Neural Computing and Applications
https://doi.org/10.1007/s00521-023-08661-3

S.I. : HYBRID APPROACHES TO NATURE-INSPIRED OPTIMIZATION ALGORITHMS AND THEIR APPLICATIONS

Surrogate-assisted evolutionary sampling particle swarm optimization for high-dimensional expensive optimization

Kuihua Huang[1] · Huixiang Zhen[2] · Wenyin Gong[2] · Rui Wang[1] · Weiwei Bian[3]

Received: 1 December 2022 / Accepted: 10 May 2023

Abstract

Surrogate-assisted evolutionary algorithms have been widely employed for solving expensive optimization problems. To address high-dimensional expensive optimization problems, we propose an evolutionary sampling-assisted particle swarm optimization method, termed ESPSO. ESPSO consists of some evolutionary sampling-assisted strategies. It first improves the initialized population with some elite samples by evolutionary sampling. Secondly, during the optimization process, the method builds a local radial basis function model using the personal historical optimal data of the population to approximate the objective function landscape. Finally, surrogate-assisted local search and surrogate-assisted trust region search are designed to find promising candidate solutions for replacing individuals in the population to accelerate the search process. Behavioral research experiments of ESPSO verified these strategies have led to improvements in the search efficiency of the algorithm in various aspects, such as initialization, population update, and optimal solution promotion. We compared ESPSO with five state-of-the-art SAEAs using 18 benchmark functions, which show that ESPSO outperforms the other compared SAEAs and get the best average ranking of 2.194.

Keywords Evolutionary sampling · Expensive optimization · Particle swarm optimization · Radial basis function · Surrogate model

✉ Wenyin Gong
wygong@cug.edu.cn

Kuihua Huang
khhuang@nudt.edu.cn

Huixiang Zhen
zhenhuixiang@cug.edu.cn

Rui Wang
ruiwangnudt@gmail.com

Weiwei Bian
karovie@163.com

1 College of System Engineering, National University of Defense Technology, Changsha 410073, China

2 School of Computer Science, China University of Geosciences, Wuhan 430074, China

3 Equipment General Technology Laboratory, Beijing Mechanical Equipment Research Institute, Beijing 100854, China

1 Introduction

Evolutionary algorithms, such as particle swarm optimization [1, 2] (PSO), differential evolution [3] (DE), genetic algorithm [4] (GA), war strategy optimization (WSO) [5], have been extensively studied in the past two decades for solving complex global optimization problems and have demonstrated strong performance. However, in real-world applications, there is a wide range of expensive optimization problems whose single fitness evaluation (FE) takes minutes or even hours. For example, neural architecture search [6], blast furnace optimization [7], and finite element analysis [8]. Traditional evolutionary algorithms require numerous function evaluation times, such as hundreds of thousands, hindering their application to expensive optimization problems.

In recent years, more and more studies have shown that surrogate-assisted evolutionary algorithms (SAEAs) work well for expensive optimization problems, which combines

Published online: 29 May 2023

the predictive ability of the surrogate model and the search power of evolutionary algorithms. During optimization, SAEAs build cheap surrogate models with historical data and employ them to replace expensive fitness evaluation for reducing computational expense. Common surrogate models include radial basis function [9–11] (RBF), Random Forests [12] (RF), Kriging [13] (or Gaussian Process, GP), and support vector machine [14] (SVM). Many SAEAs have been designed by these surrogate models. Liu et al. used Gaussian Process model to assist DE to solve medium-scale problems [15]. For solving offline-constrained multiobjective combinatorial optimization in trauma systems, Wang et al. used the random forest as a surrogate model [12]. Yi et al. used the response surface function as a low-fidelity model for potential area detection [16]. Nguyen et al. proposed a constrained competitive swarm optimizer for feature selection where SVM is used as a surrogate model [17]. Although these methods show good performance of surrogate models, no single surrogate model fits all problems. Some studies on ensemble models and model selection have shown promising results. According to the committee-based active learning, Wang et al. proposed an ensemble surrogate-based global surrogate management strategy [18]. Moreover, Zhen et al. designed an offline model selection criterion to evaluate the most promising model for offline optimization [19].

Based on the different surrogate models, researchers have designed many surrogate model management strategies. They are used to sample candidate points for the real fitness evaluation and guide surrogate model updates. Among the many surrogate management strategies, the kriging-based filling criteria strategy is a very popular strategy. For example, expected improvement (EI) [20, 21], lower confidence bound (LCB) [15], and probability of improvement (PI) [22] have been used successfully to deal with different expensive optimization problems, including single-objective global optimization and multi-objective optimization [23–26]. Kriging-based methods generally work well on low-dimensional problems, whose number of decision variables is not more than 15 [27]. However, Kriging-based methods face challenges of the curse of dimension in high-dimensional expensive optimization problems [28]. On one hand, Kriging modeling becomes time-consuming with dimensions becoming larger. On the other hand, in the high-dimensional space, the uncertainty of candidate offspring becomes very similar so selecting promising offspring becomes difficult [29]. On the contrary, the RBF model is successfully employed in recent high-dimensional expensive optimization researches [28, 30, 31]. The reasons are that RBF has an excellent fitting performance on high-dimensional problems, and the modeling time of RBF does not increase significantly with increasing dimension [32].

Almost all traditional intelligent optimization methods [33, 34] have the potential to be used to solve expensive optimization problems. With the assistance of surrogate models, the number of expensive evaluations of intelligent optimization algorithms can be reduced, thus can solve expensive optimization problems. In SAEAs, the most commonly used EAs are GA [15], PSO [18], and DE [32], in addition to CSO [17], GWO [35], etc. Many SAEAs have been proposed based on traditional evolutionary algorithms. These approaches combine surrogate models with search mechanisms of different types of basic evolutionary algorithms organically. For example, Yu et al. [36] combined a coarse GP model and a fine RBF model and proposed a multimodel-based DE. Nguyen et al. [37] presented a surrogate-assisted PSO for feature selection that adjusts surrogate sets automatically for different datasets. In addition, many SAEAs have been proposed for different types of expensive optimization problems. Many expensive multi/many-objective algorithms have also been proposed [38, 39]. Chugh et al. [40] designed a kriging model-based reference vector-guided evolutionary algorithm, which builds kriging models for every objective function. To discontinuous objective functions, Wang et al. [41] conducted region division and then presented an RBF-assisted differential evolution method.

Moreover, expensive high-dimensional optimization is challenging. With dimension increases, search space becomes larger, surrogate model accuracy decreases significantly, and the modeling surrogate time increases accordingly. To address the high-dimension challenge, SAEAs for high-dimensional expensive optimization have gained widespread attention recently. Tian et al. [42] combined GP model and social learning particle swarm optimization (SLPSO) [43], where predicted fitness and uncertainty of GP are considered as two objectives of fill sampling. Sun et al. [44] presented an RBF-assisted particle swarm optimization, where PSO and SLPSO cooperatively search global optimum. The two search mechanisms focus on exploration and local search, respectively. Yu et al. [30] presented an RBF-assisted hierarchical particle swarm optimization method (SHPSO), which uses PSO and SLPSO to work together balancing exploration and exploitation. Li et al. [45] used two swarms in the optimization process: one swarm focus on exploration by teaching-learning-based optimization and the other one searches by the PSO for faster convergence. In addition, evolutionary sampling strategies have been used widely in SAEAs [32, 46] recently, which has shown good performance in accelerating algorithm convergence. Zhen et al. [10] designed a two-stage method that improves exploitation ability with surrogate-assisted evolutionary sampling in the second stage. In [11], neighborhood evolutionary sampling with dynamic repulsion has shown good performance for expensive multimodal optimization.

According to previous research, surrogate-assisted PSO methods usually have good exploration performance and evolutionary sampling-based methods have good exploitation performance. However, few studies combine their advantages organically. To balance exploration and exploitation, we propose an evolutionary sampling-assisted particle swarm optimization algorithm (ESPSO), which combines the advantages of evolutionary sampling and particle swarm optimization. The contributions of this research can be outlined as follows:

1. In the optimization process, we employ evolutionary sampling techniques to search for initial elite samples, thereby enhancing the quality of the initial population. Furthermore, in the population update phase, we use two surrogate-assisted evolutionary sampling strategies, namely surrogate-assisted local search and surrogate-assisted trust region search, to obtain promising candidate solutions that update the population and expedite population convergence.
2. When constructing the surrogate model, we utilize the individual historical optimal data of the population as the modeling data. This approach allows for swift modeling with a fixed amount of data, while also enabling adaptive adjustments based on the search process, with a focus on the current search area.
3. In this paper, we propose an evolutionary sampling-assisted PSO algorithm. This approach effectively balances exploration and exploitation and demonstrates promising results in solving expensive optimization problems. The proposed algorithm outperforms the other five state-of-the-art SAEAs on benchmark functions.

The remaining sections of this paper are organized as follows. In Sect. 2, we provide a brief introduction to related techniques. The details of ESPSO are presented in Sect. 3. Subsequently, in Sect. 4, we conduct comparison experiments with other surrogate-assisted evolutionary algorithms and present the behavioral analysis of ESPSO. Finally, we conclude this paper and future work in Sect. 5.

