Semi-supervised Learning Assisted Particle Swarm Optimization of Computationally Expensive Problems

Chaoli Sun  
Department of Computer Science and Technology, Taiyuan University of Science and Technology  
Taiyuan, China  
State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University  
Shenyang, China  
chaoli.sun.cn@gmail.com

Yaochu Jin∗  
Department of Computer Science, University of Surrey  
Guildford, United Kingdom  
Department of Computer Science and Technology, Taiyuan University of Science and Technology  
Taiyuan, China  
yaouchu.jin@surrey.ac.uk

Ying Tan  
Department of Computer Science and Technology, Taiyuan University of Science and Technology  
Taiyuan, China  
tanying1965@gmail.com

ABSTRACT
In many real-world optimization problems, it is very time-consuming to evaluate the performance of candidate solutions because the evaluations involve computationally intensive numerical simulations or costly physical experiments. Therefore, standard population based meta-heuristic search algorithms are not best suited for solving such expensive problems because they typically require a large number of performance evaluations. To address this issue, many surrogate-assisted meta-heuristic algorithms have been proposed and shown to be promising in achieving acceptable solutions with a small computation budget. While most research focuses on reducing the required number of expensive fitness evaluations, not much attention has been paid to take advantage of the large amount of unlabelled data, i.e., the solutions that have not been evaluated using the expensive fitness functions, generated during the optimization. This paper aims to make use of semi-supervised learning techniques that are able to enhance the training of surrogate models using the unlabelled data together with the labelled data in a surrogate-assisted particle swarm optimization algorithm. Empirical studies on five 30-dimensional benchmark problems show that the proposed algorithm is able to find high-quality solutions for computationally expensive problems on a limited computational budget.

KEYWORDS
Particle swarm optimization, semi-supervised learning, computationally expensive problems

∗Corresponding author

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

ACM Reference Format:

1 INTRODUCTION
In solving many real-world engineering and science problems, performance evaluations of candidate solutions may be computationally very intensive due to the utilization of high-fidelity numerical analysis techniques. For example, a single simulation to evaluate the performance of a design of a high-frequency integrated circuit needs approximately 10 ~ 15 minutes, and a typical Terahertz computational electromagnetic simulation may take 20 ~ 30 minutes [26]. Consequently, most meta-heuristic optimization algorithms, such as genetic algorithms, particle swarm optimization and differential evolution algorithms cannot be directly applied to solving this class of problems, which are known as computationally expensive problems, mainly because these algorithms require a large number of performance evaluations to locate an optimal solution. Surrogate assisted meta-heuristic algorithms, in which computationally cheap surrogate models are employed to replace the time-consuming exact objective evaluations, have received increasing attention in recent years [20]. Commonly used surrogate models include polynomial regression [30, 37], neural networks [18, 21], support vector machines [2], radial basis function networks (RBF) [31], and Kriging models [7] (also known as Gaussian process).

Different machine learning techniques have been employed to develop model management strategies in surrogate-assisted single or multi-objective meta-heuristic algorithms. Jin et al. [21] employed an artificial neural network assisted covariance matrix adaptation evolution strategy for airfoil optimization, where the neural network is trained by error backpropagation using data weighted by the covariance matrix. Neural network assisted evolutionary aerodynamic optimization was investigated in [18]. To improve the performance of the neural network, its structure was optimized with respect to common problem classes using averaged Lamarkian inheritance. While most surrogates aim to approximate the objective functions, surrogates are also used as a classifier to distinguish
non-dominated solutions from dominated ones [1, 28, 35], select better solutions [27], or to separate feasible solutions from non-feasible ones [16]. Another machine learning concept that has been used widely in surrogate management is active learning, which has been used either empirically [23, 34] or formally in infill criteria [12, 17, 26, 29]. Since data paucity is a huge challenge in solving expensive optimization problems, it is natural to take advantage of data or knowledge from similar optimization problems to speed up evolution [9, 14, 19] by borrowing ideas from transfer learning or multi-task learning. Finally, multiple surrogates consisting of a global and a local models [25, 32, 36], where the main idea is to use the global surrogates to smoothen out the local optima and the local surrogates to capture the details of the interested local fitness landscape.

