A surrogate-assisted differential evolution for expensive optimization with equality constraints

Jinxin Zhang[a], Jing-Yu Ji[a,*], Sam Kwong[b]

[a]*College of Management, Shenzhen University, Shenzhen 518060, China*
[b]*School of Data Science, Lingnan University, Hong Kong*

Abstract

Surrogate-assisted evolutionary algorithms have achieved notable success in expensive optimization. While significant attention has been given to expensive optimization scenarios without constraints or only with inequality constraints, equality constraints also commonly emerge in constrained optimization problems. Thus, there is a pressing need for surrogate-assisted evolutionary algorithms tailored for expensive optimization problems with equality constraints. This study integrates a multilayer perceptron as a surrogate with gradient descent based local search in differential evolution to tackle the challenges caused by equalities in expensive constrained optimization. Our contributions encompass: 1) deploying a multilayer perceptron-based cheap surrogate that simultaneously fits the expensive objective function and equality constraints, 2) an enhanced gradient descent local search to manage challenging equality constraints, and 3) an individual update strategy aiming to strike a balance between objective optimization and constraint satisfaction. The proposed multilayer perceptron-based surrogate, along with the gradient descent-based local search, collaboratively navigates towards the feasible regions. The evolutionary search leveraging the surrogate broadly explores potential feasible regions, while the local search refines promising infeasible solutions into feasible ones. Experimental results highlight the capability and effectiveness of our proposed surrogate-assisted methodology for expensive optimization with equality constraints.

Keywords: Equality constraint, expensive constrained optimization, gradient descent, multilayer perceptron, surrogate-assisted differential evolution.

List of Abbreviations

COP	Constrained optimization problem.
EA	Evolutionary algorithm.
FE	Fitness evaluation.
ECOP	Expensive constrained optimization problem.
SAEA	Surrogate-assisted evolutionary algorithm.
SVM	Support vector machine.
RBFN	radial basis function network.
GP	Gaussian process.
DE	Differential evolution.

*Corresponding author
Email address: im@jijingyu.com (Jing-Yu Ji)

EECOP	Expensive equality constrained optimization problems.
GLSADE	Global and local surrogate-assisted differential evolution.
MLPS	Multilayer perceptron-based surrogate.
AGDLS	Adaptive gradient descent-based local search.
MLP	Multilayer perceptron.
GD	Gradient descent.
ReLU	Rectified Linear Unit.
OFIE-ISC	Objective function information-enhanced infill sampling criterion.
LHS	Latin hypercube sampling.
$MaxFEs$	Maximum number of expensive fitness evaluations.
MCV	Mean overall degree of constraint violations.
MOF	Mean objective function values.
SR	Success ratio.
TG	Average number of times that the gradient has been numerically calculated.
ET	Overall execution time.
GLSADE-Sur	A single iteration for MLPS-based evolution in GLSADE.
GLSADE-WoLS	GLSADE without AGDLS.
GLSADE-3 GLSADE-10	GLSADE with stagnation count being set to 3 and 10, respectively.
GPEEC, ESAO-CH, SA-C2oDE, and SParEA	Four recent SAEA to solve ECOPs.
εDEag	A model-free EA for COPs.

1. Introduction

Constrained optimization problems (COPs) tackled by meta-heuristics have been extensively studied over the years. Numerous strategies have been proposed, showcasing the efficacy of evolutionary algorithms (EAs) in solving COPs. Typically, EAs are black-box methods [1, 2, 3] and are not particularly sensitive to COP formulations. A common assumption in EAs is access to numerous fitness evaluations (FEs). This assumption breaks down when both the objective and the constraints are expensive to evaluate, incurring substantial time and/or material costs. Such a situation gives rise to expensive COPs (ECOPs) [4]. Once the number of FEs has been limited due to high costs, traditional EAs may struggle to find even a feasible solution. Therefore, surrogate-assisted EAs (SAEAs) [5] were developed. These algorithms utilize surrogates for preliminary assessments, and subsequently shortlist promising candidates. During this process, surrogates offer cost-effective approximations, reducing the need for expensive FEs [6, 7]. For example, [7] employs inexpensive surrogate models to approximate the objective and inequality-constraint functions, allowing candidate-solution quality to be estimated without incurring costly FEs. Popular surrogate models include the radial basis function networks (RBFNs) [8, 9], the support vector machine (SVM) [10], and the Gaussian process (GP) [11, 12].

For ECOPs with inequality constraints, surrogates are primarily used for two purposes: 1) global search and 2) local refinement. The former guides the evolutionary search towards feasible regions, whereas the latter focuses on refining promising individuals. Wang *et al.* [13] pioneered the integration of differential evolution (DE) with both global and local surrogates for ECOPs. They segmented the evolutionary process into two phases: global and local surrogate-assisted. For these phases, the generalized regression neural network and RBFN models were respectively employed to assist the global and local searches. In a separate

study [14], global and local RBFN models were established for two distinct search stages. The global RBFN model drives the exploration within the search space during the initial stage, whereas the local RBFN model bolsters exploitation within the identified feasible region in the subsequent stage. Wei *et al.* [15] proposed a two-stage surrogate-assisted DE that leverages GP and SVM models to aid the global and local searches in distinct stages. Contrasting with neural network-based surrogates, the GP model is devised for evolutionary population to extract feasible information from infeasible candidates, whereas the SVM classification model is tailored to intensively mine data from feasible candidates.

In the context of ECOPs with inequality constraints, a variety of numerous infill sampling criteria have been proposed to select candidates for expensive evaluations [16, 17]. Under a given infill sampling criterion, a candidate solution that is superior to the current evaluated solutions is treated as promising, i.e., it is expected to lie near the optimum of the approximated landscape. As a result, the infill sampling criteria play a pivotal role in determining evolutionary directions, encompassing both exploration and exploitation [18, 19, 20]. Song *et al.* [21] introduced an adaptive sampling approach where a notably distinct, potential candidate is prioritized, succeeded by a solution reflecting the utmost uncertainty. In another work, [22] presented a SAEA tailored for distributed and expensive constrained optimization. They put forth a combined infill sampling criterion that encompasses both individual-centric and generation-centric sample choices. The former targets the most optimal individual characterized by a minimal predicted constraint violation degree, while the latter focuses on the optimal individual that possesses unevaluated data, augmented on-demand by dispersed worker agents. Another approach is detailed in [23], where infill solutions are pinpointed via an enhanced infeasibility-driven strategy. This approach employs nondominated sorting, taking into account the number of satisfied constraints and the cumulative degree of constraint violation.

While ECOPs with inequality constraints have garnered significant attention, leading to the development of various successful SAEAs in recent years, insufficient attention has been paid to ECOPs with equality constraints [24, 25, 26]. In [24], the authors made one of the first attempts to develop an equality-constraint handling technique coupled with a surrogate-assisted optimizer, which benefits from the gradual shrinking of an expanded feasible region. This gradual shrinking smoothly guides infeasible solutions toward the feasible hypersurfaces. By mapping the feasible region to the origin of a Euclidean subspace, Su *et al.* [25] transformed all equality constraints into a single one, after which a Gaussian penalty function was introduced to convert the resulting equality-constrained optimization problem into an unconstrained one. In [26], an expected-improvement-based local search is employed to efficiently improve the accuracies of Gaussian process models in promising neighboring areas of the equality-constraint boundary. On the one hand, these aforementioned studies provide new paradigms of infill sampling criteria and surrogate model management to successfully handle equality constraints under a limited budget of expensive FEs. On the other hand, as highlighted in [15], the challenges in approximating equality constraints surpass those associated with inequality constraints. Consequently, research on expensive equality-constraint handling remains relatively scarce in the literature. Yet, the relevance of equality constraints as vital components in COPs, especially in relation to ECOPs, remains undeniable. Consequently, there is an evident need for SAEAs tailored for expensive equality COPs (EECOPs). Addressing this gap, our study introduces a global and local surrogate-assisted differential evolution (GLSADE) framework. Within this framework, a multilayer perceptron-based surrogate (MLPS) and an adaptive gradient descent-based local search (AGDLS) are synchronized efficiently. To the best of our knowledge, this is among the initial attempts in addressing EECOPs. The main contributions of this study are as follows:

- A MLPS is introduced as a multi-output model to concurrently approximate the objective function and individual equality constraints. This configuration ensures that the surrogate focuses on approximating the original objective function and constraints, rather than a composite fitness function of the

objective and all constraints or an aggregate measure of constraint violation. In this way, it effectively leverages the characteristics of the multilayer perceptron (MLP) to obtain highly accurate surrogates for the objective function and each constraint within a single training session.

- An AGDLS has been seamlessly incorporated into the evolutionary process, serving as a tool to tackle intricate equality constraints. Should the evolutionary process hit a stagnation point over a defined number of iterations, the AGDLS springs into action, optimizing constraints and objective functions in tandem.
- A distinctive infill sampling criterion has been crafted, aiming to balance between objective optimization and satisfying constraints in EECOPs.
- A test suite comprising 14 EECOPs has been gathered from existing literature. This suite serves to evaluate the proposed algorithm and investigate the efficacy of MLPS and AGDLS, shedding light on potential pathways for addressing more intricate EECOPs.

The remainder of this paper is organized as follows. Section II provides a brief overview of preliminaries. The proposed GLSADE is thoroughly discussed in Section III. In Section IV, we conduct empirical evaluations of the proposed algorithm, delve into its deeper analysis, and compare it with several state-of-the-art SAEAs for EECOPs. Finally, Section V offers conclusions and touches upon potential future work.

2. Background

In this section, we provide a concise introduction to the ECOPs, MLP for regression, DE, standard gradient descent (GD), and its associated numerical calculations.

2.1. Optimization with Constraints

COPs involve both objective functions and constraints. Without loss of generality, a single-objective COP with a total of $n_h + n_g$ constraints [27] is represented as:

$$\begin{array}{lll} \text{minimize} & f(\mathbf{x}), & \mathbf{x} = (x_1, \ldots, x_n), \\ \text{subject to} & g_i(\mathbf{x}) \leq 0, & i = 1, \ldots, n_g, \\ & h_j(\mathbf{x}) = 0, & j = 1, \ldots, n_h, \end{array} \tag{1}$$

where $x_i \in [L_i, U_i]$, L_i and U_i are the lower and upper bounds of the ith variable x_i, respectively; $f(\mathbf{x})$ is the objective function, and $\mathbf{x}$ represents the decision vector. The terms n, n_g, and n_h denote the numbers of variables, inequality, and equality constraints, respectively.