2 Preliminary techniques

2.1 Radial basis function

RBF [9, 47] is a widely used surrogate model. Recent researches show it is a very suitable surrogate model in high-dimensional expensive optimization problems. RBF can fit high-dimensional nonlinear functions well and is insensitive to the increase of dimension [28, 30]. RBF can be seen as several basis functions weighted sum. When there are N sample data $\{(\boldsymbol{x}_i, y_i) \mid \boldsymbol{x}_i \in R^d, i = 1, \cdots, N\}$, where d is the dimension of the problem, $\boldsymbol{x}_i$ is the solution and y_i is the corresponding fitness value. The RBF model trained with those data can be expressed as:

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{N} \lambda_i \varphi(\|\boldsymbol{x} - \boldsymbol{x}_i\|). \tag{1}$$

where $\varphi(\cdot)$ is radial basis kernels function, $\|\cdot\|$ is Euclidean norm, and λ_i represents the weight coefficient of ith kernel function. Common kernel functions include cubic splines, Gaussian function, multi-quadrics splines, and so on. In this work, we use Gaussian function $\varphi(x) = \exp(-x^2/\theta)$ as kernel function, where θ controls the scope of the Gaussian kernel function. With θ value becoming larger, the Gaussian kernel function local influence scope also will increase. We empirically set shape parameter $\theta = D_{\max}(dN)^{-1/d}$, where $D_{\max}$ means the maximal distance between training data. It is worth noting that parameter θ can be obtained once given the training dataset. Thus, RBF network can be obtained efficiently. In addition, the weight vector $\boldsymbol{\lambda} = (\lambda_i, \ldots, \lambda_N)^T$ is computed as follows:

$$\boldsymbol{\lambda} = \boldsymbol{\Phi}^{-1}\mathbf{F} \tag{2}$$

where $\mathbf{F} = [y_1, y_2, ..., y_N]^T$ and kernel matrix $\Phi = \left[\varphi\left(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\right)\right]_{N\times N}, i, j = 1, 2, ..., N$. When input data $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$ are all different, the interpolation matrix is positive definite.

2.2 Particle swarm optimizer

PSO [48, 49] algorithm first initializes a population consisting of NP different particles. Particles have two properties, position, and velocity. In the search process, PSO updated position and velocity by the following search mechanism:

$$\begin{aligned} \boldsymbol{v}_i^{(g+1)} =& \omega \boldsymbol{v}_i^{(g)} + c_1 r_1 \left(\boldsymbol{pbest}_i^{(g)} - \boldsymbol{x}_i^{(g)}\right) \\ &+ c_2 r_2 \left(\boldsymbol{gbest}^{(g)} - \boldsymbol{x}_i^{(g)}\right) \end{aligned} \tag{3}$$

$$\boldsymbol{x}_i^{(g+1)} = \boldsymbol{x}_i^{(g)} + \boldsymbol{v}_i^{(g+1)} \tag{4}$$

where $\boldsymbol{v}_i^{(g)}$ and $\boldsymbol{x}_i^{(g)}$ means velocity and position of the ith particle at gth generation. $\boldsymbol{gbest}^{(g)}$ denotes the global optimum position and $\boldsymbol{pbest}_i^{(g)}$ means the ith particle's personal historical optimum position. And c_1 and c_2 are two weight coefficients for personal cognitive and social cognitive. ω denotes an inertia weight, which adapts based on the search process [50]. In this paper, $\omega = 0.9 - 0.5 * \mathrm{NFE}/\mathrm{NFE}_{\max}$, where NFE denotes the

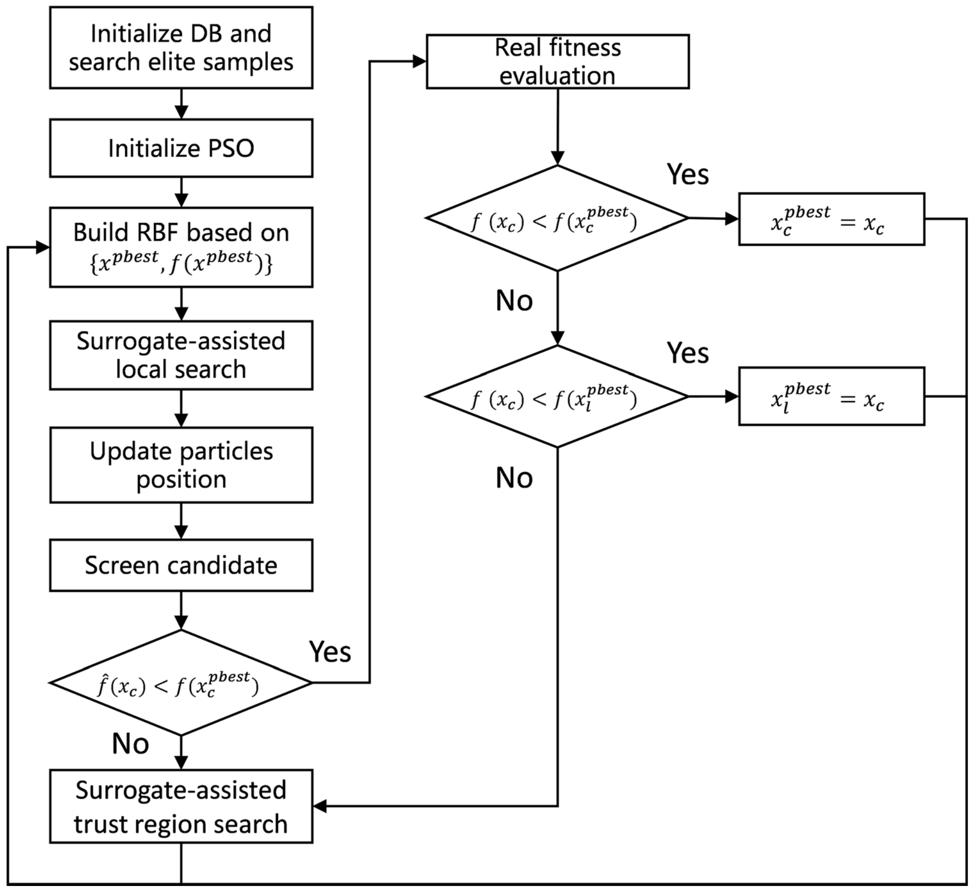

Fig. 1 Framework of ESPSO

currently used real fitness evaluation number, and NFE_{max} is the maximum number of real fitness evaluations. We set c_1 and c_2 to 2 in this work. r_1 and r_2 both are random numbers distributed within [0, 1] uniformly.

3 The proposed method: ESPSO

The proposed algorithm, known as the evolutionary sampling particle swarm optimization, capitalizes on the benefits of both particle swarm optimization and evolutionary sampling strategies. It first improves the initialized population with some elite samples by evolutionary sampling. Secondly, during the optimization process, the method builds a local radial basis function model using the personal historical optimal data of the population to approximate the objective function landscape. ESPSO efficiently screens candidate solutions for real fitness evaluation and population update, leading to enhanced convergence. Additionally, ESPSO employs two evolutionary sampling strategies to search for potential replacements for the worst particle in the population, thereby further accelerating population convergence. Based on the above strategies, it has an excellent performance in solving expensive optimization problems. The framework of ESPSO is illustrated in Fig. 1.

3.1 ESPSO framework

The pseudocode of ESPSO is shown in Algorithm 1. In ESPSO, the database ***DB*** is initialized first by LHS [51], and ESPSO uses surrogate-assisted local search to generate some elite samples that are added into ***DB***. Then, ESPSO initializes the PSO population based on the ***DB***. We used the personal historical optimum data of particles as train samples to build RBF model $\hat{f}_p$. RBF model training data change with population. The RBF model is built based on a fixed amount of data, which is population size during optimization. As the population converges, the population-based surrogate model will focus on the area being searched.

After the surrogate model is built, ESPSO searches one candidate solution by minimizing $\hat{f}_p$ using DE. For improving the candidate solution, we take the crossover between the candidate solution and the global best solution, then obtain an improved candidate solution $\boldsymbol{x_c}$. Then we calculate real fitness evaluation of the candidate solution and add the new data to the database. In addition, ESPSO updates the position of the particles of PSO and screens the most promising particle whose predicted fitness is minimum as a candidate solution $\boldsymbol{x_c}$. $\hat{f}(\boldsymbol{x_c})$ means predicted fitness by population-based surrogate and $f\left(\boldsymbol{x_c^{pbest}}\right)$ means the real fitness of personal historical optimum of $\boldsymbol{x_c}$. If $\hat{f}(\boldsymbol{x_c}) < f\left(\boldsymbol{x_c^{pbest}}\right)$ is false, this means that surrogate-assisted PSO screening may not yield a better solution. Therefore, we do not evaluate this candidate solution and use surrogate-assisted trust region search to find new solutions. On the contrary, if $\hat{f}(\boldsymbol{x_c}) < f\left(\boldsymbol{x_c^{pbest}}\right)$, that is the predicted fitness of $\boldsymbol{x_c}$ is less than its personal historical optimum, this means that surrogate-assisted PSO screening may yield a better solution. We calculate its real fitness evaluation $f(\boldsymbol{x_c})$. When the candidate solution has better fitness than its personal historical optimum, we update its personal historical position. If the candidate solution fitness has worse than its personal historical optimum but better than the worst personal historical optimum $\boldsymbol{x_l^{pbest}}$ of the population, the worst personal historical optimum $\boldsymbol{x_l^{pbest}}$ and the particle $\boldsymbol{x_l}$ is substituted with the candidate solution. However, when the candidate solution fitness is worse than the worst personal historical optimum of the population, it means the surrogate-assisted PSO screening cannot find a better solution. So we used a trust region search based on RBF to find new candidate solutions. Finally, ESPSO updates the global best solution $\boldsymbol{x^{gbest}}$. The algorithm iteratively searches until a termination condition is reached.