Despite the success of various surrogate techniques reported in the literature, most surrogate modeling techniques rely only on a small number of solutions whose fitness value is evaluated using the computationally expensive objective functions. In surrogate-assisted optimization algorithms, however, a majority of the solutions are not evaluated using the expensive objective functions and a large amount of unlabeled are not exploited. As shown by a large body of research on semi-supervised machine learning [4], the unevaluated solutions (unlabeled data) may considerably contribute to the enhancement of the learning performance, in particular when there is a lack of labeled data, thereby improving the performance of surrogate-assisted optimization. To the best of our knowledge, the only work on semi-supervised learning that makes use of semi-supervised learning was reported by Sun et al. [33], which adopts the co-training technique [3, 38] to enhance the accuracy of RBF-based surrogates assisting an interactive genetic algorithms to alleviate the user fatigue.

In this paper, we propose a model management strategy inspired by semi-supervised learning to choose unlabelled data, i.e., the solutions that have not been evaluated using the time-consuming exact objective function, to train an RBF surrogate model together with those labelled data. Similar to the co-training technique, we employ two surrogate models, one trained using labelled data only while the other trained using both labelled and unlabelled data. An empirical criterion is suggested to determine which individuals should be evaluated using the exact objective function according to the fitness values estimated by the two models, together with a criterion for selecting unlabelled data for training the surrogates.

The remainder of the paper is organized as follows. Section 2 provides a brief review of the background, including a modified particle swarm optimization, semi-supervised learning, and radial basis function networks. In Section 3, the details of the proposed approach are presented. Experimental results on five 30-dimensional test problems are given and discussed in Section 4. Finally, Section 5 concludes the paper with a summary and some ideas for future work.

2 BACKGROUND

2.1 A Modified Particle Swarm Optimization Algorithm

Without loss of generality, we consider the following single-objective optimization problem:

\[
\begin{align*}
\text{minimize:} \ f(x) \\
\text{subject to:} \ x_l \leq x \leq u
\end{align*}
\]

where \( x = (x_1, x_2, \ldots , x_D) \in \mathbb{R}^D \) is a vector of continuous decision variables, \( D \) is the dimension of the search space. \( f(x) \) is a scalar objective function, \( x_l \) and \( u \) are vectors of the lower and upper bounds of search space, respectively.

Particle swarm optimization (PSO), which simulates behaviors of social animals such as bird flocking or fish schooling, was proposed by Kennedy and Eberhart in 1995 [11]. After that, many particle swarm optimization variations have been proposed for either speeding up the convergence [8] or improving the diversity of the population to prevent the search from getting stuck in a local optimum [5, 24]. The social learning particle swarm optimization (SL-PSO) is a PSO variant proposed by Cheng and Jin [6] that does not reply the global best and personal best positions. In SL-PSO, the position of particle \( i \) will be updated as follows:

\[
x_{jd}(t+1) = \begin{cases} 
  x_{jd}(t) + \Delta x_{jd}(t+1) & \text{if } pr_j(t) \leq pr_j^l \\
  x_{jd}(t) & \text{otherwise}
\end{cases}
\]

with

\[
\Delta x_{jd}(t+1) = r_1 \cdot \Delta x_{jd}(t) + r_2 \cdot (x_{jd}(t) - x_{jd}(t)) + r_3 \cdot \epsilon \cdot (\tilde{x}_d(t) - x_{jd}(t))
\]

where \( pr_j, 0 \leq pr_j \leq 1 \), is a randomly generated probability and \( pr_j^l \) is the probability threshold for particle \( j \) to update its position, \( r_1, r_2 \) and \( r_3 \) are three random numbers uniformly generated in the range \([0, 1]\). \( x_{kd}(t) \) represents the d-th (1 \leq d \leq D) element of particle \( k \) whose fitness is better than \( f(x_j) \), \( \tilde{x}_d(t) = \frac{\sum_{i=1}^{n} x_{jd}(t)}{n} \) is the mean position value on d-th dimension of the swarm, \( \epsilon \) is a parameter called the social influence factor that controls the influence of \( \tilde{x}_d(t) \).