Given a decision vector $\mathbf{x}$, the overall degree of constraint violation is computed by $C(\mathbf{x})$:

$$C(\mathbf{x}) = \sum_{i=1}^{n_g} \hat{g}_i(\mathbf{x}) + \sum_{j=1}^{n_h} \hat{h}_j(\mathbf{x}), \tag{2}$$

where $\hat{g}(\mathbf{x})$ and $\hat{h}(\mathbf{x})$ are:

$$\begin{array}{ll} \hat{g}_i(\mathbf{x}) = \max\{g_i(\mathbf{x}), 0\}, & i = 1, \ldots, n_g, \\ \hat{h}_j(\mathbf{x}) = \max\{|h_j(\mathbf{x})| - 0.0001, 0\}, & j = 1, \ldots, n_h. \end{array} \tag{3}$$

A tolerance of 0.0001 is used for equality-constraint feasibility [28, 29]. A candidate $\mathbf{x}$ is feasible when all constraints are satisfied, equivalently, the overall violation $C(\mathbf{x}) = 0$; otherwise, it is infeasible.

2.2. Feasibility Rule

In constrained optimization, whether or not evaluations are expensive, solutions are only useful if they are feasible, so effective constraint handling is central to EAs. Among many strategies, Deb's feasibility rule [30] is widely adopted for its simplicity: it compares two candidates using three ordered rules to decide which one is preferred.

1. The feasible solution is preferred over the infeasible one.
2. If both solutions are infeasible, the one with a smaller $C(\mathbf{x})$ value is preferred.
3. If both solutions are feasible, the one with a better $f(\mathbf{x})$ value is preferred.

The first two rules drive the population toward feasibility, while the third promotes exploitation among feasible candidates. In practice, the feasibility rule quickly channels the search into the feasible set—an advantage in constrained expensive optimization, where evaluation budgets are tight. This benefit is even more pronounced under equality constraints, which compress the feasible manifold and demand rapid convergence. Nevertheless, prior studies [31, 32] report a susceptibility to local entrapment, since the first two tests impose a greedy preference structure.

2.3. Deep Learning Neural Network

Recently, multi-output surrogate models have been utilized to discern the structure of the objective function and each constraint, thereby enhancing the representation of the feasible region's boundaries. In this study, we adopt the MLP neural network [33], which is widely used in various meta-modeling applications [34, 35], as a multi-output surrogate to simulate ECOPs. An MLP consists of neurons, with the output (y) of each neuron being calculated as follows:

$$y = \phi\left(\sum_{i=1}^{n} \omega_i x_i + b\right) \tag{4}$$

where, ω_i represents the weight for the ith variable x_i of the input decision vector $\mathbf{x}$, b is the bias term, and ϕ is the activation function. We use the Rectified Linear Unit ($ReLU$) [36] as the activation function for each neuron, defined as:

$$ReLU(x) = \max\{0, x\} \tag{5}$$

The $ReLU$ function is differentiable at all points except at zero, which is mathematically advantageous. In addition to its robust performance, $ReLU$ has become the default activation function for most deep learning tasks.

MLPs are widely recognized as universal regression models [37], making them highly effective for modeling black-box functions. The development of the back-propagation learning algorithm, which determines the weights in an MLP, has significantly contributed to their popularity among researchers and practitioners. In the context of constrained expensive optimization, MLP neural networks are advantageous as they can fit the objective function and each constraint simultaneously as multiple outputs. Since the objective function and constraints share the same decision vector, there exists an intrinsic relationship between objective optimization and constraint satisfaction. Training the network on data that includes both the objective function and constraints can uncover this interconnectedness, helping to prevent surrogate overfitting and reduce training times.

2.4. Differential evolution

Traditional DE algorithms [38] were primarily designed for unconstrained single-objective optimization problems. As the search engine, DE and its variants employ mutation, crossover and selection operators to generate the trial vectors of offspring.

2.4.1. Mutation

For each target decision vector $\mathbf{x}_i, i = 1, \ldots, NP$, a mutation operator generates its mutant vector $\mathbf{v}_i$. This study employs the current-to-$p$best/1 mutation strategy:

$$\mathbf{v}_{i,G} = \mathbf{x}_{i,G} + F_i \cdot (\mathbf{x}_{pbest,G} - \mathbf{x}_{i,G}) + F_i \cdot (\mathbf{x}_{r_1,G} - \mathbf{x}_{r_2,G}). \tag{6}$$

Here, $\mathbf{x}_{pbest,G}$ denotes a random selection from the top $p\%$ individuals in the current population at the Gth generation. F_i is the scale factor for $\mathbf{x}_i$, and r_1, r_2, and i are mutually different integers.

2.4.2. Crossover

Following mutation, the crossover operator is applied to $\mathbf{v}_i$ and $\mathbf{x}_i$, forming a trial/offspring vector $\mathbf{u}_{i,G} = (u_{i,1,G}, u_{i,2,G} \ldots, u_{i,n,G})$ at the Gth generation. The binomial crossover scheme used for the jth variable of $\mathbf{u}_{i,G}$ is:

$$u_{i,j,G} = \begin{cases} v_{i,j,G}, & \text{if } (rand_{i,j,G}[0,1] \leq C_r) \text{ or } (j = j_{rand}), \\ x_{i,j,G}, & \text{otherwise.} \end{cases} \tag{7}$$

where C_r represents the crossover rate, j_{rand} is a random value chosen from $\{1, \ldots, n\}$, and $rand_{i,j,G}[0,1]$ is a uniform random number in $[0,1]$.

2.4.3. Selection

Once offspring have been reproduced, the selection operator is employed to determine whether the offspring $\mathbf{u}_i$ or its target decision vector $\mathbf{x}_i$ can survive into the next generation. The selection operator of DE works as follows:

$$\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G} & \text{if } f(\mathbf{u}_{i,G}) \leq f(\mathbf{x}_{i,G}), \\ \mathbf{x}_{i,G} & \text{otherwise.} \end{cases} \quad i = 1, \ldots, NP \tag{8}$$

2.5. Standard gradient descent

GD is a fundamental method for numerical optimization: starting from an initial guess, it repeatedly updates the iterate in the negative gradient direction to reduce the objective. As illustrated in Fig. 1, for a simple convex objective the iterates move monotonically downhill along the steepest descent direction until reaching a (local) minimum.

Lately, GD as a repair method [14] has been developed to assist constraint-handling techniques so as to rapidly pinpoint feasible solutions for model-free COPs. In this way, the gradient for classic COPs of (1) at

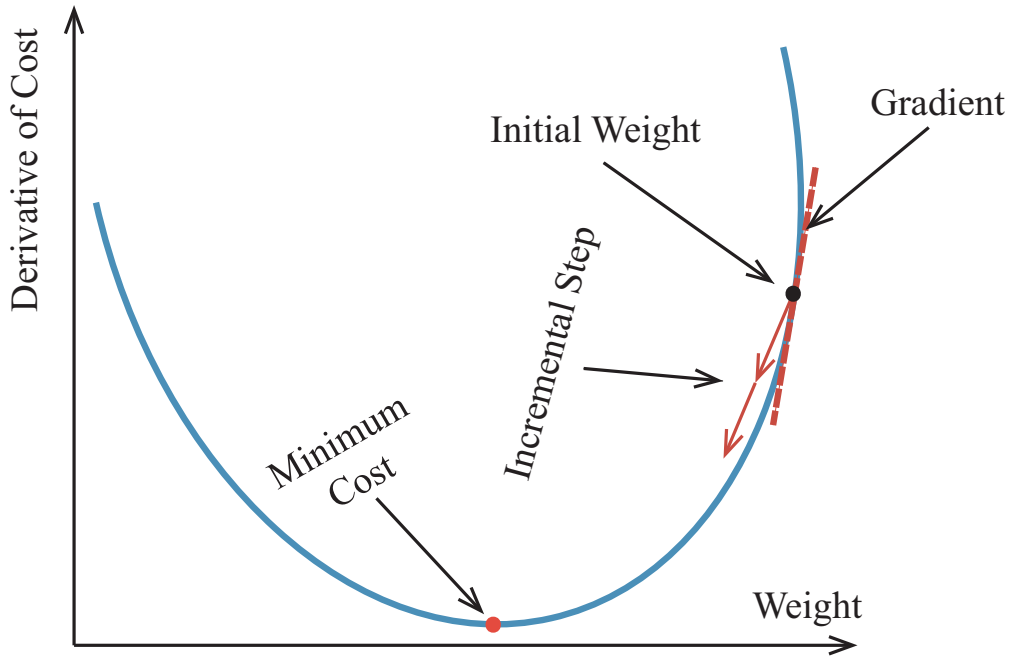


Figure 1. An illustration of the standard GD algorithm. The black point represents the initial weight, while the red point indicates the minimum cost. Incremental step against the gradient guides the initial weight to this minimum cost.

a given n-dimensional vector $\mathbf{x} = (x_1, \ldots, x_n)$ is represented as:

$$\nabla_{\mathbf{x}} \hat{G} = \begin{bmatrix} \frac{\partial \hat{g}_1}{\partial x_1}(\mathbf{x}), & \frac{\partial \hat{g}_1}{\partial x_2}(\mathbf{x}), & \ldots, & \frac{\partial \hat{g}_1}{\partial x_n}(\mathbf{x}) \\ \ldots, & \ldots, & \ldots, & \ldots \\ \frac{\partial \hat{g}_{ng}}{\partial x_1}(\mathbf{x}), & \frac{\partial \hat{g}_{ng}}{\partial x_2}(\mathbf{x}), & \ldots, & \frac{\partial \hat{g}_{ng}}{\partial x_n}(\mathbf{x}) \\ \frac{\partial \hat{h}_1}{\partial x_1}(\mathbf{x}), & \frac{\partial \hat{h}_1}{\partial x_2}(\mathbf{x}), & \ldots, & \frac{\partial \hat{h}_1}{\partial x_n}(\mathbf{x}) \\ \ldots, & \ldots, & \ldots, & \ldots \\ \frac{\partial \hat{h}_{n_h}}{\partial x_1}(\mathbf{x}), & \frac{\partial \hat{h}_{n_h}}{\partial x_2}(\mathbf{x}), & \ldots, & \frac{\partial \hat{h}_{n_h}}{\partial x_n}(\mathbf{x}) \end{bmatrix}, \tag{9}$$

where ∂ and ∇ signify a partial derivative and the vector differential operator, respectively. $\nabla_{\mathbf{x}} \hat{G}$ suggests the fastest increase direction of constraints. Conversely, its opposite, $-\nabla_{\mathbf{x}} \hat{G}$, indicates the most significant improvement for constraints. For objective function and constraints in black-box format, the gradient is approximated linearly. For instance, the gradient of $\hat{g}_i(\mathbf{x})$ for the jth variable of $\mathbf{x}$ can be numerically evaluated as:

$$\frac{\partial \hat{g}_i}{\partial x_j}(\mathbf{x}) = \frac{\hat{g}_i(x_1, \ldots, x_j + \Delta x, \ldots, x_n) - \hat{g}_i(x_1, \ldots, x_j, \ldots, x_n)}{\Delta x}, \tag{10}$$

where Δx approaches zero.