Algorithm 1 ESPSO

Initialize $\boldsymbol{DB}$ by LHS, which consist of NP sample points;
$NFE = NP$;
for $i = 1$ *to* m **do**
  Build global surrogate model $\hat{f}_g$ based on $\boldsymbol{DB}$;
  Search candidate solution by minimizing $\hat{f}_g$ using SLPSO [43];
  Calculate the real fitness evaluation of the candidate solution, add the new sample to $\boldsymbol{DB}$, and $NFE = NFE + 1$;
end
Initialize population $\boldsymbol{P}$ with the best NP samples in $\boldsymbol{DB}$;
for $NFE < NFE_{max}$ **do**
  Build surrogate model $\hat{f}_p$ based on particle's personal best position of current population;
  Use surrogate-assisted local search (Algorithm 2);
  Update the new position of the particles of PSO;
  Screen the particle whose predicted fitness $\hat{f}_p(\boldsymbol{x_c})$ is minimum as candidate solution $\boldsymbol{x_c}$;
  if $\hat{f}_p(\boldsymbol{x_c}) < f\left(\boldsymbol{x_c^{pbest}}\right)$ **then**
    Calculate the real fitness evaluation $f(\boldsymbol{x_c})$ of the candidate solution, add it to $\boldsymbol{DB}$, and $NFE = NFE + 1$;
    if $f(\boldsymbol{x_c}) < f\left(\boldsymbol{x_c^{pbest}}\right)$ **then**
      $\boldsymbol{x_c^{pbest}} = \boldsymbol{x_c}$;
    end
    else
      if $f(\boldsymbol{x_c}) < f\left(\boldsymbol{x_l^{pbest}}\right)$ **then**
        $\boldsymbol{x_l^{pbest}} = \boldsymbol{x_c}$; $\boldsymbol{x_l} = \boldsymbol{x_c}$;
      end
      else
        Use STR (Algorithm 4);
      end
    end
  end
  else
    Use STR (Algorithm 4);
  end
  Update global best $\boldsymbol{x^{gbest}}$;
end

3.2 Surrogate-assisted local search

To accelerate population convergence, we use surrogate-assisted local search in each generation to obtain one candidate solution to replace the worst particle $\boldsymbol{x_l}$ of the population and updated its personal historical optimum $\boldsymbol{x_l^{pbest}}$. Algorithm 2 shows the pseudocode of surrogate-assisted local search. Firstly, we calculate the range $[\boldsymbol{lb}, \boldsymbol{ub}]$ of local search based on Eqs. 5 and 6, where k denotes the data size building the population-based surrogate model, and d denotes the variable dimension. Then, we utilize DE to search for a candidate solution $\boldsymbol{x_c}$ within this search range by minimizing the surrogate model $\hat{f}_p$.

$$\begin{cases} \boldsymbol{lb} = [l_1, l_2, \ldots, l_d] \\ \boldsymbol{ub} = [u_1, u_2, \ldots, u_d] \end{cases} \quad (5)$$

$$\begin{cases} l_j = \min\left(x_j^i\right) \\ u_j = \max\left(x_j^i\right) \end{cases} i = 1, 2, \ldots, k \quad (6)$$

To improve $\boldsymbol{x_c}$, we take crossover $\boldsymbol{x_c}$ with the global best solution to improve the candidate solution, as shown in Algorithm 3. In Algorithm 3, we input candidate solution $\boldsymbol{x_c}$, global best solution $\boldsymbol{x^{best}}$ and surrogate model $\hat{f_p}$. At first, $\boldsymbol{x_c}$ and $\boldsymbol{x^{best}}$ form a population $\boldsymbol{P_c}$. Then, we generate a random variable sequence $\boldsymbol{M}$ for the variable dimension. In $\boldsymbol{P_c}$, the best solution is assigned to $\boldsymbol{x_b}$. After that, the algorithm takes crossover at each variable iteratively and selects the promising solution by $\hat{f_p}$ as $\boldsymbol{x_b}$. When all variables have been dealt with, the $\boldsymbol{x_b}$ is assigned to $\boldsymbol{x_c}$. Then $\boldsymbol{x_c}$ is return to Algorithm 2. Finally, we calculate the real fitness evaluation of $\boldsymbol{x_c}$, and add $\{\boldsymbol{x_c}, f(\boldsymbol{x_c})\}$ into the database $\boldsymbol{DB}$. After getting the improved candidate solution, we compute the real fitness evaluation of this candidate solution. If its real fitness evaluation is better than the worst personal historical optimal solution $\boldsymbol{x_l^{pbest}}$, we use the candidate solution to replace the worst particle $\boldsymbol{x_l}$.

3.3 Surrogate-assisted trust region search

In ESPSO, we use surrogate-assisted trust region search (STR) to obtain new solutions when the PSO-screened solutions perform poorly. There are two cases here. One situation is when the predicted fitness value of the candidate solution screened out by PSO is worse than its personal historical optimum. Instead of evaluating the true fitness value, we use STR. Another case is that when the real fitness evaluation value of the candidate solution is worse than the worst personal historical optimum $\boldsymbol{x_l^{pbest}}$ of population, we use STR to obtain new solutions to improve PSO population. The pseudocode for STR is shown in Algorithm 4. STR considers iterating $h_{\max}$ times in the search process. $h_{\max}$ is set to 3, which is the same setting as [31]. In each iteration, STR first built a population-based RBF model $\hat{f_p}$. The model train data are the personal historical optimum of the particles in trust region $[\boldsymbol{x_{best}} - \Delta^k, \boldsymbol{x_{best}} + \Delta^k] \cap [\boldsymbol{lb}, \boldsymbol{ub}]$, where $\boldsymbol{x_{best}}$ is the best sample point. $[\boldsymbol{lb}, \boldsymbol{ub}]$ is obtained based on Eqs. 5 and 6, where k is the size of data building the population-based surrogate model. Moreover, the initial trust region radius Δ^0 is half of the distance between the minimum and the maximum response points. Radius Δ^h updating is as follows:

Algorithm 2 Surrogate-assisted local search

Obtain $[\boldsymbol{lb}, \boldsymbol{ub}]$ by Eqs. 5 and 6;
Search candidate solution $\boldsymbol{x_c}$ in $[\boldsymbol{lb}, \boldsymbol{ub}]$ by minimizing $\hat{f}_p$ using DE;
Take crossover between candidate solution and global best solution to improve $\boldsymbol{x_c}$ (Algorithm 3);
Calculate the real fitness evaluation of the candidate solution, add it to $\boldsymbol{DB}$, and $NFE = NFE + 1$;
if $f(\boldsymbol{x_c}) < f\left(\boldsymbol{x_l^{pbest}}\right)$ **then**
    $\boldsymbol{x_l^{pbest}} = \boldsymbol{x_c}$; $\boldsymbol{x_l} = \boldsymbol{x_c}$;
end

Algorithm 3 Crossover with global best solution

Input: Candidate solution $\boldsymbol{x_c}$, global best $\boldsymbol{x^{gbest}}$, surrogate model $\hat{f}_p$
Output: New candidate solution $\boldsymbol{x_c}$
$\boldsymbol{P_c} = [\boldsymbol{x_c}, \boldsymbol{x^{gbest}}]$;
Random sequence of decision variables $\boldsymbol{M}$ = randperm(d);
Find the best solution in $\boldsymbol{P_c}$ as $\boldsymbol{x_b}$;
for *each i in* $\boldsymbol{M}$ **do**
    $\boldsymbol{P_t} = [\boldsymbol{x_b}; \boldsymbol{x_b}]_{2\times d}$;
    $\boldsymbol{P_t}[:, i] = \boldsymbol{P_c}[:, i]$;
    Screen the most promising solution $\boldsymbol{u}$ from $\boldsymbol{P_t}$ by $\hat{f}_p$;
    $\boldsymbol{x_b} = \boldsymbol{u}$;
end
$\boldsymbol{x_c} = \boldsymbol{x_b}$;

$$\Delta^{h+1} = \begin{cases} 0.25\Delta^h & \text{if} \quad \rho^h \leq 0.25 \\ \Delta^h & \text{if} \quad 0.25 \leq \rho^h \leq 0.75 \\ \xi\Delta^h & \text{if} \quad \rho^h \geq 0.75 \end{cases} \tag{7}$$