The SL-PSO algorithm can obtain better results on large scale optimization problems, however, the speed of converge is slow especially at the beginning of the search, which is a drawback when the allowed number of fitness evaluations is limited. Therefore, in our work, we propose that particles learn from a randomly selected personal best position whose fitness is not worse than their own personal best fitness. Therefore, Eq. (3) will be modified to be:

\[
\Delta x_{jd}(t+1) = r_1 \cdot \Delta x_{jd}(t) + r_2 \cdot (p_{jd}(t) - x_{jd}(t)) + r_3 \cdot \epsilon \cdot (\tilde{x}_d(t) - x_{jd}(t))
\]

This way, the modified PSO is able to achieve faster convergence while still preserving an adequate degree of diversity in the beginning of the search.

2.2 Semi-supervised Learning

Semi-supervised learning (SSL) aims to utilize both labeled and unlabeled data to learn a mapping from a training data set made of pairs \((x_i, f((x_i)))\). The data set \( X = \{x_1, x_2, \ldots , x_n\} \) is composed of two parts: \( X_l = \{x_{i1}, x_{i2}, \ldots , x_{in}\} \).
In this work, we use the following cubic spline function:

\[ f(x) \] are given, and \( X_{ul} = \{ x_{u_1}, x_{u_2}, \ldots, x_{u_n} \} \)

whose labels \( F = \{ f(x_{u_1}), f(x_{u_2}), \ldots, f(x_{u_n}) \} \) are unknown, where \( n \) is the total number of data in the data set, \( l_j, j = 1, 2, \ldots, l \) and \( u_k, k = 1, 2, \ldots, u \) represents the \( j \)-th labeled and \( k \)-th unlabeled data, respectively. \( l_j \) is the total number of labeled data and \( u_k \) is the total number of unlabeled data, thus \( l_j + u_k = n \). The main goal of semi-supervised learning is to train a model by taking into account the unlabeled data to achieve better prediction of unseen data than models trained using labeled data only [4].

One central issue in semi-supervised learning is to determine which unlabeled data should be added into the training dataset to help improve the learning performance. Many semi-supervised learning methods have been suggested, including Expectation Maximization with generative mixture models, self-training, co-training, transductive support vector machines, and graph-based methods. The reader can refer to [4] for more details.

### 2.3 Radial Basis Function Network

RBF networks are one class of commonly utilized surrogate models. Empirical results have shown that the approximation performance of RBF networks scales well to the increase in the dimension of the function to be approximated [13, 22]. In this work, the RBF network is employed as a global surrogate model to assist the modified PSO to search for optimal solutions of computationally expensive problems.

An RBF network is a real-valued function \( \phi: \mathbb{R}^D \to \mathbb{R} \). Let \( D = \{ (x_i, f(x_i)), i = 1, 2, \ldots, N \} \) denote the dataset for training the RBF network, where \( x_i \in \mathbb{R}^D \) and \( f(x_i) \in \mathbb{R} \) are the inputs and output, respectively, \( N \) is the number of training data. Various radial basis functions [10, 15], including Gaussian, splines and multiquadrics can be used as the basis function of the RBF networks, and in this work, we use the following cubic spline function:

\[
\phi(x) = (||x - x'||)^3
\]

Consequently the surrogate model can be written in the following form:

\[
\Phi(x) = \sum_{i=1}^{ND} \beta_i(||x - x_i||)^3 + Q(x)
\]

where \( || \cdot || \) denotes the Euclidean norm in \( \mathbb{R}^D \), the coefficient \( \beta_i \) is a real number, \( ND \) is the number of training data. \( Q(x) \) is in the linear space \( \prod_m \) of polynomials of degree at most \( m \) in \( \mathbb{R}^D \). The polynomial is given by the following general form where \( m \) is the polynomial degree, \( a \) is a vector of real coefficients and \( p_k(x) \) are the monomial components:

\[
Q(x) = \sum_{k=1}^{m} \alpha_k p_k(x)
\]

In this work, \( Q(x) = a_1 + \sum_{k=1}^{D} x_k \) is adopted as the polynomial form. All parameters \( a_k, k = 1, 2, \ldots, m \) and \( \beta_i, i = 1, 2, \ldots, ND \) are completely determined by the interpolation condition and the positive definition condition for the basis function matrix.