In terms of constraints, gradient-based repair methods [39, 40] aim to find the nearest feasible solution

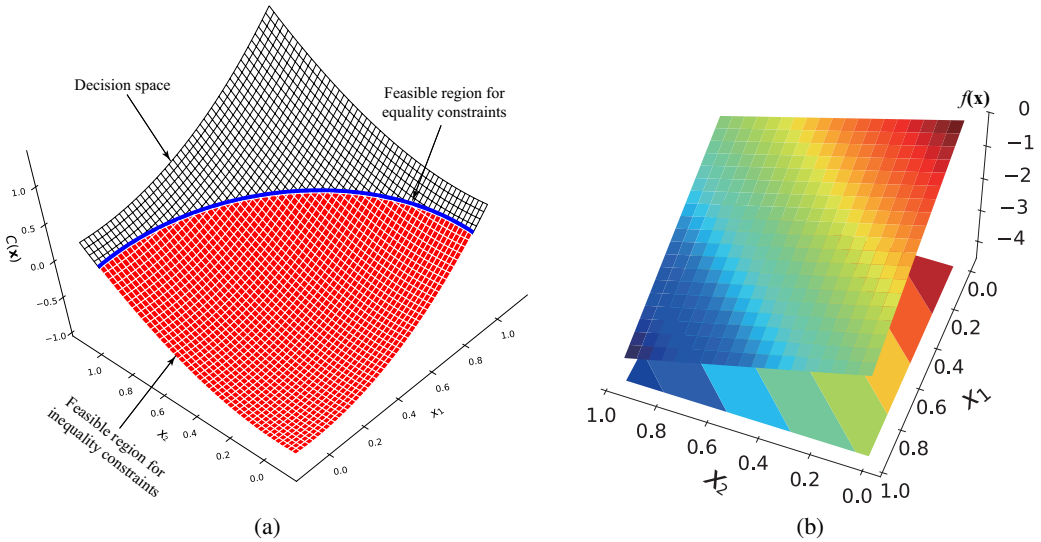


Figure 2. An illustration briefing the complexity of EECOP. (a) landscape of the search space. (b) landscape of the objective space. The feasible region for inequality constraints is indicated by the red area in (a), while the feasible region for equality constraints is delineated by the blue line in (a).

from an infeasible individual. The refinement after the local search process can be expressed as:

$$\begin{cases} \mathbf{x}' = \mathbf{x} + \Delta\boldsymbol{\eta} \\ \nabla_{\mathbf{x}}\hat{G}\Delta\boldsymbol{\eta} = -\Delta_{\mathbf{x}}\hat{G} = -(\hat{g}_1(\mathbf{x}),\ldots,\hat{g}_{n_g}(\mathbf{x}),\hat{h}_1(\mathbf{x}),\ldots,\hat{h}_{n_h}(\mathbf{x}))^T \end{cases} \tag{11}$$

Here, $\mathbf{x}$ represents an infeasible solution, $\mathbf{x}'$ a proximate feasible solution, $\Delta_{\mathbf{x}}\hat{G}$ is the vector of constraint violations, and $\Delta\boldsymbol{\eta}$ is the increment step size, computable by:

$$\Delta\boldsymbol{\eta} = -\nabla_{\mathbf{x}}\hat{G}^{-1}\Delta_{\mathbf{x}}\hat{G} \tag{12}$$

where $\nabla_{\mathbf{x}}\hat{G}^{-1}$ is the pseudo-inverse of $\nabla_{\mathbf{x}}\hat{G}$.

3. The proposed algorithm

As the solution of EECOP is an emerging research topic, there have been limited efforts devoted to addressing equality constraints in such problems. To clarify our contributions, we initially discuss the motivation behind this study and subsequently detail each component of the GLSADE approach.

3.1. Motivation

We begin by elucidating the challenges inherent in EECOPs. For this purpose, we reference the benchmark function G03 from CEC 2006 test suite [41], defined as follows:

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) = -(\sqrt{n})^n \prod_{i=1}^{n} x_i, \\ \text{subject to} \quad & h_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 - 1 = 0, \end{aligned} \tag{13}$$

where $n = 10$ and $0 \leq x_i \leq 1, i = 1, \ldots, 10$. As a comparative exercise, we modify the above constraint to an inequality constraint:

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) = -(\sqrt{n})^n \prod_{i=1}^{n} x_i, \\ \text{subject to} \quad & g_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 - 1 \leq 0. \end{aligned} \tag{14}$$

For better visualization, the feasible regions for (13) and (14) with two variables are depicted in Fig. 2a. The transition from an inequality to an equality constraint significantly reduces the feasible region from the larger red area to the narrow blue line. Notably, compared to the entire search space, the blue line's proportion is minuscule, making it difficult to approximate with limited sampling under constrained FEs. Moreover, the search process in EAs typically evolves gradually. Hence, the red area can be effectively explored by evolutionary search with surrogate assistance. However, while exploring the blue line, surrogate models are often built using data from infeasible individuals, which may not accurately predict potential search directions.

This discussion leads to two primary motivations for developing the GLSADE framework. Firstly, evaluations of equality constraints in ECOPs often rely on expensive physical experiments or time-consuming simulations. Despite the prevalence of these issues, few studies have addressed EECOPs. Our research aims to develop a SAEA to meet this need. Secondly, the feasible region in EECOPs is narrower compared to ECOPs with inequality constraints, presenting additional challenges for current constraint-handling techniques and surrogate models in efficiently finding feasible solutions within a limited computational budget. To effectively utilize available information, we employ a MLP as a multi-output surrogate model. The MLP, with its multiple hidden layers, plays a pivotal role in complex regression tasks. Its robust architecture is not just limited to approximating the objective function but extends to simultaneously approximating equality constraints as well. This dual capability underlines the versatility and importance of MLP in computational modeling. Additionally, to enhance the evolutionary process and refine search directions, we introduce two local search strategies based on MLP and GD.

3.2. MLPS-based evolution

In the context of EECOPs, feasible solutions are typically located in narrow areas. To efficiently explore potential areas through global search and to exploit high-quality solutions via local search, we have designed a MLPS-based evolution that facilitates both global and local searching capabilities. Additionally, we have developed a specific infill sampling criterion for EECOPs within the MLPS framework to fully leverage the enhanced capabilities of better-trained models. The high-level pseudo-code for the MLPS-based evolution is presented in **Algorithm 1**.

Once MLPS has been trained as a regression model using the precisely evaluated individuals in set A, it can simultaneously approximate the objective function and each of the equality constraints. The well-trained MLPS, serving as an alternative to expensive FEs, is used to assess individuals and drive the evolution of population P. DE operators, as the search engine used in the field of model-free constrained EAs, are employed to generate offspring within the bounds $[L_i, U_i], i = 1, \ldots, n$. For selection, the feasibility rule [30] is applied to compare individuals. As DE and the feasibility rule are established techniques, their detailed application is not elaborated in **Algorithm 1**. Like the model-free algorithm, above steps are iteratively repeated till the termination is satisfied.

The MLPS-based evolution utilizes P as the initial population for DE optimization of the surrogate. This process is akin to a model-free evolutionary progress, and the output is considered to be optimal solutions for the corresponding optimization problem. However, MLPS may not accurately reflect the true values of each individual, and the issue of premature convergence, common in normal evolutionary processes, also

Algorithm 1: MLPS-based Evolution

Input:
R: the received set of exactly evaluated solutions;
P: the set of NP individuals;
$[L_i, U_i], i = 1, \ldots, n$: the lower and upper bounds of n variables.
Initialization:
Train the MLP model as a surrogate using set R.
Surrogate-based Evolution:
while *termination criteria are not met* **do**
    Generate offspring population Q using DE operators;
    Evaluate Q with the well-trained surrogate;
    Calculate the $C(\mathbf{x})$ for each individual in Q;
    Update P with Q using the feasibility rule.
end
Output:
The individuals in P.

occurs in MLPS-based evolution. Therefore, special selection of potential candidates is necessary via the infill sampling criterion.

3.3. Improved infill sampling criterion

Given the limited budget for expensive FEs, only a few individuals can be precisely evaluated following the MLPS-based evolution. Furthermore, as highlighted in [32], information gleaned from the optimization of the objective function is instrumental in mitigating search bias, particularly bias that arises due to a preference for constraint satisfaction. In light of these considerations, we have formulated an objective function information-enhanced infill sampling criterion, namely (OFIE-ISC). This criterion aims to identify the most promising candidate, one that maximizes improvement in both objective function optimization and constraint satisfaction. The implementation of OFIE-ISC is detailed in **Algorithm 2**.

Following surrogate-based evaluations, the proposed OFIE-ISC selects the most potential individual by considering both the optimization of the objective function and constraint satisfaction. Initially, individuals that could potentially improve all equality constraints are given priority. If such individuals exist, they are used to alleviate search bias caused by significantly violated constraints in the early stage, ensuring consistent handling of all inequality constraints. If no such individual exists, those who can potentially reduce the overall degree of constraint violation, $C(\mathbf{x})$, are then considered, as feasible solutions are always preferable to infeasible ones. Subsequently, the objective function, $f(\mathbf{x})$, is considered in the proposed OFIE-ISC to further mitigate search bias. Hence, the individual with the lowest $f(\mathbf{x})$ value is selected for exact FE execution. In scenarios where no potential improvement is identified based on these criteria, the individual from the input population P' with the lowest $C(\mathbf{x})$ value is chosen as a new datum to update the training set TS.

3.4. AGDLS

Local search, widely integrated into various EAs for solving challenging COPs, has seen recent advancements with the incorporation of GD to expedite convergence in expensive optimization scenarios [42, 43, 44]. However, the numerical calculation of gradients, even for a single point, necessitates a substantial number of expensive FEs, thereby limiting the applicability of GD-based local search in ECOPs.