After setting the trust region, we obtain candidate point $\boldsymbol{x_c}$ with minimizing $\hat{f_p}$. Then we calculated its real fitness evaluation and added $\{\boldsymbol{x_c}, f(\boldsymbol{x_c})\}$ into $\boldsymbol{D_{new}}$. If fitness of $\boldsymbol{x_c}$ is better than $\boldsymbol{x_{best}}$, $\boldsymbol{x_{best}} = \boldsymbol{x_c}$. Then the trust ratio ρ^h is calculated, the radius of Δ^{h+1} of the trust region is updated, and $h = h + 1$. When h is not smaller than $h_{\max}$, STR updates the PSO population based on $\boldsymbol{D_{new}}$. By iterating over each solution $\boldsymbol{x_c}$ in $\boldsymbol{D_{new}}$, STR compares $\boldsymbol{x_c}$ with the worst personal historical optimum $\boldsymbol{x_l^{pbest}}$. If fitness value of $\boldsymbol{x_c}$ is better than $\boldsymbol{x_l^{pbest}}$, $\boldsymbol{x_l^{pbest}}$ is replaced with $\boldsymbol{x_c}$. By the way, STR improved the PSO population quality.

assessed using six widely-used test functions, as described in Table 1. These test functions encompassed a range of characteristics, including unimodal and multimodal functions, as well as highly complex multimodal functions. The problem dimensions varied from 30 to 100 dimensions. ESPSO was executed on these six test functions in three dimensions, and the results are presented in Table 2. Notably, ESPSO exhibited excellent performance in solving the Ellipsoid, Rosenbrock, and Ackley functions, even in the challenging 100-dimensional case, obtaining satisfactory solutions. However, ESPSO is average in RHC problems since it has a very complicated multimodal function landscape. Further analysis will be introduced in the following comparative experiments.

In comparison experiments with other state-of-the-art SAEAs. We used the Wilcoxon rank-sum test at the 5% level of significance. ESPSO performed better in most test functions. Meanwhile, the Friedman test is used for ranking them. ESPSO got the best average ranking. In Figs. 2, 3 and 4, the convergence curves show the mean value of 20 independent times run results. Finally, we conducted behavioral research of ESPSO to verify the effectiveness of improved strategies.

Algorithm 4 STR.

```
Set the search range [lb, ub] of population based on Eqs. 5 and 6;
h_max = 3, h = 0;
for h < h_max do
    Build population-based local RBF model f̂_p based on personal historical optimal
    data of current population in trust region [x_best − Δ^h, x_best + Δ^h] ∩ [lb, ub]
    where x_best is the current best solution;
    Search candidate solution x_c in trust region by minimizing f̂_p;
    Add {x_c, f(x_c)} into D_new, and NFE = NFE + 1;
    if f(x_c) < f(x_best) then
        x_best = x_c;
    end
    ρ^h = (f(x_best) − f(x_c)) / (f̂_p(x_best) − f̂_p(x_c));
    Calculate Δ^{h+1} by Eq. 7;
    h = h + 1;
end
for each x_c in D_new do
    Find the particle x_l who has the worst personal historical optimum x_l^pbest of
    the population;
    if f(x_c) < f(x_l^pbest) then
        x_l^pbest = x_c; x_l = x_c;
    end
end
```

4 Numerical experiments

To verify the proposed algorithm, five state-of-the-art SAEAs are used to compare with ESPSO. For fairness, we set the maximum real fitness evaluation budget to 1000, and all numerical experiments in this paper are run independently 20 times. The performance of ESPSO was

4.1 Parameter settings

Experiment environment and parameters setting are shown in Table 3. Firstly, ESPSO generates initial data by LHS. The initial data size is the same as PSO population size *NP*. *NP* is 80 if the dimension is less than 100, otherwise is set to 120. Then ESPSO uses surrogate-assisted local search to generate some elite samples that are added into ***DB***, where parameter m is 10. For PSO parameters, $c_1 = 2$, $c_2 = 2$, and $\omega = 0.9 - 0.5 * \mathrm{NFE}/\mathrm{NFE}_{\max}$. In surrogate-assisted trust region search, $h_{\max} = 3$.

4.2 Comparison with five state-of-the-art DDEAs

We compared ESPSO with five state-of-the-art SAEAs to verify the algorithm effect. The five SAEAs consist of SHPSO [30], ESAO [28], SAMSO [45], CA-LLSO [53] and SA-HLWCA [54]. Brief description of the comparison algorithm are shown in Table 4. Their parameter setting is the same as their original papers. Table 5 shows their test results, and the mean and variance are obtained by 20 independent runs. In each row of the table, bold results are the best result on the corresponding test problem. Moreover, we used Wilcoxon rank-sum test to compare the performance of ESPSO and five methods. In this table, the symbols "+" means ESPSO is significantly better than the compared algorithm, "−" is ESPSO is worse than it significantly, and "≈" is their performance is similar.

In the comparison between ESPSO and SHPSO, ESPSO outperformed SHPSO in almost all test questions. ESPSO is significantly better than SHPSO on 14 of the 18 test questions. Although SHPSO also includes a local search strategy based on SLPSO, in the PSO population screening part, the SHPSO may screen many offspring for true fitness evaluation at each generation. However, in the surrogate-assisted PSO screening, only one candidate solution is screened each time for fitness evaluation, which saves the number of evaluations. ESPSO significantly outperformed SHPSO on most test questions. But for very complex multimodal problems, SHPSO screens many offspring for true fitness evaluation, retaining more diversity, so SHPSO works better on SRR-100D and RHC-50.

In the comparison between ESPSO and ESAO, the results of ESPSO are significantly better than ESAO on 12 of the 18 test questions. ESAO combines two evolutionary sampling strategies, alternately using RBF-assisted DE offspring screening and RBF-assisted local search. However, the convergence ability of ESAO is not good enough due to the lack of information exchange with RBF-assisted local search due to maintaining a separately evolved DE population. Different from ESAO, ESPSO will replace the worst particles in the PSO population after obtaining new candidate points by local search, thus the convergence ability of the algorithm is improved. Comparison results show that model management of ESPSO has more effective performance.

In the comparison between ESPSO and SAMSO, ESPSO is significantly better than SAMSO on 11 of the 18 test questions. In each iteration, SAMSO constructs a global RBF with all sample data, then employed two population-based optimization methods, TLBO and PSO, to search for the optimal value. ESPSO is significantly better than SAMSO in the four problems of Ellipsoid, Rosenbrock, Ackley, and Griewank, because ESPSO adopts evolutionary sampling strategies to assist PSO algorithm, which accelerates the convergence of the population. However, SAMSO adopts a dual-population search mechanism. The complex multimodal optimization problems SRR and RHC perform better.

In the comparison between ESPSO and CA-LLSO, ESPSO is significantly better than CA-LLSO on 16 of the 18 test questions. CA-LLSO is a classifier-assisted optimization algorithm. Relatively speaking, the improved PSO algorithm ESPSO based on evolutionary sampling has a stronger convergence ability. As shown in Table 5,

Table 1 Brief description of the benchmark functions

Problem	Dimensions (d)	Property	Domain of $\boldsymbol{x}$	Optimum
Ellipsoid	30, 50, 100	Unimodal	$[-5.12, 5.12]^d$	0
Rosenbrock	30, 50, 100	Multimodal	$[-2.048, 2.048]^d$	0
Ackley	30, 50, 100	Multimodal	$[-32.768, 32.768]^d$	0
Griewank	30, 50, 100	Multimodal	$[-600, 600]^d$	0
Shifted rotated rastrigin function (SRR) (F10 in CEC05) [52]	30, 50, 100	Very complicated multimodal	$[-5, 5]^d$	− 330
Rotated hybrid composition function (RHC) (F19 in CEC05) [52]	30, 50, 100	Very complicated multimodal	$[-5, 5]^d$	10

Table 2 Results of ESPSO on benchmark test functions

Problems	*d*	Best	Worst	Median	Mean	Std
Ellipsoid	30	2.84E−17	6.42E−12	2.94E−16	3.28E−13	1.44E−12
Ellipsoid	50	3.11E−10	2.12E−08	1.19E−09	3.32E−09	5.29E−09
Ellipsoid	100	2.49E−03	6.30E−03	3.47E−03	3.49E−03	8.91E−04
Rosenbrock	30	2.64E+01	2.94E+01	2.82E+01	2.82E+01	6.35E−01
Rosenbrock	50	4.74E+01	4.98E+01	4.84E+01	4.84E+01	5.04E−01
Rosenbrock	100	9.75E+01	1.02E+02	9.84E+01	9.88E+01	1.19E+00
Ackley	30	1.05E−04	1.18E−03	2.26E−04	3.85E−04	3.45E−04
Ackley	50	1.50E−03	1.76E−01	2.24E−03	1.34E−02	3.89E−02
Ackley	100	1.39E+00	2.08E+00	1.68E+00	1.73E+00	2.07E−01
Griewank	30	8.35E−05	4.13E−02	6.04E−04	4.97E−03	1.03E−02
Griewank	50	4.99E−03	2.05E−01	1.38E−02	3.18E−02	4.91E−02
Griewank	100	4.07E−01	8.42E−01	5.76E−01	6.06E−01	1.11E−01
SSR	30	− 2.76E+02	− 6.07E+01	− 1.95E+02	− 1.87E+02	6.34E+01
SSR	50	− 1.75E+02	1.13E+02	− 8.39E+01	− 4.81E+01	1.00E+02
SSR	100	7.55E+02	1.19E+03	8.52E+02	8.91E+02	1.06E+02
RHC	30	9.24E+02	9.81E+02	9.42E+02	9.48E+02	1.76E+01
RHC	50	9.82E+02	1.05E+03	1.03E+03	1.02E+03	2.01E+01
RHC	100	1.31E+03	1.44E+03	1.36E+03	1.37E+03	3.47E+01