### 3 SEMI-SUPERVISED LEARNING ASSISTED PSO

In this work, we call solutions that are evaluated using the computationally expensive objective function labelled data and those are not labelled data. Thus, only a limited amount of labelled data will be available but a large amount of unlabelled data can be generated. Therefore, in this work, we aim to utilize some unlabelled data together with the limited labelled to train the RBF network based surrogate model. Two key questions must be answered in semi-supervised learning assisted particle swarm optimization, i.e.,

1. which particles should be evaluated using the expensive objective function, and
2. which ones evaluated using the surrogate can be included in the data set for training the surrogate.

In the following, we start with introducing the general framework of the proposed semi-supervised learning assisted PSO algorithm, called SSL-assisted PSO, followed by a detailed description of a model management strategy.

### 3.1 General Framework of Semi-Supervised Learning Assisted PSO

Fig. 1 presents a general framework of the proposed SSL-assisted PSO. In the, \( DB \) is an archive where all labelled data are stored, while in \( EDB \) all labelled data and some selected unlabelled data are stored. \( M_1 \) and \( M_2 \) are two RBF models trained using the data in \( DB \) and \( EDB \), respectively. Both models are used to approximate the fitness of all particles in the current population and some of which will be re-evaluated using the expensive fitness function. The particles in the current population that have been evaluated using the expensive objective function will then be used to select some particles in the current population evaluated using the surrogate to be stored in \( EDB \). Finally, the global best position will be updated if the best of the personal best positions is better than the current global best. Make sure that this personal best position is evaluated using the expensive objective function. Otherwise the particle that has the best approximated fitness value will be re-evaluated using the expensive function and the global best and personal best will be updated if the real fitness is indeed better. All particles evaluated using the expensive objective function will be stored in both \( DB \) and \( EDB \).

It should be noted that we also use the surrogate model trained using the labelled data only to assist the PSO. The reason for using both \( M_1 \) and \( M_2 \) is to develop a surrogate management strategy that can select a reasonable number of solutions to be evaluated using the expensive objective function. If we rely on \( M_2 \) only, usually we need to evaluate any particle if its fitness value according to \( M_2 \) is better than its personal best. This strategy, according to our pilot studies, will often lead to too many expensive fitness evaluations, making it hard for the algorithm to find a good solution on a limited computational budget. Alternatively, if we select particles whose fitness value is better than the global best solution for fitness evaluation using the expensive objective function, very small number of solutions will be selected and the algorithm is very likely to get stuck. Refer to the following subsection for more details about the management strategy making use of both \( M_1 \) and \( M_2 \).
The first step in model management is to determine which particles and EDB optimum introduced by the surrogates. Reduces the possibility of the algorithm getting stuck in a local evaluation has been made in the current generation. This mechanism using the real objective function, when no expensive fitness eval-
best fitness approximated by the surrogates must be re-evaluated
objective function in updating the global best position. Algorithm 1
be approximated at first using two surrogates, the second generation onward, all particles in the population will
particles in the initial population will be evaluated using the real
DB objective function. From Fig. 1, we can see that data in
in the archives; these will be set to \( \mathcal{M}_1 \).

Finally, some solutions may also be evaluated using the real objective function in updating the global best position. Algorithm 1 gives the pseudo code of the process to update the global best position. Note that in the proposed algorithm, the particle having the best fitness approximated by the surrogates must be re-evaluated using the real objective function, when no expensive fitness evaluation has been made in the current generation. This mechanism reduces the possibility of the algorithm getting stuck in a local optimum introduced by the surrogates.