Algorithm 2: OFIE-ISC

Input:
P: the set of NP exactly evaluated individuals;
P': the set of NP approximately evaluated individuals;
TS: the set of all exactly evaluated solutions.
Initialization:
Set $Q = \{\}$;
Set $Q' = \{\}$.
Comparison:
for *i=1* **to** NP **do**
  if $\forall j \in 1,\ldots,n_h, \hat{h}_j(\boldsymbol{x}_{i,P'}) < \hat{h}_j(\boldsymbol{x}_{i,P})$ **then**
    Put $x_{i,P}$ into Q;
    Put $x_{i,P'}$ into Q';
  end
end
if *Q and Q' are empty* **then**
  for *i=1* **to** NP **do**
    if $C(\boldsymbol{x}_{i,P'}) < C(\boldsymbol{x}_{i,P})$ **then**
      Put $x_{i,P}$ into Q;
      Put $x_{i,P'}$ into Q';
    end
  end
end
if *Q and Q' are empty* **then**
  Identify the individual $\mathbf{x}_b$ with the lowest $C(\mathbf{x})$ value in P';
else
  Identify the individual $\mathbf{x}_b$ with the lowest $f(\mathbf{x})$ value in Q';
end
Identify the individual $\mathbf{x}_w$ with the highest $C(\mathbf{x})$ value in P;
Update:
Exactly evaluate $\mathbf{x}_b$ using expensive FE;
Archive $\mathbf{x}_b$ in TS;
Replace $\mathbf{x}_w$ with $\mathbf{x}_b$ using the feasibility rule.
Output:
The updated TS and P.

In this study, we first propose an improved version of GD-based local search, namely AGDLS, and second, integrate it with MLPS-based local search to refine promising solutions within a constrained computational budget. As surrogate-based local search is a common concept in SAEAs, we focus primarily on describing AGDLS in this section. Further implementation details are provided in the subsequent section. **Algorithm 3** presents the high-level pseudo-code of AGDLS.

Given a decision vector $\mathbf{x}$ and its equality constraint violations $\hat{h}(\mathbf{x}), i = 1,\ldots,n_h$, the corresponding negative of the gradient is

$$\nabla_{\mathbf{x}} G = \begin{bmatrix} \frac{\partial \hat{h}_1}{\partial x_1}(\mathbf{x}), & \frac{\partial \hat{h}_1}{\partial x_2}(\mathbf{x}), & \ldots, & \frac{\partial \hat{h}_1}{\partial x_n}(\mathbf{x}) \\ \ldots, & \ldots, & \ldots, & \ldots \\ \frac{\partial \hat{h}_{n_h}}{\partial x_1}(\mathbf{x}), & \frac{\partial \hat{h}_{n_h}}{\partial x_2}(\mathbf{x}), & \ldots, & \frac{\partial \hat{h}_{n_h}}{\partial x_n}(\mathbf{x}) \end{bmatrix} \tag{15}$$

Suppose the increment step size and improvements of each constraint on infeasible individual $\mathbf{v}$ to its nearest better solution $\mathbf{u}$ are $\Delta\boldsymbol{\eta}$ and $\Delta\hat{h}_j$ for $j = 1,\ldots,n_h$. Then, the relationship among them can be approximated as follows:

$$\left[\Delta\hat{h}_1,\ldots,\Delta\hat{h}_{n_h}\right]^T = -\Delta\boldsymbol{\eta}\nabla_{\mathbf{v}} G, \tag{16}$$

leading to

$$\Delta\boldsymbol{\eta} = -\left[\Delta\hat{h}_1,\ldots,\Delta\hat{h}_{n_h}\right]^T \nabla_{\mathbf{v}} G^{-1}. \tag{17}$$

In accordance with (17), $\nabla_{\mathbf{v}} G^{-1}$ can be numerically calculated. Thus, once the potential improvements of each $\Delta\hat{h}$ are estimated, $\Delta\boldsymbol{\eta}$ is correspondingly obtained by the right-hand term of (17). Conversely, $\mathbf{u}$, as the refinement of $\mathbf{v}$, can be achieved as follows:

$$\mathbf{u} = \mathbf{v} + \Delta\boldsymbol{\eta} \tag{18}$$

The estimation of $\Delta\hat{h}_j = \hat{h}_j(\mathbf{v}) - \hat{h}_j(\mathbf{u}),\ j = 1,\ldots,n_h$, for (17) is suggested from previous studies, where $\mathbf{u}$ is directly considered the nearest feasible solution to $\mathbf{v}$. Therefore, it follows that $\hat{h}_j(\mathbf{u}) = 0$ and, in reality, $\Delta\hat{h}_j = \hat{h}_j(\mathbf{v})$ for $j = 1,\ldots,n_h$.

According to (15), computing the gradient information requires n times the number of expensive FEs. After a repair, the gradient information may not vanish immediately. Based on these considerations, an adaptive scheme has been designed to reuse gradient information until maximal improvement is achieved. As outlined in the while loop of **Algorithm 3**, as long as the refined $\mathbf{v}$ shows improvement, the initially calculated gradient information based on the originally input infeasible individual of $\mathbf{v}$ will continue to be reused. More details on applying AGDLS under specific application conditions will be given in the complete GLSADE framework in the next subsection.

3.5. Complete GLSADE framework

In this section, we describe the complete GLSADE framework in detail.

The flowchart and pseudo-code of GLSADE are presented in Fig.3 and **Algorithm 4**, respectively. Essentially, GLSADE maintains the following information:

1. the population P comprising NP individuals, $\{\mathbf{x}_i | i = 1,\ldots,NP\}$;
2. the objective function value, each constraint violation, and the overall degree of constraint violations for each individual $f(\mathbf{x}_i),\ C(\mathbf{x}_i),\ \hat{h}_j(\mathbf{x}_i),\ i = 1,\ldots,NP,\ j = 1,\ldots,n_h$;
3. the training set TS, archiving all exactly evaluated individuals.

Algorithm 3: AGDLS

Input:
$\mathbf{v}$: the given infeasible individual to exploit;
n: the number of decision variables;
TS: the set of all exactly evaluated solutions;
uFE: the number of used exact FEs.
Initialization:
Calculate $\nabla_{\mathbf{v}} G$ by (10) and (15);
$uFE = uFE + n$;
Calculate $\nabla_{\mathbf{v}} G^{-1}$;
Self-adaptive GD:
while *there is an improvement* **do**
  Calculate $\Delta\boldsymbol{\eta}$ by (17);
  Obtain the refinement $\mathbf{u}$ by (18);
  Evaluate $\mathbf{u}$ using expensive FE;
  $uFE = uFE + 1$;
  Archive $\mathbf{u}$ in TS;
  if $C(\boldsymbol{u}) < C(\boldsymbol{v})$ **then**
    Set $\mathbf{v} = \mathbf{u}$;
  else
    Exit the while loop.
  end
end
Output:
$\mathbf{v}$, uFE, and TS.

During the initialization phase, NP individuals are randomly generated using Latin hypercube sampling (LHS) within the problem-dependent search space. These NP initial individuals are then exactly evaluated using expensive FEs. The decision vector $\mathbf{x} = (x_1, \ldots, x_n)$, along with the objective function value and each constraint, are archived into the training set TS as features and values.

After initialization, the MLPS-based evolution as the global search, i.e., the first item in **Algorithm 1**, is applied at each generation. In this context, all exactly evaluated data in TS are used to approximate the fitness landscape across the entire search space. Therefore, the original problem-dependent lower and upper bounds $[L_i,\ U_i], i = 1, \ldots, n$ are used to regulate the search space. Subsequently, NP evolved individuals from the MLPS-based global search are output, and then the proposed OFIE-ISC in **Algorithm 2** is employed to identify the most potential candidate to update the current population P. Once the evolutionary search has stagnated for *stg* generations, the surrogate-based local search is employed. As shown in **Algorithm 4**, there are two types of local search. The first is based on the MLPS evolution procedure, annotated as "MLPS-based Local Search" in **Algorithm 1**, and the second is AGDLS, described in **Algorithm 3**. For the local search, the infeasible solution $\mathbf{x}_b$ with the smallest $C(\mathbf{x})$, together with its $NP-1$ nearest neighbors in terms of Euclidean distance from $\mathbf{x}_b$, is selected from TS, and then MLPS-based local search is conducted on within a small area $[L_{b,i},\ U_{b,i}], i = 1, \ldots, n$ defined by the selected NP individuals

$$\begin{cases} L_{b,i} = \min\{\mathbf{x}_{1,i}, \mathbf{x}_{2,i}, \ldots, \mathbf{x}_{NP,i}\} \\ U_{b,i} = \max\{\mathbf{x}_{1,i}, \mathbf{x}_{2,i}, \ldots, \mathbf{x}_{NP,i}\} \end{cases}, \quad i = 1, \ldots, n. \tag{19}$$

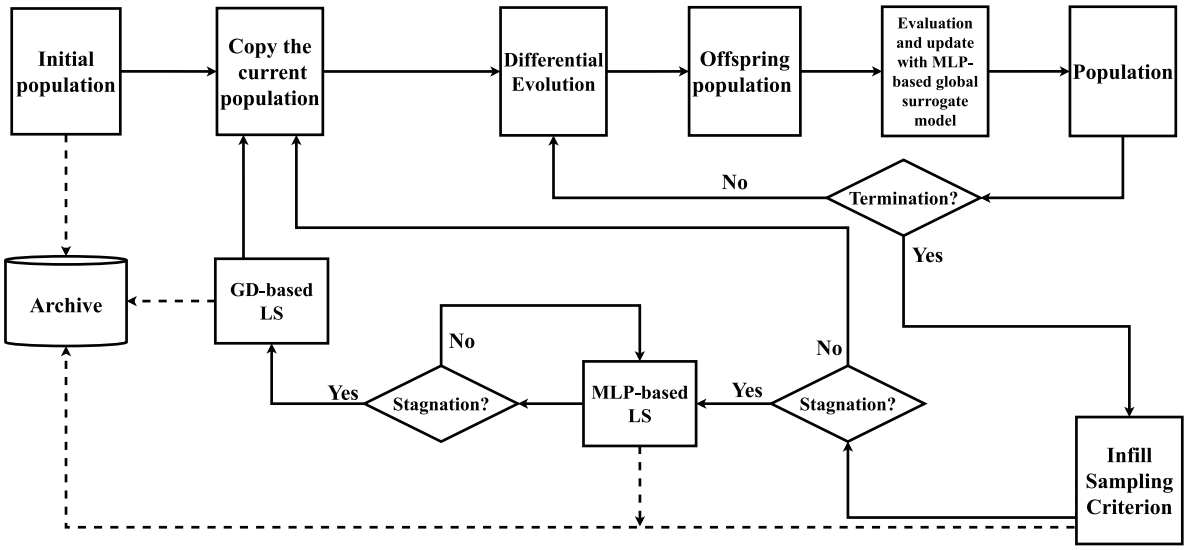


Figure 3. The GLSADE flowchart. Solid arrows represent the algorithm flow, while dotted arrows indicate the archiving of exactly evaluated solutions.