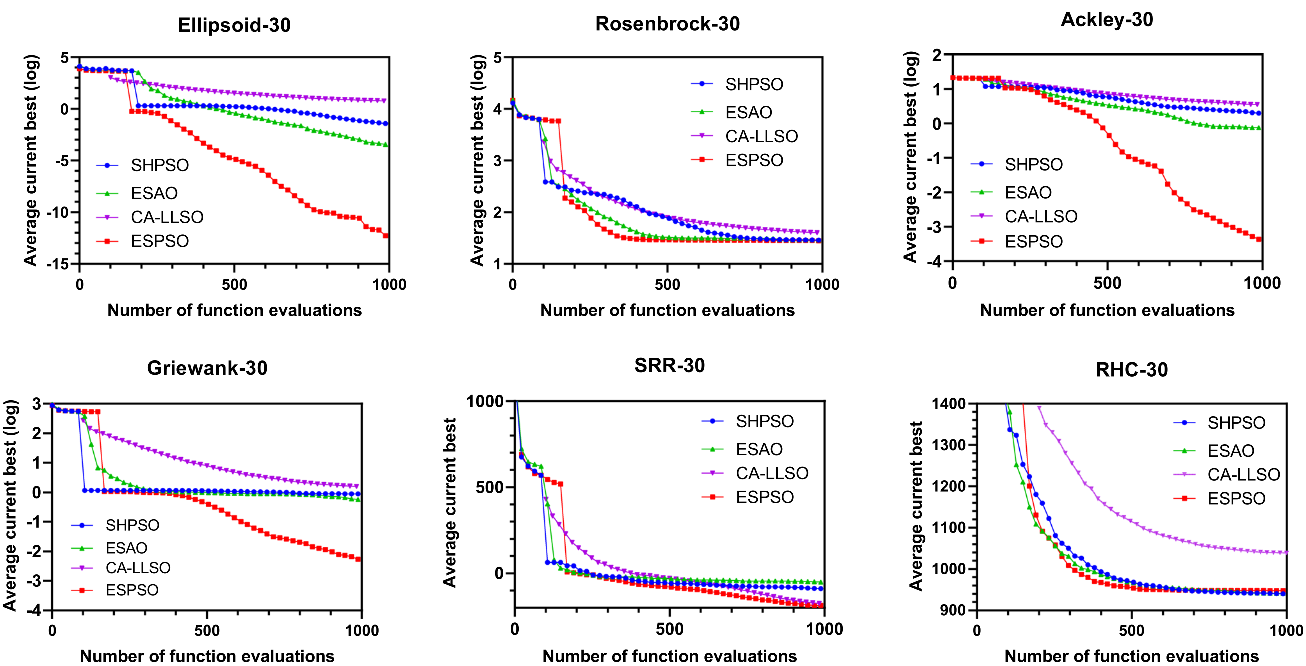

Fig. 2 Convergence curves of different algorithms for the 30D benchmark problems

ESPSO works better on all tested functions except the 100-dimensional RHC and the 30-dimensional SRR.

In the comparison between ESPSO and SA-HLWCA, ESPSO is significantly better than SA-HLWCA on 12 of the 18 test questions. SA-HLWCA also combines global and local searches. It has a similar framework to ESAO. The experiment result is similar to ESAO too. We can see that ESPSO significantly performs better in most problems except RHC problems than SA-HLWCA.

In addition, we use the Friedman test to rank the performance of these algorithms. ESPSO has the best average ranking at 2.194. ESPSO is significantly better than ESAO at 4.028, SHPSO at 4.333, and CA-LLSO at 5.389. And ESPSO is slightly better than SA-HLWCA at 2.306 and SAMSO at 2.75. The convergence diagram is shown in

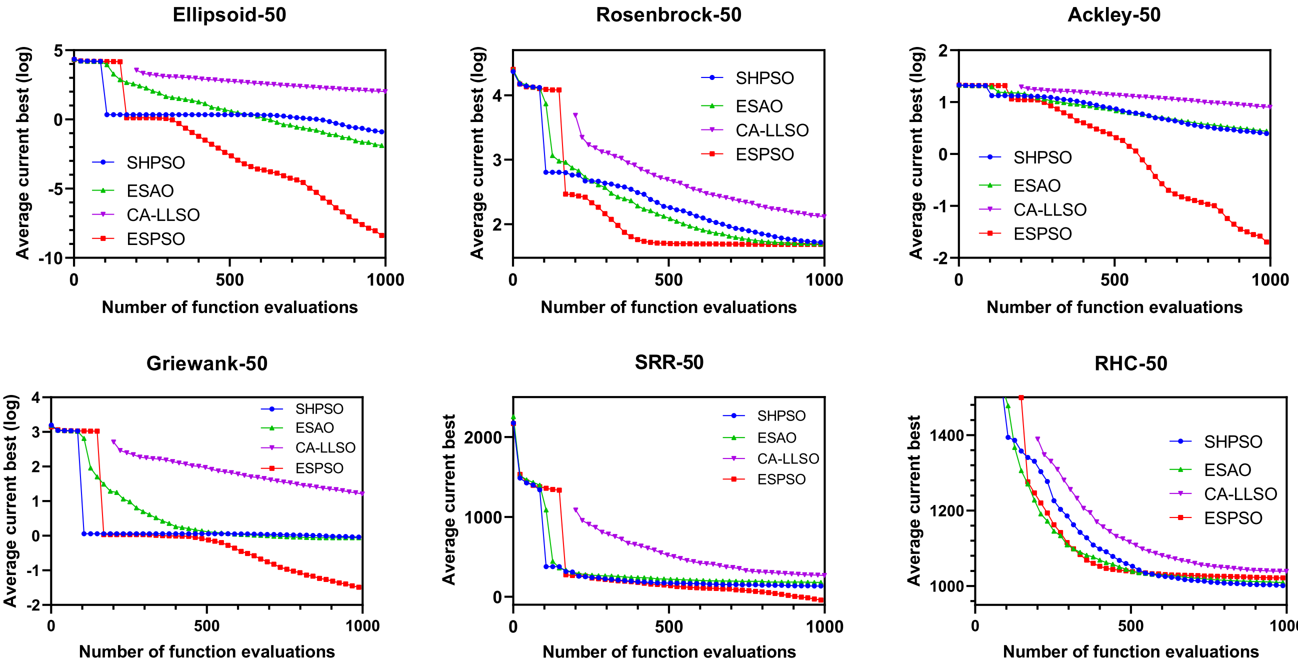

Fig. 3 Convergence curves of different algorithms for the 50D benchmark problems

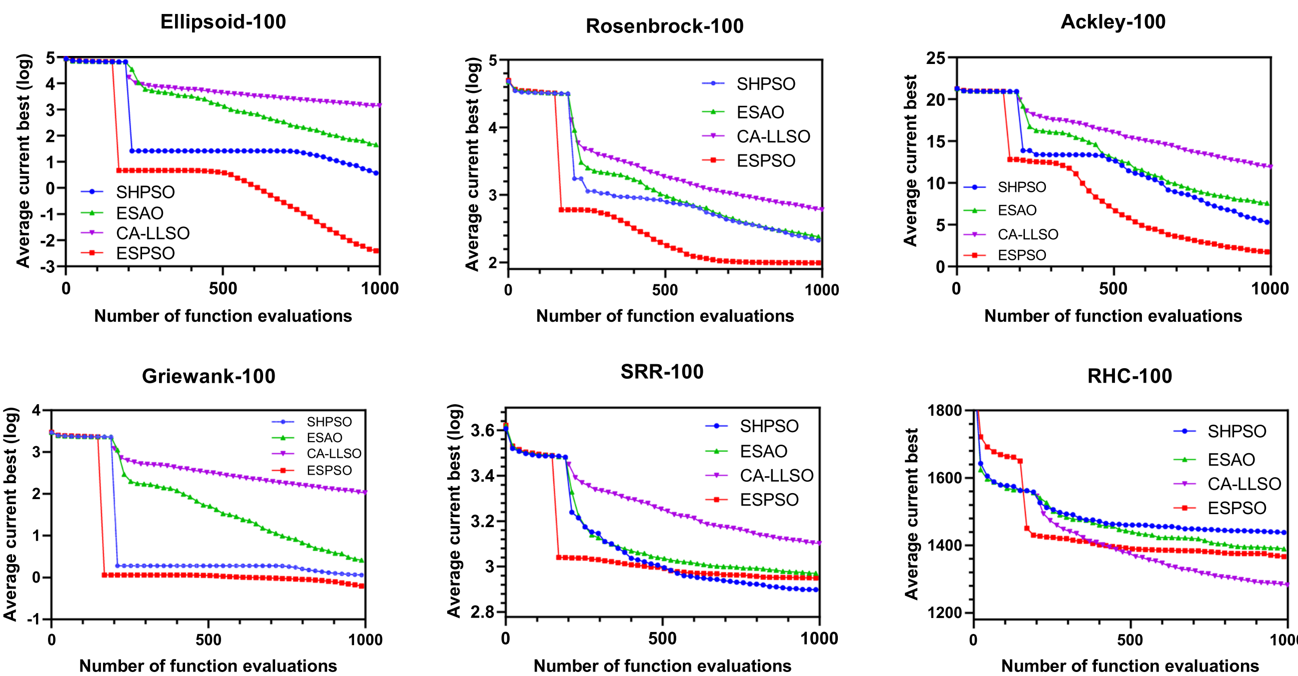

Fig. 4 Convergence curves of different algorithms for the 100D benchmark problems