### Algorithm 1: Update of the global best position

1. Find the particle \( k \) that has the best personal best fitness;
2. if \( f(pbest_k) < f(gbest) \) then
3. if \( f(pbest_k) \) is approximated using the surrogate model then
4. Re-evaluate the fitness of solution \( pbest_k \) and store them
5. in the archives;
6. Compare \( f(pbest_k) \) and \( f(gbest) \) again and update the global
7. best position;
8. if no solution has been evaluated using the real objective function then
9. Find the individual \( j \) that has the best fitness value;
10. Re-evaluate the fitness of solution \( x_j \) and store it in the
11. archives;
12. Update the personal best position of particle \( j \);
13. Compare \( f(pbest_j) \) and \( f(gbest) \) again and update the
14. global best position;
15. end if
end if
16. end if

### 3.3 Selecting unlabelled data

In semi-supervised learning, it is critical to select appropriate unlabelled data (solutions evaluated by the surrogate) to be added into the labelled data. Fig. 2 provides an illustrative example to explain which unlabelled data might be helpful if they are included in the training dataset. In the figure, the solid line denotes the real objective function and the dashed line the approximated objective function before any new unlabelled data is added in training (represented by \( f_{M_2}(\cdot) \)). Assume \( x_1 \) a solution in the current generation that is selected to be re-evaluated using the real fitness function, and its real fitness value is denoted by a square. \( x_2 \) is a solution in the current generation and its approximated value is denoted by a dot. Now assume \( x_2 \) is the selected unlabelled data to be added in the training dataset and the updated surrogate is denoted by the dot dashed line (represented by \( f_{M_2}(\cdot) \)). In the upper panel of Fig. 2, we can see that adding solutions \( x_2 \) into the training data set reduces the approximation error of surrogate \( f_{M_2}(x_1) \), thus it is beneficial to include \( x_2 \) in the training dataset. In the lower panel of the figure, however, including \( x_2 \) in the training dataset will increase the approximation error of \( f_{M_2}(x_1) \). Therefore, \( x_2 \) should not be added in the training dataset.

The above idea for selecting the unlabelled data is implemented in Algorithm 2. For each individual whose fitness value is estimated by surrogate \( M_2 \), we temporarily add it to the archive \( EDB \) and then update \( M_2 \). Then we calculate the change in the approximation error of all solutions in the current population that have been evaluated using the real objective function and find out the minimum error reduction (the solution that increases the approximation error will be excluded outright). Suppose \( \text{EvalIndi} \) is a set of all individuals whose fitness has been evaluated using the real objective function. If solution \( i \) whose fitness is estimated by \( M_2 \) is added to the training set and \( M_2 \) is updated to be \( M_2' \), then the following difference in approximation errors before and after the surrogate is retrained
using the dataset in which unlabelled data \( i \) is added will be:

\[
\varphi(x_i) = \min_{j \in E \cup \text{Indi}} \left( |f_{M_1}(x_j) - f(x_j)| - |f_{M_2}(x_j) - f(x_j)| \right)
\]  

(8)

Finally, solution \( i \) that achieves the maximum \( \varphi(x_i) \) will be eventually added to the EDB and the surrogate will be updated.

**Algorithm 2** The strategy for selecting unlabeled data

1. for each individual \( i \) with an approximated fitness value do
2. Add it into the training dataset and train an RBF model;
3. Estimate the fitness value of the solutions evaluated using the real objective function in the current population using the RBF model;
4. Calculate the \( \varphi(x_i) \) for each approximated individual using Eq. (8)
5. end for
6. Find the individual having the best \( \varphi(x_i) \) and store it in archive EDB;

4 EXPERIMENTAL RESULTS AND DISCUSSIONS

To examine the performance of the proposed SSL-assisted PSO algorithm, we conduct a set of empirical studies by comparing its performance with a state-of-the-art surrogate-assisted PSO on five widely used uni-modal and multi-modal benchmark problems. The characteristics of these test problems are given in Table 1. Although all these benchmark problems are computationally cheap per se, we suppose that they are computationally expensive problems and only a limited number of fitness evaluations is allowed for performing the optimization. The size of the population of all compared algorithms is set to 50, with the probability threshold \( pr_j^L, j = 1, 2, \ldots, D \) being set to 1 and \( \epsilon \) set to 0.001. 20 independent runs are conducted for each test problem and the optimization is terminated when the maximum number of fitness evaluations are exhausted, which we set to 11 * dimension in this study. The size of the training set for \( M_1 \) is set to 2 * (dimension + 1) and the size of the training set for \( M_2 \) is set to twice of that for \( M_1 \), i.e., 4 * (dimension + 1), since we can obtain more training data from EDB than from DB. Both data that are recently stored in the archives will be chosen to train the surrogates.