Similarly, if no improvement is obtained through the surrogate-based local search, AGDLS is then activated. The evolutionary process continues until the maximum number of expensive FEs ($MaxFEs$) is reached. Throughout the evolutionary progress, for any vector subjected to expensive evaluation, its values $\mathbf{x}, f(\mathbf{x}), \hat{h}_j(\mathbf{x}), j = 1, \ldots, n_h$ are archived in the training set TS.

The evolutionary process described above outlines the workflow of the proposed GLSADE for addressing EECOPs. The specific modifications and developments tailored for EECOPs are as follows:

- The MLPS-based global search is the primary search engine for exploring the search space. As discussed in Section 3.1, the feasible region for EECOPs is notably slim. Hence, we posit that an improvement achieved by the MLPS-based global search indicates that the evolutionary search is not stagnant and capable of discovering better solutions. Therefore, the MLPS-based global search continues to be employed until no further improvement is found based on the current training set TS.

- With regard to local search, MLPS-based local search is initially utilized until no improvement is found, followed by a single employment of AGDLS. This hybrid approach is grounded in two main considerations: firstly, surrogate-based local search is a popular concept in expensive optimization and has achieved considerable success; secondly, AGDLS requires numerically calculating the gradient, which involves n times the expense of FEs for a given decision vector. Thus, AGDLS is employed only when MLPS-based local search is ineffective.

- The MLPS-based evolution is designed for both global and local search. The input data for R, P, and $[L_i,\ U_i], i = 1, \ldots, n$ depend on the specific application scenario. To construct the global surrogate, all exactly evaluated data in TS are utilized, thereby leveraging the overall information to explore the feasible region and avoid premature convergence. Conversely, for building the local surrogate, the NP nearest neighbors of the best infeasible solution are used, focusing on local information around the target individual for refinement.

Algorithm 4: GLSADE Framework

Input:
NP: the number of initial individuals;
stg: the maximum count for stagnation;
n: the number of variables;
$[L_i, U_i], i = 1, \ldots, n$: the lower and upper bounds of n variables
$MaxFEs$: the maximum number of expensive FEs.
Initialization:
Initiate NP individuals (Set P) in $[L_i, U_i], i = 1, \ldots, n$ using LHS;
Evaluate the NP individuals using expensive FEs;
Set $uFE = NP$;
Archive the evaluated NP individuals (Set TS).
Evolutionary Progress:
while $uFE < MaxFEs$ **do**
——————**MLPS-based Global Search**——————
Set $A = TS$;
Copy the population P to P_c;
Set $flg = 0$;
Execute **Algorithm 1** with A, $[L_i, U_i]$, and P_c;
Execute **Algorithm 2** with P, P_c, and TS;
Set $uFE = uFE + 1$;
if *there is no improvement* **then**
Set $flg = flg + 1$;
else
Set $flg = 0$;
end
while flg *is equal to* stg **do**
——————**MLPS-based Local Search**——————
Set $B = \{\}$;
Identify the best infeasible solution $\mathbf{x}_b$ from P;
Identify the nearest $NP - 1$ neighbors of $\mathbf{x}_b$ from TS;
Put $\mathbf{x}_b$ and its $NP - 1$ neighbors into B;
Make a copy of B as P_l;
Identify the lower and upper bounds $[L_{b,i}, U_{b,i}], i = 1, \ldots, n$ of B;
Execute **Algorithm 1** with B, $[L_{b,i}, U_{b,i}], i = 1, \ldots, n$, and P_l;
Set $uFE = uFE + 1$;
if $\boldsymbol{x}_b$ *has not been updated* **then**
——————**GD-based Local Search**——————
Execute **Algorithm 3** with $\mathbf{x}_b$, n, TS, and uFE;
Set $flg = 0$.
end
end
end
Output:
The best individual $\mathbf{x}_b$ in P.

3.6. Computational time complexity

The computational complexity of GLSADE in one generation is discussed next. Excluding the expensive FEs, the major components contributing to the computational complexity are:

1. the time complexity of constructing a three-hidden-layer MLP for global search is $O(N_{itr} \cdot T_s \cdot Ne^2)$, where N_{itr} is the number of training iterations, T_s is the size of the receiving set R, and Ne is the number of neurons in the three layers;
2. the time complexity of surrogate-based evolution in MLPS is $O(sg \cdot NP \cdot n)$, where sg is the number of evolution generations;
3. the worst-case time complexity of AGDLS is $O(n^3)$.

Therefore, the computational time complexity in each generation of GLSADE is approximate to $O(N_{itr} \cdot T_s \cdot Ne^2) + O(sg \cdot NP \cdot n) + \dfrac{O(N_{itr} \cdot T_s \cdot Ne^2) + O(sg \cdot NP \cdot n) + O(n^3)}{stg}$.

4. Empirical studies

To evaluate the performance of the proposed GLSADE, empirical experiments have been conducted. Firstly, the benchmark test functions and parameter settings for GLSADE are detailed. Secondly, the effect of the parameter *stg*, signifying the maximum count for stagnation, is examined. Thirdly, the effectiveness of MLPS-based evolution and the proposed hybrid local search are empirically verified. Finally, to demonstrate GLSADE's advantages in handling EECOPs, it is compared with five other algorithms using the chosen benchmark test functions.

4.1. Test functions and experiment settings

Table 2. Properties of 14 EECOPs

Func.	n	Objective Type	Feasibility Region	NE
G03	10	Polynomial	0.0000%	1
G11	2	Quadratic	0.0000%	1
G13	5	Nonlinear	0.0000%	3
G14	10	Nonlinear	0.0000%	3
G15	3	Quadratic	0.0000%	2
G17	6	Nonlinear	0.0000%	4
C05	10	Separable	0.0000%	2
C06	10	Separable	0.0000%	2
C09	10	Non Separable	0.0000%	1
C10	10	Non Separable	0.0000%	1
C05′	30	Separable	0.0000%	2
C06′	30	Separable	0.0000%	2
C09′	30	Non Separable	0.0000%	1
C10′	30	Non Separable	0.0000%	1

Given the emergent nature of EECOPs within the domain of ECOPs, there is currently no well-established test suite or SAEAs specifically for EECOPs. Therefore, this study utilizes all six equality-constrained test functions from the IEEE CEC 2006 test suite [41] and eight test functions from the IEEE CEC 2010 test suite [45]. The objective and constraint functions of these problems are treated as expensive. Detailed information is available in Table 2. Here, the feasibility region is the proportion of feasible region(s) over the whole search space. The value 0.0000% means the proportion is smaller than 0.0001%. NE indicates the number of equalities that the corresponding EECOP includes. It is important to note that, although there are more challenging and complex test instances available, the selected functions are sufficient to demonstrate the significant differences and achievements pertinent to this emerging topic.

The stagnation counter *stg*(maximum tolerated non-improving iterations) is introduced for GLSADE and fixed to 6 in our experiments. The initial population size is $NP = 100$. For offspring reproduction in Algorithm 1, we adopt the DE operators and parameter settings from [38], where the scale factor and crossover rate are generated for each individual, and the value of p as well as the archive size are kept unchanged. Surrogate modeling employs a three-hidden-layer MLP implemented in PyTorch v1.10.0 to jointly approximate the objective and constraints. Following [42], each hidden layer uses $\min\{3000, 100 \cdot n\}$ neurons to balance memorization and generalization [46]. Network training minimizes the surrogate loss with the AdamW optimizer using its default hyperparameters, which provided reliable performance across our tests.

For each EECOP instance, we perform 20 independent trials under a fixed budget of $MaxFEs = 1000$ From these runs, we collect the best solution per trial. If a trial¡¯s best solution is infeasible, we report the mean overall degree of constraint violations (MCV) over the 20 best-of-run solutions; if it is feasible, we report the mean objective function values (MOF). When no feasible solution is obtained in any trial,

Table 3. MOF, MCV, and SR Obtained by GLSADE and GLSADE-Sur.

Func.	GLSADE-Sur					GLSADE				
Metrics	*MCV*	*MOF*	*SR*	*TG*	*ET*	*MCV*	*MOF*	*SR*	*TG*	*ET*
G03	0.00E+00	-3.64E-01	1.00	1.5	18 min	**0.00E+00**	**-6.42E-01**	**1.00**	**0.0**	25 min
G11	2.59E-03	8.26E-01	0.45	0.0	UA	**0.00E+00**	**7.50E-01**	**1.00**	**0.5**	48 min
G13	0.00E+00	8.11E-01	1.00	3.5	62 min	**0.00E+00**	**8.09E-01**	**1.00**	**2.0**	72 min
G14	UA	UA	0.00	0.0	UA	**1.17E-04**	**-4.37E+01**	**0.90**	**3.0**	60 min
G15	UA	UA	0.00	0.0	UA	**0.00E+00**	**9.63E+02**	**1.00**	**5.0**	52 min
G17	UA	UA	0.00	0.0	UA	**0.00E+00**	**8.93E+3**	**1.00**	**4.4**	44 min
C05	UA	UA	0.00	0.0	UA	**0.00E+00**	**2.26E+02**	**1.00**	**1.0**	38 min
C06	4.81E-02	-1.99E+02	0.35	1.0	UA	**0.00E+00**	**2.63E+02**	**1.00**	**2.0**	33 min
C09	0.00E+00	2.62E+09	1.00	1.0	30 min	**0.00E+00**	**1.52E+09**	**1.00**	**2.0**	37 min
C10	0.00E+00	6.34E+09	1.00	1.5	34 min	**0.00E+00**	**4.00E+09**	**1.00**	**3.0**	29 min
C05′	**0.00E+00**	**2.49E+02**	**1.00**	**6.0**	37 min	0.00E+00	2.94E+02	1.00	1.8	51 min
C06′	UA	UA	0.00	0.0	UA	**0.00E+00**	**3.70E+02**	**1.00**	**1.0**	77 min
C09′	0.00E+00	8.38E+12	1.00	4.5	68 min	**0.00E+00**	**7.70E+12**	**1.00**	**3.4**	84 min
C10′	**0.00E+00**	**5.30E+12**	**1.00**	**2.5**	53 min	0.00E+00	7.41E+12	1.00	2.0	69 min