Figs. 2, 3 and 4, which includes ESPSO and the other three comparison algorithms. From the figures, we can see that ESPSO can converge quickly and obtain better results under the limited fitness value evaluation. On most problems, ESPSO converges faster than other algorithms, especially on Ellipsoid, Rosenbrock, Ackley, and Griewank. For test problems of different dimensions, ESPSO has a big advantage. For very complex multimodal optimization problems, ESPSO can also achieve fast convergence in the early stage. It also shows that when the algorithm fitness evaluation budget is less than 500, the advantage of ESPSO's rapid convergence is more obvious. The result verifies that the EA based on evolutionary sampling has an efficient search capability. The

Table 3 Experiment environment and parameters setting

Experiment environment	
Computer	Intel(R) Core(TM) i5-8500 268(3.00 GHz)
Operating system	Windows 10
Matlab	Matlab 2020b
Parameters setting	
NP	80 ($d < 100$), 120 ($d \geq 100$)
m	10
c_1	2
c_2	2
ω	$0.9 - 0.5 * \mathrm{NFE}/\mathrm{NFE}_{\max}$
$h_{\max}$	3

evolutionary sampling-assisted PSO can combine diversity and fast convergence well. ESPSO gets the best performance compared to the other five state-of-the-art SAEAs.

4.3 Behavioral research of the ESPSO

To investigate the behavior of the ESPSO algorithm, we conducted a comprehensive analysis of its components. There are five components of ESPSO behavior, namely: (1) surrogate-assisted PSO screening (s1), which screens candidate solutions for real fitness evaluation, (2) surrogate-assisted local search (s2), which generates elite samples to improve the initial PSO population, (3) surrogate-assisted trust region search (s3), which samples around the optimal individual, (4) surrogate-assisted local search (s4) to enhance the exploitation ability of the population, and (5) crossover operation (s5) to improve the candidate solution with the global best solution in surrogate-assisted local search.

To study the effects of the different components, we compared the benchmark results of ESPSO and its four degenerate variants. Specifically, we compared ESPSO-v1, which consists of PSO + s1, ESPSO-v2, which uses PSO + s1 + s2, ESPSO-v3, which uses PSO + s1 + s2 + s3, and ESPSO-v4, which uses PSO + s1 + s2 + s3 + s4, with ESPSO that uses all five components. We evaluated the performance of these variants on various test problems and conducted a Friedman test to determine the significance of the results. The results of these ESPSO variants are provided in Table 6.

Our findings showed that the variants performed better as more ESPSO components were added to the PSO. Specifically, ESPSO-v1, which only adds surrogate-assisted PSO screening on top of PSO, produced relatively poor results. However, ESPSO-v2, which incorporates an initial surrogate-assisted local search, quickly identifies excellent individuals and significantly improves the algorithm's performance on almost all test problems. Furthermore, ESPSO-v3, which replaces poorly performing particles in the population using RBF-assisted STR, showed significant improvement over ESPSO-v2 in all test problems. ESPSO-v4, which adds a local search based on the surrogate model, produced only slight improvement compared to ESPSO-v3.

Finally, based on ESPSO-v4, we introduced a full crossover operation between the candidate solutions obtained by RBF-assisted local search and the optimal solution, which further improved the algorithm's performance. Overall, our study demonstrates the effectiveness of the ESPSO algorithm and highlights the importance of incorporating surrogate-assisted techniques and evolutionary sampling strategies in solving high-dimensional and expensive optimization problems. In ESPSO, every component plays an active role.

4.4 Discussion

The experimental analysis of ESPSO behavior has demonstrated that the proposed surrogate-based improvement strategies have a positive effect. These strategies have led to improvements in the search efficiency of the algorithm in various aspects, such as initialization, population

Table 4 Brief description of the comparison algorithms

Time	Algorithm	Characteristics
2018	SHPSO	Surrogate assisted screen and local search, PSO
2019	ESAO	Surrogate-assisted evolutionary sampling, DE
2021	SAMSO	Two swarms, PSO, teaching-learning-based optimization
2021	CA-LLSO	Classifier-based surrogate model, level-based learning swarm optimizer
2022	SA-HLWCA	Combine global and local search, water cycle algorithm
2023	ESPSO	Initialize elite sample, two evolutionary sampling strategies, adaptive surrogate model, PSO

Table 5 Comparison with state-of-the-art DDEAs

Problems	*d*	SHPSO	ESAO	SAMSO	CA-LLSO	SA-HLWCA	ESPSO
Ellipsoid	30	2.12E−01 ± 1.52E−01(+)	2.75E−02 ± 6.96E−02(+)	5.30E−03 ± 5.76E−03(+)	3.44E+00 ± 2.36E+00(+)	4.84E−03 ± 6.81E−03(+)	**3.28E−13 ± 1.44E−12**
Ellipsoid	50	4.03E+00 ± 2.06E+00(+)	7.40E−01 ± 5.55E−01(+)	5.13E−01 ± 5.13E+01(+)	5.70E+01 ± 2.43E+01(+)	5.87E−02 ± 2.30E−02(+)	**3.32E−09 ± 5.29E−09**
Ellipsoid	100	7.61E+01 ± 2.14E+01(+)	1.28E+03 ± 1.34E+02(+)	7.21E+01 ± 7.21E+01(+)	1.00E+03 ± 1.62E+02(+)	2.78E+00 ± 4.97E−01(+)	**3.49E−03 ± 8.91E−04**
Rosenbrock	30	2.86E+01 ± 4.04E−01(≈)	**2.50E+01 ± 1.57E+00(−)**	2.83E+01 ± 8.54E−01(≈)	3.45E+01 ± 4.18E+00(+)	2.75E+01 ± 7.56E−01(−)	2.82E+01 ± 6.35E−01
Rosenbrock	50	5.08E+01 ± 3.03E+00(+)	**4.74E+01 ± 1.71E+00(−)**	5.01E+01 ± 7.68E−01(+)	9.54E+01 ± 1.15E+01(+)	4.79E+01 ± 4.67E−01(−)	4.84E+01 ± 5.04E−01
Rosenbrock	100	1.66E+02 ± 2.64E+01(+)	5.79E+02 ± 4.48E+01(+)	2.86E+02 ± 5.25E+01(+)	4.58E+02 ± 4.68E+01(+)	9.94E+01 ± 6.09E−01(+)	**9.88E+01 ± 1.19E+00**
Ackley	30	1.44E+00 ± 7.74E−01(+)	2.52E+00 ± 8.40E−01(+)	6.28E−01 ± 5.42E−01(+)	3.22E+00 ± 6.20E−01(+)	6.31E−01 ± 5.76E−01(+)	**3.85E−04 ± 3.45E−04**
Ackley	50	1.84E+00 ± 5.64E−01(+)	1.43E+00 ± 2.49E−01(+)	1.53E+00 ± 4.36E−01(+)	6.41E+00 ± 5.62E−01(+)	1.06E+00 ± 5.02E−01(+)	**1.34E−02 ± 3.89E−02**
Ackley	100	4.11E+00 ± 5.92E−01(+)	1.04E+01 ± 2.11E−01(+)	6.12E+00 ± 4.09E−01(+)	1.06E+01 ± 4.28E−01(+)	2.22E+00 ± 2.54E−01(+)	**1.73E+00 ± 2.07E−01**
Griewank	30	9.21E−01 ± 8.81E−02(+)	9.53E−01 ± 5.04E−02(+)	5.38E−01 ± 1.44E−01(+)	1.36E+00 ± 2.20E−01(+)	1.98E−01 ± 1.66E−01(+)	**4.97E−03 ± 1.03E−02**
Griewank	50	9.45E−01 ± 6.14E−02(+)	9.40E−01 ± 4.21E−02(+)	6.66E−01 ± 1.07E−01(+)	8.25E+00 ± 1.81E+00(+)	4.14E−01 ± 1.02E−01(+)	**3.18E−02 ± 4.91E−02**
Griewank	100	1.07E+00 ± 2.05E−02(+)	5.73E+01 ± 5.84E+00(+)	1.06E+00 ± 2.64E−02(+)	7.88E+01 ± 1.06E+01(+)	1.06E+00 ± 1.79E−02(+)	**6.06E−01 ± 1.11E−01**
SRR	30	− 9.28E+01 ± 2.25E+01(+)	6.33E+00 ± 2.65E+01(+)	**− 2.39E+02 ± 2.43E+01(−)**	− 1.93E+02 ± 4.05E+01(≈)	− 1.70E+02 ± 6.89E+01(≈)	− 1.87E+02 ± 6.34E+01
SRR	50	1.34E+02 ± 3.23E+01(+)	1.99E+02 ± 4.58E+01(+)	**− 1.69E+02 ± 3.17E+01(−)**	1.97E+02 ± 6.00E+01(+)	1.01E+02 ± 7.72E+01(+)	− 4.81E+01 ± 1.00E+02
SRR	100	8.02E+02 ± 7.23E+01(−)	**7.13E+02 ± 2.65E+01(−)**	7.37E+02 ± 4.20E+01(−)	1.08E+03 ± 6.92E+01(+)	1.06E+03 ± 6.26E+01(+)	8.91E+02 ± 1.06E+02
RHC	30	9.40E+02 ± 9.02E+00(≈)	9.32E+02 ± 8.94E+00(−)	**9.22E+02 ± 3.66E+00(−)**	1.02E+03 ± 2.78E+01(+)	9.25E+02 ± 4.54E+00(−)	9.48E+02 ± 1.76E+01
RHC	50	9.97E+02 ± 2.21E+01(−)	9.75E+02 ± 3.71E+01(−)	9.70E+02 ± 2.92E+01(−)	1.12E+03 ± 3.00E+01(+)	**9.50E+02 ± 7.06E+01(−)**	1.02E+03 ± 2.01E+01
RHC	100	1.42E+03 ± 3.82E+01(+)	1.37E+03 ± 2.75E+01(≈)	1.29E+03 ± 3.34E+01(−)	1.25E+03 ± 1.67E+01(−)	**9.71E+02 ± 9.17E+01(−)**	1.37E+03 ± 3.47E+01
+/≈/−		14/2/2	12/1/5	11/1/6	16/1/1	12/1/5	NA
Average ranking		4.333	4.028	2.75	5.389	2.306	**2.p194**
p-value		0.0006	0.0033	0.3730	0.0000	0.8586	NA

