solving Aerodynamic Optimization Problems. *Mathematics*, 11, 3844, 2023
- [33] ZHONG-HUA Han, STEFAN Görtz: Hierarchical kriging model for variable-fidelity surrogate modeling. *AIAA Journal*, 50(9), 1885-1896, 2012
- [34] ZHONG-HUA Han, STEFAN Görtz, RALF Zimmermann: Improving variable-fidelity surrogate modeling via gradient-enhanced kriging and a generalized hybrid bridge function. *Aerospace Science and Technology*, 25(1), 177-189, 2013

- [35] DONALD R. Jones, MATTHIAS Schonlau and WILLIAM J. Welch: Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13, 455-492, 1998
- [36] MSAHIRO Kanazaki, TAKASHI Matsuno, KENGO Maeda, HIROMITSU Kawazoe: Efficient global optimization applied to wind tunnel evaluation-based optimization for improvement of flow control by plasma actuators. *Engineering Optimization*, 47(9), 1226-1242, 2014
- [37] SHENGGUAN Xu, HONGQUAN Chen, JIALE Zhang: A study of Nash-EGO algorithm for aerodynamic shape design optimizations. *Structural and Multidisciplinary Optimization*, 59, 1241-1254, 2018
- [38] DAVID Ginsbourger, RODOLPHE LE Riche, LAURENT Carraro : Kriging is well-suited to parallelize optimization. In: Tenne, Y., Goh, CK. (eds) *Computational Intelligence in Expensive Optimization Problems. Adaptation Learning and Optimization*, Springer, Berlin, Heidelberg, 2, 2010
- [39] DAWEI Zhan, JIACHANG Qian, YUANSHENG Cheng: Pseudo expected improvement criterion for parallel EGO algorithm. *Journal of Global Optimization*, 68, 641-662, 2017
- [40] BRUNO Vuillod, MATHILDE Zani, LUDOVIC Hallo, MARCO Montemurro: Handling noise and overfitting in surrogate models based on non-uniform rational basis spline entities. *Computer Methods in Applied Mechanics and Engineering*, 425, 116913, 2024
- [41] QINGYU Wang, TAKUJI Nakashima, CHENGUANG Lai, XINRU Du, TAIGA Kanehira, YASUFUMI Konishi, HIROYUKI Okuizumi, HIDEMI Mutsuda: An improved system for efficient shape optimization of vehicle aerodynamics with “noisy” computations. *Structural and Multidisciplinary Optimization*, 65, 125, 2022
- [42] M. D. Mckay, R. J. Beckman, W. J. Conover: A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output From a Computer Code. *Technometrics*, 21(2), 239-245, 1979
- [43] RONALD L. Iman, W.J. Conover: Small sample sensitivity analysis techniques for computer models. with an application to risk assessment. *Communications in statistics-theory and methods*, 9(17), 1749-1842, 1980
- [44] Max D. Morris, Toby J. Mitchell: Exploratory designs for computational experiments. *Journal of Statistical Planning Inference*, 43(3), 381–402, 1995
- [45] STUART J. Bates, JOHANN Sienz, VASSILI V. Toropov: Formulation of the optimal latin hypercube design of experiments using a permutation genetic algorithm. 45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference, Palm Springs, California, 1-7, 2024