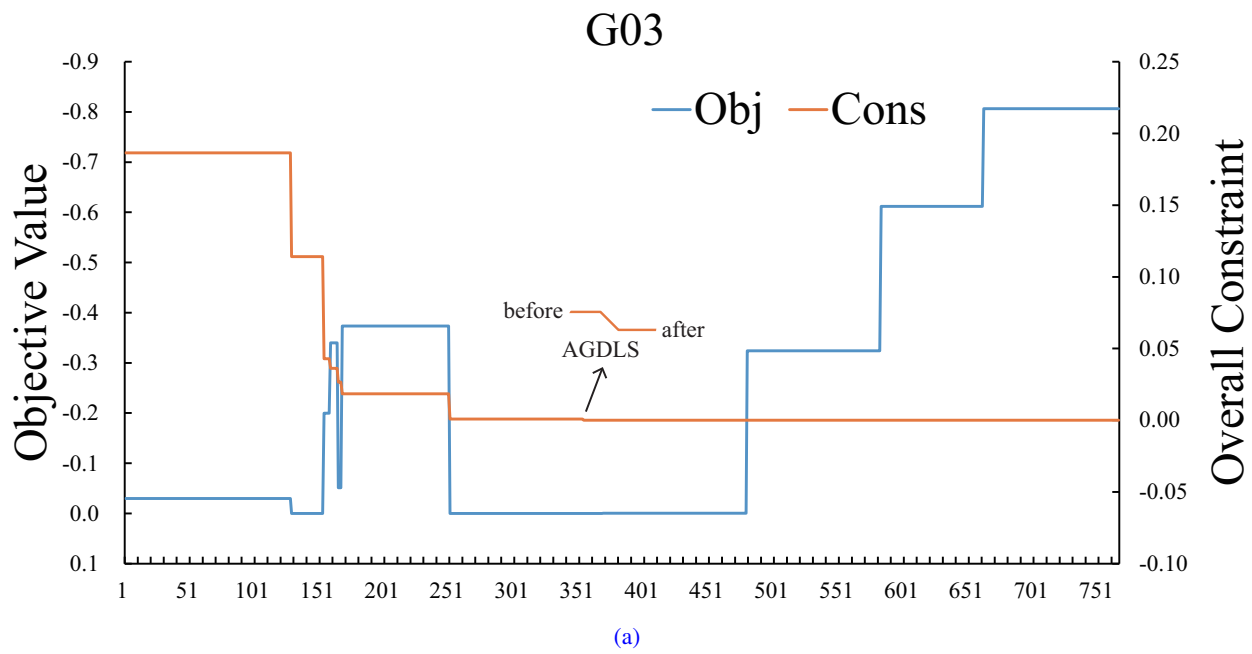
both *MCV* and *MOF* are marked as UA (unavailable). We also compute the success ratio (*SR*), i.e., the fraction of runs that attained feasibility. Pairwise algorithm comparisons follow a feasibility-first protocol: we prioritize *SR* then *MOF* (lower is better), and finally *MOF* among feasible results. Since GLSADE employs numerical gradients, we additionally record, for each problem, the average number of times the gradient is numerically computed over the 20 trials (denoted by *TG*).

4.2. Search ability of MLPS-based evolution

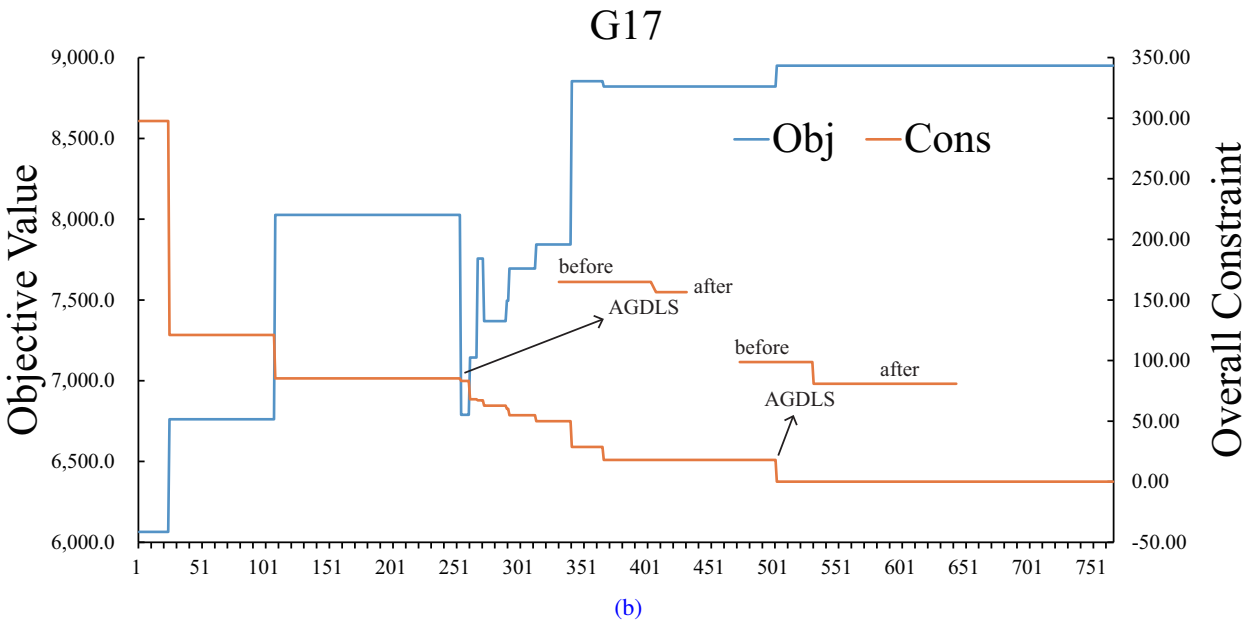
In SAEAs, surrogate models are commonly used to pre-screen offspring; most prior works [47, 48, 14] perform a single, comparison-based surrogate check without further surrogate-driven evolution. In contrast, we introduce an MLPS-based evolutionary stage (see Algorithm 1) that operates for both global and local search. Because MLPS evaluations are inexpensive relative to real function calls, we allow a long inner budget (2,000 iterations) to fully exploit a well-trained surrogate starting from the given population. To isolate the contribution of this surrogate-driven search, we also study a ablated variant, GLSADE-Sur, in which the while-loop in Algorithm 1 is truncated to a single iteration. Comparative outcomes for GLSADE-Sur and GLSADE are reported in Table 3; we additionally record the total execution time (*ET*) for completeness.

Table 3 shows that truncating the surrogate-driven loop to a single iteration GLSADE-Sur substantially weakens performance on G11, G14, G15, G17, C05, C06, and C06′. In five cases, no feasible solution is obtained across 20 runs; the success ratios on G11 and C06 drop to 0.45 and 0.35, respectively, indicating feasibility is rarely achieved. In contrast, enabling the full 2000-iteration surrogate evolution in Algorithm 1 markedly strengthens the search. With this setting, GLSADE attains SR = 1.0 on 13 of the 14 problems and SR = 0.9 on G14, meaning feasibility is reached in nearly every independent run.

The results indicate that the enhanced search capability primarily stems from the stronger candidate set produced by the MLPS-driven evolutionary stage. Population-based search refines solutions incrementally across generations, and this staged improvement also benefits surrogate training and usage. Allowing a long inner budget gives MLPS-based evolution enough iterations to mature the population and discover higher-quality solutions. The trade-off is computational: relative to the single-iteration variant GLSADE-Sur, the full GLSADE configuration incurs greater overall execution time to realize these gains.



(a)



(b)

Figure 4. Convergence plots of GLSADE on the G03 and G17 test functions. The plots show the objective function value and the overall degree of constraint violation versus the number of generations. The convergence changes caused by AGDLS are highlighted. (a) G03 test function. (b) G17 test function.

Table 4. MOF, MCV, and SR Obtained by GLSADE and GLSADE-WoLS.

Func.	GLSADE-WoLS			GLSADE			
Metrics	*MCV*	*MOF*	*SR*	*MCV*	*MOF*	*SR*	*TG*
G03	1.40E-04	-2.04E-04	0.20	**0.00E+00**	**-6.42E-01**	**1.00**	**0.0**
G11	0.00E+00	8.44E-01	1.00	**0.00E+00**	**7.50E-01**	**1.00**	**0.5**
G13	UA	UA	0.00	**0.00E+00**	**8.09E-01**	**1.00**	**2.0**
G14	UA	UA	0.00	**1.17E-04**	**-4.37E+01**	**0.90**	**3.0**
G15	UA	UA	0.00	**0.00E+00**	**9.63E+02**	**1.00**	**5.0**
G17	UA	UA	0.00	**0.00E+00**	**8.93E+3**	**1.00**	**4.4**
C05	UA	UA	0.00	**0.00E+00**	**2.26E+02**	**1.00**	**1.0**
C06	UA	UA	0.00	**0.00E+00**	**2.63E+02**	**1.00**	**2.0**
C09	UA	UA	0.00	**0.00E+00**	**1.52E+09**	**1.00**	**2.0**
C10	UA	UA	0.00	**0.00E+00**	**4.00E+09**	**1.00**	**3.0**
C05′	UA	UA	0.00	**0.00E+00**	**2.94E+02**	**1.00**	**1.8**
C06′	UA	UA	0.00	**0.00E+00**	**3.70E+02**	**1.00**	**1.0**
C09′	UA	UA	0.00	**0.00E+00**	**7.70E+12**	**1.00**	**3.4**
C10′	UA	UA	0.00	**0.00E+00**	**7.41E+12**	**1.00**	**2.0**

4.3. Effectiveness of AGDLS

In this work, we combine a GD-based local repair with a surrogate-guided local search to better cope with equality constraints. While surrogate-based local refinement is widely used and effective in expensive optimization, our emphasis here is on the efficacy of the proposed AGDLS, a local search scheme tailored specifically for EECOPs.

To activate AGDLS, both MLPS-based global and local phases must fail to yield an improvement. To quantify its contribution, we construct an ablated variant, GLSADE-WoLS, that disables AGDLS: after an unsuccessful MLPS-based local step, the algorithm immediately terminates the while-loop and resumes global search. Comparative outcomes for GLSADE-WoLS and the full GLSADE are reported in Table 4.