The best results among the compared methods are highlighted in bold

Table 6 Evolutionary sampling particle swarm optimizer behavioral research

Problems	*d*	ESPSO-v1	ESPSO-v2	ESPSO-v3	ESPSO-v4	ESPSO
Ellipsoid	30	4.10E+00 ± 9.10E+00	1.57E−02 ± 2.84E−02	5.84E−05 ± 4.10E−05	1.41E−09 ± 5.86E−10	**3.28E−13 ± 1.44E−12**
Ellipsoid	50	2.89E+02 ± 2.02E+02	2.37E−01 ± 1.15E−01	9.98E−04 ± 4.86E−04	4.22E−05 ± 1.85E−05	**3.32E−09 ± 5.29E−09**
Ellipsoid	100	6.86E+03 ± 1.87E+03	1.84E+00 ± 7.46E−01	5.18E−01 ± 2.71E−01	1.03E+00 ± 5.06E−01	**3.49E−03 ± 8.91E−04**
Rosenbrock	30	9.55E+01 ± 1.16E+02	5.62E+01 ± 4.14E+01	2.85E+01 ± 4.66E−01	2.83E+01 ± 4.82E−01	**2.82E+01 ± 6.35E−01**
Rosenbrock	50	3.70E+02 ± 2.00E+02	1.17E+02 ± 4.00E+01	4.86E+01 ± 4.54E−01	4.84E+01 ± 4.61E−01	**4.84E+01 ± 5.04E−01**
Rosenbrock	100	2.09E+03 ± 7.89E+02	3.53E+02 ± 1.21E+02	1.08E+02 ± 5.59E+00	1.09E+02 ± 4.76E+00	**9.88E+01 ± 1.19E+00**
Ackley	30	8.66E+00 ± 3.29E+00	7.73E+00 ± 1.61E+00	2.60E−03 ± 5.71E−04	7.86E−04 ± 2.22E−04	**3.85E−04 ± 3.45E−04**
Ackley	50	1.17E+01 ± 2.13E+00	9.42E+00 ± 1.89E+00	1.14E−01 ± 9.50E−02	7.87E−02 ± 1.28E−01	**1.34E−02 ± 3.89E−02**
Ackley	100	1.55E+01 ± 1.12E+00	1.10E+01 ± 1.09E+00	3.53E+00 ± 3.82E−01	3.76E+00 ± 3.00E−01	**1.73E+00 ± 2.07E−01**
Griewank	30	1.19E+00 ± 2.44E−01	7.94E−01 ± 3.07E−01	8.78E−03 ± 2.70E−02	**1.98E−03 ± 3.59E−03**	4.97E−03 ± 1.03E−02
Griewank	50	1.98E+01 ± 8.74E+00	9.26E−01 ± 1.24E−01	8.42E−02 ± 9.68E−02	**2.14E−02 ± 7.20E−03**	3.18E−02 ± 4.91E−02
Griewank	100	2.38E+02 ± 4.16E+01	1.04E+00 ± 3.98E−02	8.90E−01 ± 7.93E−02	9.57E−01 ± 8.63E−02	**6.06E−01 ± 1.11E−01**
SSR	30	− 5.80E+01 ± 4.02E+01	− 8.18E+01 ± 4.36E+01	− 1.47E+02 ± 3.59E+01	− 1.45E+02 ± 3.17E+01	**− 1.87E+02 ± 6.34E+01**
SSR	50	1.93E+02 ± 3.40E+01	1.52E+02 ± 5.00E+01	3.52E+01 ± 5.56E+01	4.71E+01 ± 5.78E+01	**− 4.81E+01 ± 1.00E+02**
SSR	100	1.26E+03 ± 1.96E+02	9.32E+02 ± 8.72E+01	9.11E+02 ± 7.87E+01	**8.76E+02 ± 1.07E+02**	8.91E+02 ± 1.06E+02
RHC	30	9.64E+02 ± 1.46E+01	9.63E+02 ± 1.79E+01	**9.36E+02 ± 8.07E+00**	9.41E+02 ± 1.27E+01	9.48E+02 ± 1.76E+01
RHC	50	1.06E+03 ± 3.84E+01	1.03E+03 ± 3.44E+01	**1.01E+03 ± 2.84E+01**	1.02E+03 ± 3.29E+01	1.02E+03 ± 2.01E+01
RHC	100	1.45E+03 ± 3.50E+01	1.40E+03 ± 5.11E+01	1.38E+03 ± 6.43E+01	1.37E+03 ± 3.37E+01	**1.37E+03 ± 3.47E+01**
Average ranking		5.000	4.000	2.444	2.139	1.417
p-value		0.000	0.000	0.051	0.171	NA

The best results among the compared methods are highlighted in bold update, and optimal solution promotion. ESPSO method combines the advantages of both particle swarm optimization and evolutionary sampling strategies, balancing exploration and exploitation. In comparison with other state-of-the-art surrogate-assisted evolutionary algorithms, it exhibits fast convergence and performs well in solving expensive optimization problems, shown in Table 5 and convergence curves comparison. Meanwhile, we build an RBF model based on personal historical optimal data on the population to focus on the search region. The method uses a small, fixed amount of data to model, which has enhanced computing efficiency. However, there is still some issue for further improvement, particularly in the case of very complex multimodal functional landscapes, which may lead to premature convergence, as observed in the RHC problem. Therefore, additional research is required to address these limitations and enhance the performance of ESPSO in challenging optimization scenarios.

5 Conclusion

In the realm of optimization, high-dimensional and expensive problems pose significant challenges. To address these challenges, we propose a novel algorithm that combines an evolutionary sampling strategy with the PSO framework. Specifically, we introduce the evolutionary sampling particle swarm optimization algorithm, which builds an adaptive RBF model based on the personal historical optimum data of the population. The model enables the algorithm to focus on the current search area as the population converges. On the other hand, the algorithm utilizes surrogate-assisted evolutionary sampling strategies to improve the initialed population, and the evolutionary sampling strategies drive the population to converge rapidly by sampling candidate solutions to replace poorly performing particles in the population. Combining the above strategies, ESPSO shows excellent performance and comparative experiments also demonstrated it. Moreover, component analysis verified the effectiveness of the proposed improvement strategies. In the future, designing new evolutionary sampling strategies is a meaningful direction, and we also can investigate combining the evolutionary sampling strategies with other population-based algorithms to achieve even better optimization results. Moreover, applying evolutionary sampling strategies to solve expensive multi-objective optimization problems is a promising direction.

Acknowledgements This work was partly supported by the National Natural Science Foundation of China under Grant No. 62076225, and the Natural Science Foundation for Distinguished Young Scholars of Hubei under Grant No. 2019CFA081.