- [46] RUICHEN Jin, WEI Chen, AGUS Sudjianto: An efficient algorithm for constructing optimal design of computer experiments. in ASME 2003 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 545-554, 2003
- [47] THANYADOL Chantarawong, JARATSRI Rungrattanaubol, ANAMAI Na-Udom: Enhancement of enhanced stochastic evolutionary algorithm for computer simulated experiment. Information Technology Journal, 6(2), 65–69, 2010
- [48] A. Grosso, A. R. M. J. U. Jamali, M. Locatelli: Finding maximin latin hypercube designs by iterated local search heuristics. European Journal of Operational Research, 197(2), 541–547, 2009
- [49] QINGYU Wang, TAKUJI Nakashima, CHENGUANG Lai, HIDEKI Mutsuda, TAIGA Kanehira, YASUFUMI Konishi, HIROYUKI Okuizumi: Modified algorithms for fast construction of optimal latin-hypercube design. IEEE Access, 8, 191644-191658, 2020
- [50] JONATHAN Ollar, CHARLES Mortished, ROYSTON Jones, JOHANN Sienz, VASSILI Toropov: Gradient based hyper-parameter optimisation for well conditioned kriging metamodels. Structural and Multidisciplinary Optimization, 55, 2029-2044, 2017
- [51] QINGYU Wang, TAKUJI Nakashima, CHENGUANG Lai, BO Hu, XINRU Du, ZHONGZHENG Fu, TAIGA Kanehira, YASUFUMI Konishi, HIROYUKI Okuizumi, HIDEKI Mutsuda: Enhanced expected hypervolume improvement criterion for parallel multi-objective optimization. Journal of Computational Science, 65, 101903, 2022
- [52] RAINER Storn, KENNETH Price: Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11, 341-359, 1997
- [53] F. van den Bergh, A. P. Engelbrecht: A study of particle swarm optimization particle trajectories. Information Sciences, 176(8), 937-971, 2006
- [54] JING Liang, B. Y. Qu, PONNUTHURAI N. Suganthan, ALFREDO G. Hernández-Díaz: Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report, 201212(34), 281-295, 2013
- [55] SWAGATAM Das, PONNUTHURAI N. Suganthan: Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems. Kolkata: Jadavpur University, Nanyang Technological University, 341-359, 2010

- [56] QINGYU Wang, TAKUJI Nakashima, TAIGA Kanehira, HIDEKI Mutsuda: A Python toolbox for surrogate-based optimization. IEEE 12th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 80-84, 2022
- [57] DONALD W. Zimmerman, BRUNO D. Zumbo: Relative Power of the Wilcoxon test, the Friedman test, and repeated Measures ANOVA on ranks. The Journal of Experimental Education, 62(1), 75-86, 1993
- [58] JIAQING Kou, LAURA Botero-Bolivar, R. Ballano, OSCAR Marino, LEANDRO D. Santana, EUSEBIO Valero, ESTEBAN Ferrer: Aeroacoustic airfoil shape optimization enhanced by autoencoders. Expert Systems With Applications, 217, 119513, 2023
- [59] YIN Yu, ZHOUJIE Lyu, ZELU Xu: On the influence of optimization algorithm and initial design on wing aerodynamic shape optimization. Aerospace Science and Technology, 75, 183-199, 2018
- [60] JAMES L. Thomas, M. D. Salas: Far-field boundary conditions for transonic lifting solutions to the Euler equations. AIAA Journal, 24(7), 1074-1080, 1986
- [61] RENAN F. Soares, KEVIN F. Garry, JENNIFER Holt: Comparison of the far-field aerodynamic wake development for three driver model configurations using a cost-effective RANS simulation. SAE Technical Paper 2017-01-1514, 2017
- [62] NASA Langley Turbulence Modeling Resource website, <http://turbmodels.larc.nasa.gov>.
- [63] JIE Zhou, GANQU Cui, SHENG DING Hu, ZHENG YAN Zhang, CHENG Yang, ZHIYUAN Liu, LIFENG Wang, CHANGCHENG Li, MAOSONG Sun: Graph neural networks: A review of methods and applications. AI Open, 1, 57-81, 2020
- [64] MITSUHIRO Sato, YUETSU Kodama, MIWAKO Tsuji, TESUYA Odajima: Co-Design and System for the Supercomputer “Fugaku”. IEEE Micro, 42(2), 26-34, 2022
- [65] NICLAS Jansson, RAHUL Bale, KEIJI Onishi, MAKOTO Tsubokura: CUBE: A scalable framework for large-scale industrial simulations. International Journal of High Performance Computing Applications, 33(4), 678-698, 2018
- [66] DIKSHIT Chauhan, ANUPAM Yadav: Optimizing the parameters of hybrid active power filters through a comprehensive and dynamic multi-swarm gravitational search algorithm. Engineering Applications of Artificial Intelligence, 123, 106469, 2023