The contrast between GLSADE-WoLS and GLSADE is unambiguous. When AGDLS is disabled, GLSADE-WoLS fails on 12 of the 14 problems, whereas the full GLSADE, augmented with AGDLS, handles equality constraints far more effectively. Table 4 further indicates that AGDLS is invoked only sparingly; for most cases, no more than two finite-difference gradient evaluations are needed to refine feasibility. This modest overhead substantially improves the practicality of GLSADE for expensive constrained settings. Convergence traces for G03 and G17 (Figs. 4(a)–4(b)) show that, after periods of stagnation (orange segments), activating the hybrid local search yields small but decisive reductions in violation. In Fig. 4(a), the overall violation hovers near zero for roughly 100 generations prior to AGDLS—an indication that residual, fine-grained infeasibilities are difficult to resolve via global search alone. In such regimes, AGDLS provides the necessary fine-tuning, markedly strengthening GLSADE's ability to secure feasibility.

We ascribe the performance gains to the targeted deployment of AGDLS. The procedure is triggered only when no further improvement is detected, which keeps its invocation rate low. By that stage, the preceding MLPS-based global and local searches have typically pushed the population close to the feasible manifold, so AGDLS is well positioned to convert near-feasible candidates into feasible ones. This selective activation both limits overhead and markedly improves handling of equality constraints.

4.4. Effect of maximum count for stagnation

The parameter *stg* serves as a stagnation counter for the MLPS-based global search. Its inclusion is motivated by two considerations. First, the MLPS surrogate may be imperfect (e.g., over/underfitting), so

Table 5. MOF, MCV, and SR Obtained by GLSADE, GLSADE-3, and GLSADE-10.

Func.	GLSADE-3				GLSADE-10				GLSADE			
Metrics	*MCV*	*MOF*	*SR*	*TG*	*MCV*	*MOF*	*SR*	*TG*	*MCV*	*MOF*	*SR*	*TG*
G03	**0.00E+00**	**-7.86E-01**	**1.00**	**0.0**	0.00E+00	-6.53E-01	1.00	0.0	0.00E+00	-6.42E-01	1.00	0.0
G11	0.00E+00	9.17E-01	1.00	0.0	0.00E+00	7.92E-01	1.00	1.5	**0.00E+00**	**7.50E-01**	**1.00**	**0.5**
G13	3.65E-01	9.09E-01	0.75	2.0	7.78E-02	6.54E-01	0.55	2.0	**0.00E+00**	**8.09E-01**	**1.00**	**2.0**
G14	**0.00E+00**	**-4.63E+01**	**1.00**	**3.0**	0.00E+00	-4.44E+01	1.00	2.5	1.17E-04	-4.37E+01	0.90	3.0
G15	0.00E+00	9.63E+02	1.00	5.0	**0.00E+00**	**9.61E+02**	**1.00**	**3.5**	0.00E+00	9.63E+02	1.00	5.0
G17	0.00E+00	9.05E+03	1.00	5.8	**0.00E+00**	**8.93E+03**	**1.00**	**4.5**	**0.00E+00**	**8.93E+3**	**1.00**	**4.4**
C05	**0.00E+00**	**1.16E+02**	**1.00**	**2.0**	0.00E+00	2.20E+02	1.00	2.5	0.00E+00	2.26E+02	1.00	1.0
C06	**0.00E+00**	**9.16E+01**	**1.00**	**4.0**	6.37E-05	2.32E+02	0.35	3.0	0.00E+00	2.63E+02	1.00	2.0
C09	0.00E+00	1.92E+10	1.00	2.0	0.00E+00	6.26E+09	1.00	1.5	**0.00E+00**	**1.52E+09**	**1.00**	**2.0**
C10	0.00E+00	1.50E+10	1.00	2.6	0.00E+00	4.14E+09	1.00	1.0	**0.00E+00**	**4.00E+09**	**1.00**	**3.0**
C05′	**0.00E+00**	**2.86E+02**	**1.00**	**2.6**	8.37E-05	3.32E+02	0.65	2.5	0.00E+00	2.94E+02	1.00	1.8
C06′	0.00E+00	5.02E+02	1.00	3.0	5.70E-05	4.25E+02	0.85	1.0	**0.00E+00**	**3.70E+02**	**1.00**	**1.0**
C09′	**0.00E+00**	**4.91E+12**	**1.00**	**3.0**	0.00E+00	1.11E+13	1.00	1.5	0.00E+00	7.70E+12	1.00	3.4
C10′	0.00E+00	7.73E+12	1.00	3.0	0.00E+00	7.90E+12	1.00	1.5	**0.00E+00**	**7.41E+12**	**1.00**	**2.0**

declaring stagnation after a single miss is unreliable; several consecutive non-improving attempts provide a more robust signal. Second, the global phase relies on OFIE-ISC (Algorithm 2) to balance exploration and exploitation, so benefits may accrue gradually rather than via immediately greedy gains. Accordingly, *stg* throttles premature activation of the local search. To assess its effect, we evaluate two variants—GLSADE-3 and GLSADE-10—by fixing *stg* to 3 and 10, respectively; results are summarized in Table 5.

Compared with the baseline GLSADE, the variant GLSADE-3 is marginally worse overall, whereas GLSADE-10 exhibits a clear decline. Both GLSADE and GLSADE-3 successfully solve 13 of the 14 problems, while GLSADE-10 reaches feasibility on only 10. In addition, GLSADE-10 yields inferior *MCV* and *MOF* on most instances relative to the other two settings. Consistent with its design, GLSADE-3 triggers AGDLS more often than GLSADE, and GLSADE-10 invokes it the least. These outcomes indicate that an overly large *stg* prolongs the MLPS-based global phase, expending budget re-exploring an already narrow, well-searched feasible neighborhood and leaving too few real evaluations for effective refinement. Conversely, a very small *stg* (as in GLSADE-3) declares stagnation too quickly, increasing calls to both MLPS- and GD-based local search and spending more evaluations on finite-difference gradients; this reduces exploratory capacity and can precipitate early stagnation. Balancing these effects, a mid-range choice (approximately $stg = 6$) provides a more favorable tradeoff between exploration and local refinement while curbing unnecessary gradient computations.

4.5. Comparisons with state-of-the-art SAEAs

As a relatively new topic, few SAEAs are purpose-built for equality-constrained expensive optimization (EECOPs). To provide a reference baseline, we compare against four recent SAEAs originally designed for inequality-only ECOPs: GPEEC [49], ESAO-CH, SA-C^{2}oDE [15], and SParEA [23]. GPEEC, SA-C^{2}oDE, and SParEA employ GP surrogates (SParEA uses four surrogate types), while ESAO-CH is an enhanced version of ESAE [50] that uses radial basis function networks (RBFNs) to approximate the objective and each constraint. For a model-free reference, we also include εDEag [51], the winner of the CEC 2010 constrained real-parameter competition. All methods are run with parameter settings recommended in their original papers. Success-ratio *SR* outcomes are summarized in Table 6, with the best results highlighted in **bold**.

Table 6. The *SR* values obtained by GLSADE and the Five Competing SAEAs.

Func.	εDEag	GPEEC	ESAO-CH	SA-C^{2}oDE	SParEA	GLSADE
G03	0.00	0.80	0.92	0.96	**1.00**	**1.00**
G11	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
G13	0.00	0.20	0.28	0.52	0.16	**1.00**
G14	0.00	0.00	0.08	0.08	0.00	**0.90**
G15	0.00	0.00	0.00	0.00	0.00	**1.00**
G17	0.00	0.04	0.24	0.20	0.00	**1.00**
C05	0.00	0.00	0.45	0.00	0.00	**1.00**
C06	0.00	0.00	0.65	0.00	0.00	**1.00**
C09	0.00	0.00	0.00	0.00	0.00	**1.00**
C10	0.00	0.00	0.00	0.00	0.00	**1.00**
C05′	0.00	0.00	0.70	0.00	0.00	**1.00**
C06′	0.00	0.00	0.85	0.00	0.00	**1.00**
C09′	0.00	0.00	0.00	0.00	0.00	**1.00**
C10′	0.00	0.00	0.00	0.00	0.00	**1.00**

The model-free baseline, εDEag, yields $SR = 0$ on all cases—under the tight evaluation budget it is unable to enforce feasibility on any of the 14 problems. Likewise, the four surrogate-assisted comparators, originally engineered for inequality-only ECOPs, exhibit marked degradation on EECOPs: as summarized in Table 6, they fail to reach feasibility on most tests. This shortfall reflects designs (evolutionary operators and infill rules) tuned to inequality-bounded feasible sets and not to the thin manifolds imposed by equality constraints. In contrast, GLSADE consistently attains feasibility across the full suite within the same computational budget. These results underscore that equality-constrained expensive optimization poses a qualitatively different challenge and motivate the need for SAEA frameworks specifically crafted for EECOPs, such as GLSADE.

5. Conclusion

This work concentrates on constrained optimization problems with costly equality constraints, termed EECOPs. To tackle the vanishingly small feasible volume under tight evaluation budgets, we propose GLSADE, a tailored surrogate-assisted framework. GLSADE models the objective and equality constraints *jointly* via a multi-output MLPS regressor, enabling concurrent learning of the fitness landscape and constraint surfaces with deep learning. The search is organized to balance exploration and exploitation: an MLPS-driven evolutionary phase conducts global exploration toward feasibility, followed by a hybrid local refinement that targets near-feasible candidates. Compared with classical gradient-descent repair, the proposed local search, regulated by a stagnation detector, uses very few expensive evaluations while more effectively exploiting the feasible set. In combination, the MLPS-based global/local phases and the AGDLS repair module provide a practical paradigm for surrogate-assisted EAs to effectively solve expensive optimization with equality constraints.

We evaluated the performance of GLSADE on 14 benchmark test functions. In comparison with five competing algorithms, GLSADE demonstrated superior search ability of handling equality constraints with a limited expensive FEs and its overall superiority. The experimental study validates the effectiveness of the proposed MLPS-based evolution and AGDLS in seeking high-quality solutions.

Future work will concentrate on improving MLPS and OFIE-ISC in SAEAs to handle ECOPs with both equalities and inequalities.

References

[1] Z.-G. Chen, Z.-H. Zhan, S. Kwong, J. Zhang, Evolutionary computation for intelligent transportation in smart cities: A survey [review article], IEEE Computational Intelligence Magazine 17 (2) (2022) 83–102.

[2] Y. Zhou, Y. Qiu, S. Kwong, Region purity-based local feature selection: A multiobjective perspective, IEEE Transactions on Evolutionary Computation 27 (4) (2023) 787–801.