Data availibility The data generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Xue Y, Xue B, Zhang M (2019) Self-adaptive particle swarm optimization for large-scale feature selection in classification. ACM Trans Knowl Discov Data 13(5):1–27
2. Wang F, Zhang H, Zhou A (2021) A particle swarm optimization algorithm for mixed-variable optimization problems. Swarm Evol Comput 60:100808
3. Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
4. Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–73
5. Ayyarao TS, Ramakrishna N, Elavarasan RM, Polumahanthi N, Rambabu M, Saini G, Khan B, Alatas B (2022) War strategy optimization algorithm: a new effective metaheuristic algorithm for global optimization. IEEE Access 10:25073–25105
6. Xue Y, Wang Y, Liang J, Slowik A (2021) A self-adaptive mutation neural architecture search algorithm based on blocks. IEEE Comput Intell Mag 16(3):67–78
7. Chugh T, Chakraborti N, Sindhya K, Jin Y (2017) A data-driven surrogate-assisted evolutionary algorithm applied to a many-objective blast furnace optimization problem. Mater Manuf Process 32(10):1172–1178
8. Manca AG, Pappalardo CM (2020) Topology optimization procedure of aircraft mechanical components based on computer-aided design, multibody dynamics, and finite element analysis. In: Design, simulation, manufacturing: the innovation exchange, Springer, pp 159–168
9. Park J, Sandberg IW (1991) Universal approximation using radial-basis-function networks. Neural Comput 3(2):246–257
10. Zhen H, Gong W, Wang L, Ming F, Liao Z (2021) Two-stage data-driven evolutionary optimization for high-dimensional expensive problems. IEEE Trans Cybern. https://doi.org/10.1109/TCYB.2021.3118783
11. Zhen H, Xiong S, Gong W, Wang L (2023) Neighborhood evolutionary sampling with dynamic repulsion for expensive multimodal optimization. Inf Sci 630:82–97. https://doi.org/10.1016/j.ins.2023.02.049
12. Wang H, Jin Y (2020) A random forest-assisted evolutionary algorithm for data-driven constrained multiobjective combinatorial optimization of trauma systems. IEEE Trans Cybern 50(2):536–549
13. Kleijnen JP (2009) Kriging metamodeling in simulation: a review. Eur J Oper Res 192(3):707–716
14. Poloczek J, Kramer O (2013) Local svm constraint surrogate models for self-adaptive evolution strategies. In: Timm IJ, Thimm M (eds) KI 2013: advances in artificial intelligence. Springer, Berlin, Heidelberg, pp 164–175
15. Liu B, Zhang Q, Gielen GG (2013) A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expen-

sive optimization problems. IEEE Trans Evol Comput 18(2):180–192
16. Yi J, Shen Y, Shoemaker CA (2020) A multi-fidelity rbf surrogate-based optimization framework for computationally expensive multi-modal problems with application to capacity planning of manufacturing systems. Struct Multidiscip Optim 62(4):1787–1807
17. Nguyen BH, Xue B, Zhang M (2022) A constrained competitive swarm optimiser with an svm-based surrogate model for feature selection. IEEE Trans Evol Comput
18. Wang H, Jin Y, Doherty J (2017) Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems. IEEE Trans Cybern 47(9):2664–2677
19. Zhen H, Gong W, Wang L (2022) Offline data-driven evolutionary optimization based on model selection. Swarm Evol Comput 71:101080
20. Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. J Glob Optim 13(4):455–492
21. Jiao R, Zeng S, Li C, Jiang Y, Jin Y (2019) A complete expected improvement criterion for gaussian process assisted highly constrained expensive optimization. Inf Sci 471:80–96
22. Emmerich MT, Giannakoglou KC, Naujoks B (2006) Single-and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. IEEE Trans Evol Comput 10(4):421–439
23. Zhang Q, Liu W, Tsang E, Virginas B (2009) Expensive multiobjective optimization by moea/d with Gaussian process model. IEEE Trans Evol Comput 14(3):456–474
24. Xue Y, Tang Y, Xu X, Liang J, Neri F (2022) Multi-objective feature selection with missing data in classification. IEEE Trans Emerg Top Comput Intell 6(2):355–364
25. Li F, Gao L, Garg A, Shen W, Huang S (2021) A comparative study of pre-screening strategies within a surrogate-assisted multi-objective algorithm framework for computationally expensive problems. Neural Comput Appl 33(9):4387–4416
26. Liu Y, Liu J, Tan S, Yang Y, Li F (2022) A bagging-based surrogate-assisted evolutionary algorithm for expensive multi-objective optimization. Neural Comput Appl 34(14):12097–12118
27. Zhen H, Gong W, Ling W (2021) Data-driven evolutionary sampling optimization for expensive problems. J Syst Eng Electron 32(2):318–330
28. Wang X, Wang GG, Song B, Wang P, Wang Y (2019) A novel evolutionary sampling assisted optimization method for high-dimensional expensive problems. IEEE Trans Evol Comput 23(5):815–827
29. Tian J, Tan Y, Zeng J, Sun C, Jin Y (2018) Multiobjective infill criterion driven gaussian process-assisted particle swarm optimization of high-dimensional expensive problems. IEEE Trans Evol Comput 23(3):459–472
30. Yu H, Tan Y, Zeng J, Sun C, Jin Y (2018) Surrogate-assisted hierarchical particle swarm optimization. Inf Sci 454:59–72
31. Cai X, Gao L, Li X (2020) Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems. IEEE Trans Evol Comput 24(2):365–379
32. Zhen H, Gong W, Wang L (2022) Evolutionary sampling agent for expensive problems. IEEE Trans Evol Comput. https://doi.org/10.1109/TEVC.2022.3177605
33. Akyol S, Alatas B (2017) Plant intelligence based metaheuristic optimization algorithms. Artif Intell Rev 47:417–462
34. Alatas B, Bingol H (2020) Comparative assessment of light-based intelligent search and optimization algorithms. Light Eng 28(6)
35. Dong H, Dong Z (2020) Surrogate-assisted grey wolf optimization for high-dimensional, computationally expensive black-box problems. Swarm Evol Comput 57:100713
36. Yu H, Kang L, Tan Y, Zeng J, Sun C (2021) A multi-model assisted differential evolution algorithm for computationally expensive optimization problems. Complex Intell Syst 7(5):2347–2371
37. Nguyen HB, Xue B, Andreae P (2018) Pso with surrogate models for feature selection: static and dynamic clustering-based methods. Memet Comput 10(3):291–300
38. Hao H, Zhou A, Qian H, Zhang H (2022) Expensive multiobjective optimization by relation learning and prediction. IEEE Trans Evol Comput 26:1157
39. Pan L, He C, Tian Y, Wang H, Zhang X, Jin Y (2018) A classification-based surrogate-assisted evolutionary algorithm for expensive many-objective optimization. IEEE Trans Evol Comput 23(1):74–88
40. Chugh T, Jin Y, Miettinen K, Hakanen J, Sindhya K (2016) A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. IEEE Trans Evol Comput 22(1):129–142
41. Wang Y, Lin J, Liu J, Sun G, Pang T (2022) Surrogate-assisted differential evolution with region division for expensive optimization problems with discontinuous responses. IEEE Trans Evol Comput 26(4):780–792. https://doi.org/10.1109/TEVC.2021.3117990
42. Tian J, Tan Y, Zeng J, Sun C, Jin Y (2018) Multiobjective infill criterion driven gaussian process-assisted particle swarm optimization of high-dimensional expensive problems. IEEE Trans Evol Comput 23(3):459–472
43. Cheng R, Jin Y (2015) A social learning particle swarm optimization algorithm for scalable optimization. Inf Sci 291(C):43–60
44. Sun C, Jin Y, Cheng R, Ding J, Zeng J (2017) Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems. IEEE Trans Evol Comput 21(4):644–660
45. Li F, Cai X, Gao L, Shen W (2021) A surrogate-assisted multiswarm optimization algorithm for high-dimensional computationally expensive problems. IEEE Trans Cybern 51(3):1390–1402. https://doi.org/10.1109/TCYB.2020.2967553
46. Chen G, Li Y, Zhang K, Xue X, Wang J, Luo Q, Yao C, Yao J (2021) Efficient hierarchical surrogate-assisted differential evolution for high-dimensional expensive optimization. Inf Sci 542:228–246
47. Meyer-Baese A, Schmid V (2014) Chapter 7 - foundations of neural networks. In: Meyer-Baese A, Schmid V (eds) Pattern recognition and signal analysis in medical imaging, 2nd edn. Academic Press, Oxford, pp 197–243
48. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international Conference on Neural Networks, vol 4, pp 1942–1948 IEEE
49. Wang F, Wang X, Sun S (2022) A reinforcement learning level-based particle swarm optimization algorithm for large-scale optimization. Inf Sci 602:298–312
50. Bansal JC, Singh PK, Saraswat M, Verma A, Jadon SS, Abraham A (2011) Inertia weight strategies in particle swarm optimization. In: 2011 Third World Congress on Nature and Biologically Inspired Computing, pp 633–640 https://doi.org/10.1109/NaBIC.2011.6089659

51. Helton JC, Davis FJ (2003) Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. Reliabil Eng Syst Saf 81(1):23–69
52. Suganthan PN, Hansen N, Liang JJ, Deb K, Tiwari S (2005) Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. Nat Comput 341–357
53. Wei F-F, Chen W-N, Yang Q, Deng J, Luo X-N, Jin H, Zhang J (2021) A classifier-assisted level-based learning swarm optimizer for expensive optimization. IEEE Trans Evol Comput 25(2):219–233. https://doi.org/10.1109/TEVC.2020.3017865
54. Chen C, Wang X, Dong H, Wang P (2022) Surrogate-assisted hierarchical learning water cycle algorithm for high-dimensional expensive optimization. Swarm Evol Comput 75:101169