[3] Z.-J. Wang, J.-R. Jian, Z.-H. Zhan, Y. Li, S. Kwong, J. Zhang, Gene targeting differential evolution: A simple and efficient method for large-scale optimization, IEEE Transactions on Evolutionary Computation 27 (4) (2023) 964–979.

[4] J. Zhang, M. Li, X. Yue, X. Wang, M. Shi, A hierarchical surrogate assisted optimization algorithm using teaching-learning-based optimization and differential evolution for high-dimensional expensive problems, Applied Soft Computing 152 (2024) 111212.

[5] Y. Jin, Surrogate-assisted evolutionary computation: Recent advances and future challenges, Swarm and Evolutionary Computation 1 (2) (2011) 61–70.

[6] S.-C. Chu, X. Yuan, J.-S. Pan, B.-S. Lin, Z.-J. Lee, A multi-strategy surrogate-assisted social learning particle swarm optimization for expensive optimization and applications, Applied Soft Computing 162 (2024) 111876.

[7] S. Liu, H. Wang, W. Peng, W. Yao, A surrogate-assisted evolutionary feature selection algorithm with parallel random grouping for high-dimensional classification, IEEE Transactions on Evolutionary Computation 26 (5) (2022) 1087–1101.

[8] R. G. Regis, Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions, IEEE Transactions on Evolutionary Computation 18 (3) (2014) 326–347.

[9] F. Li, Z. Shang, Y. Liu, H. Shen, Y. Jin, Inverse distance weighting and radial basis function based surrogate model for high-dimensional expensive multi-objective optimization, Applied Soft Computing 152 (2024) 111194.

[10] J.-Y. Ji, M. L. Wong, Surrogate-assisted parameter re-initialization for differential evolution, in: 2022 IEEE Symposium Series on Computational Intelligence (SSCI), 2022, pp. 1592–1599.

[11] P. Lualdi, R. Sturm, A. Camero, T. Siefkes, An uncertainty-based objective function for hyperparameter optimization in gaussian processes applied to expensive black-box problems, Applied Soft Computing 154 (2024) 111325.

[12] Q. Zhang, W. Liu, E. Tsang, B. Virginas, Expensive multiobjective optimization by MOEA/D with Gaussian process model, IEEE Transactions on Evolutionary Computation 14 (3) (2010) 456–474.

[13] Y. Wang, D.-Q. Yin, S. Yang, G. Sun, Global and local surrogate-assisted differential evolution for expensive constrained optimization problems with inequality constraints, IEEE Transactions on Cybernetics 49 (5) (2019) 1642–1656.

[14] Y. Liu, J. Liu, Y. Jin, F. Li, T. Zheng, A surrogate-assisted two-stage differential evolution for expensive constrained optimization, IEEE Transactions on Emerging Topics in Computational Intelligence 7 (3) (2023) 715–730.

[15] F.-F. Wei, W.-N. Chen, W. Mao, X.-M. Hu, J. Zhang, An efficient two-stage surrogate-assisted differential evolution for expensive inequality constrained optimization, IEEE Transactions on Systems, Man, and Cybernetics: Systems (2023) 1–14.

[16] X. Cai, L. Gao, X. Li, Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems, IEEE Transactions on Evolutionary Computation 24 (2) (2020) 365–379.

[17] C. Sun, Y. Jin, R. Cheng, J. Ding, J. Zeng, Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems, IEEE Transactions on Evolutionary Computation 21 (4) (2017) 644–660.

[18] Y. Liu, J. Liu, J. Ding, S. Yang, Y. Jin, A surrogate-assisted differential evolution with knowledge transfer for expensive incremental optimization problems, IEEE Transactions on Evolutionary Computation (2023) 1–1.

[19] H. Zhen, W. Gong, L. Wang, F. Ming, Z. Liao, Two-stage data-driven evolutionary optimization for high-dimensional expensive problems, IEEE Transactions on Cybernetics 53 (4) (2023) 2368–2379.

[20] S. Qin, C. Sun, Q. Liu, Y. Jin, A performance indicator-based infill criterion for expensive multi-/many-objective optimization, IEEE Transactions on Evolutionary Computation 27 (4) (2023) 1085–1099.

[21] Z. Song, H. Wang, Y. Jin, A surrogate-assisted evolutionary framework with regions of interests-based data selection for expensive constrained optimization, IEEE Transactions on Systems, Man, and Cybernetics: Systems 53 (10) (2023) 6268–6280.

[22] F.-F. Wei, W.-N. Chen, Q. Li, S.-W. Jeon, J. Zhang, Distributed and expensive evolutionary constrained optimization with on-demand evaluation, IEEE Transactions on Evolutionary Computation 27 (3) (2023) 671–685.

[23] K. H. Rahi, H. K. Singh, T. Ray, Partial evaluation strategies for expensive evolutionary constrained optimization, IEEE Transactions on Evolutionary Computation 25 (6) (2021) 1103–1117.

[24] S. Bagheri, W. Konen, T. Back, Equality constraint handling for surrogate-assisted constrained optimization, in: 2016 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2016, pp. 1924–1931.

[25] Y. Su, L. Xu, E. D. Goodman, Hybrid surrogate-based constrained optimization with a new constraint-handling method, IEEE Transactions on Cybernetics 52 (6) (2020) 5394–5407.

[26] Z. Yang, H. Qiu, L. Gao, L. Chen, X. Cai, Constraint boundary pursuing-based surrogate-assisted differential evolution for expensive optimization problems with mixed constraints, Structural and Multidisciplinary Optimization 66 (2) (2023) 40.

[27] C. A. C. Coello, Constraint-handling techniques used with evolutionary algorithms, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '22, Association for Computing Machinery, New York, NY, USA, 2022, pp. 1310–1333.

[28] Y. Li, X. Lu, X. Yao, Negatively correlated search for constrained optimization, in: 2023 IEEE Congress on Evolutionary Computation (CEC), 2023, pp. 1–10.

[29] J.-Y. Ji, W.-J. Yu, J. Zhong, J. Zhang, Density-enhanced multiobjective evolutionary approach for power economic dispatch problems, IEEE Transactions on Systems, Man, and Cybernetics: Systems 51 (4) (2021) 2054–2067.

[30] K. Deb, An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering 186 (2) (2000) 311–338.

[31] R. Mallipeddi, P. N. Suganthan, Ensemble of constraint handling techniques, IEEE Transactions on Evolutionary Computation 14 (4) (2010) 561–579.

[32] Y. Wang, B.-C. Wang, H.-X. Li, G. G. Yen, Incorporating objective function information into the feasibility rule for constrained evolutionary optimization, IEEE Transactions on Cybernetics 46 (12) (2016) 2938–2952.

[33] C. M. Bishop, Neural networks for pattern recognition, Oxford University Press, 1995.

[34] J. Gu, W. Hua, W. Yu, Z. Zhang, H. Zhang, Surrogate model-based multiobjective optimization of high-speed pm synchronous machine: Construction and comparison, IEEE Transactions on Transportation Electrification 9 (1) (2023) 678–688.

[35] L. Wen, H. Wang, A meta-learning-based surrogate-assisted evolutionary algorithm for expensive multi-objective optimization problems, in: 2024 IEEE Congress on Evolutionary Computation (CEC), 2024, pp. 1–8.

[36] A. L. Maas, A. Y. Hannun, A. Y. Ng, et al., Rectifier nonlinearities improve neural network acoustic models, 2013.

[37] S. Lee, J. Y. Choeh, Predicting the helpfulness of online reviews using multilayer perceptron neural networks, Expert Systems with Applications 41 (6) (2014) 3041–3046.

[38] J. Zhang, A. C. Sanderson, JADE: Adaptive differential evolution with optional external archive, IEEE Transactions on Evolutionary Computation 13 (5) (2009) 945–958.

[39] P. Chootinan, A. Chen, Constraint handling in genetic algorithms using a gradient-based repair method, Computers & Operations Research 33 (8) (2006) 2263–2281.

[40] T. Takahama, S. Sakai, Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites, in: 2006 IEEE international conference on evolutionary computation, IEEE, 2006, pp. 1–8.

[41] J. J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. C. Coello, K. Deb, Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization, Journal of Applied Mechanics 41 (8) (2006) 8–31.

[42] J.-Y. Ji, Z. Tan, S. Zeng, E. W. K. See-To, M.-L. Wong, A surrogate-assisted evolutionary algorithm for seeking multiple solutions of expensive multimodal optimization problems, IEEE Transactions on Emerging Topics in Computational Intelligence 8 (1) (2024) 377–388.

[43] R. Chen, K. Li, Data-driven evolutionary multi-objective optimization based on multiple-gradient descent for disconnected pareto fronts, in: International conference on evolutionary multi-criterion optimization, Springer, 2023, pp. 56–70.

[44] Y. Liu, J. Liu, Y. Jin, F. Li, T. Zheng, A surrogate-assisted two-stage differential evolution for expensive constrained optimization, IEEE Transactions on Emerging Topics in Computational Intelligence 7 (3) (2023) 715–730.

[45] R. Mallipeddi, P. N. Suganthan, Problem definitions and evaluation criteria for the CEC 2010 competition on constrained real-parameter optimization, Nanyang Technological University, Singapore 24 (2010) 910.

[46] C. L. P. Chen, Z. Liu, Broad learning system: An effective and efficient incremental learning system without the need for deep architecture, IEEE Transactions on Neural Networks and Learning Systems 29 (1) (2018) 10–24.

[47] Y. Liu, Z. Yang, D. Xu, H. Qiu, L. Gao, A surrogate-assisted differential evolution for expensive constrained optimization problems involving mixed-integer variables, Information Sciences 622 (2023) 282–302.

[48] Z. Yang, H. Qiu, L. Gao, X. Cai, C. Jiang, L. Chen, Surrogate-assisted classification-collaboration differential evolution for expensive constrained optimization problems, Information Sciences 508 (2020) 50–63.

[49] B. Liu, Q. Zhang, G. Gielen, A surrogate-model-assisted evolutionary algorithm for computationally expensive design optimization problems with inequality constraints, in: S. Koziel, L. Leifsson, X.-S. Yang (Eds.), Simulation-Driven Modeling and Optimization, Springer International Publishing, Cham, 2016, pp. 347–370.

[50] X. Wang, G. G. Wang, B. Song, P. Wang, Y. Wang, A novel evolutionary sampling assisted optimization method for high-dimensional expensive problems, IEEE Transactions on Evolutionary Computation 23 (5) (2019) 815–827.

[51] T. Takahama, S. Sakai, Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation, in: IEEE Congress on Evolutionary Computation, 2010, pp. 1–9.