Recall that two surrogates are used in the proposed algorithm, of which surrogate \( M_1 \) is trained using labelled data only and model \( M_2 \) is trained using both labelled and unlabelled data. We first compare the proposed algorithm with a variant called SSL-only PSO, in which only model \( M_2 \) is utilized and those particles whose fitness is better than their personal best position will be evaluated using the original objective function. Figures 3~7 plot the convergence profiles of the two algorithms on the five 30-dimensional test problems averaged over 20 runs. From these figures, we can see that SSL-assisted PSO outperforms SSL-only PSO given 11 * dimension expensive fitness evaluations in terms of the final results as well as convergence speed, indicating that the proposed surrogate management strategy making use of two surrogates are effective.

Next, we compare SSL-assisted PSO with two additional variants, one PSO assisted by an RBF network trained by labelled data only, and the other without using any surrogates, as well as CAL-SAPSO, a most recently published surrogate-assisted PSO that is inspired by committee-based active learning [34]. Here, we select CAL-SAPSO for comparison because it is a state-of-the-art PSO assisted by surrogates and has been shown to be able to outperform many existing surrogate-assisted single objective evolutionary algorithms.

<table>
<thead>
<tr>
<th>Benchmark Problem</th>
<th>Description</th>
<th>Characteristics</th>
<th>Global Optimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1 Ellipsoid</td>
<td>Uni-modal</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>F2 Rosenbrock</td>
<td>Multi-modal with narrow valley</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>F3 Ackley</td>
<td>Multi-modal</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>F4 Griewank</td>
<td>Multi-modal</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>F5 Rastrigin</td>
<td>Very complicated multi-modal</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: An illustrative example showing the influence of adding an unlabelled data into the training data set. Upper: The unlabelled data is beneficial; Lower: The unlabelled data is harmful.
including the Gaussian process surrogate model assisted evolutionary algorithm for medium-scale expensive problems [26] and the ensemble-based generalized surrogate single-objective memetic algorithm [25]. Table 2 lists the statistical results of the compared algorithms on the five test problems with 10-, 20- and 30-dimensions. From Table 2, we can observe that SSL-assisted PSO has obtained better results on all five problems than PSO and SL-assisted PSO. In comparison with CAL-SAPSO, SSL-assisted PSO also obtained better result on Ackley and Rosenbrock functions, which showed that our proposed SSL-assisted PSO is good for problems that are multi-modal and the global optima can be easily distinguished from local optima.

Fig. 8∼12 present the convergence profiles of the compared algorithms on the five benchmark problems with 30 dimensions. From Figures 8∼12, we find that the convergence speed of SSL-assisted PSO is much faster than PSO and SL-assisted PSO on all five benchmark problems except for the Rastrigin function. From Figures 6 and 5, we also notice that CAL-SAPSO, which uses 150 fitness evaluations for offline surrogate training, is not able to achieve much performance improvement during the optimization on the Ackley and Griewank functions. By contrast, SSL-assisted PSO continues to improve its solution quality. Finally, we note that SSL-assisted PSO is outperformed by CAL-SAPSO on the Rastrigin function. The reason we guess is that the Rastrigin function is a multi-modal function but the differences between the global optima and the local ones are small, which makes it more difficult to smooth out the local optima and reduce the effectiveness of using the semi-supervised learning assisted surrogate.

5 CONCLUSIONS

A particle swarm optimization assisted by semi-supervised learning is proposed in this paper for solving computationally expensive problems. Two surrogate models, one being built using labelled data only and the other using both labelled and unlabelled data, are employed to approximate the objective function. A surrogate
Table 2: Comparisons of the statistical results

<table>
<thead>
<tr>
<th>Problem</th>
<th>d</th>
<th>SSL-assisted PSO</th>
<th>SL-assisted PSO</th>
<th>PSO</th>
<th>CAL-SAPSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ellipsoid</td>
<td>10</td>
<td>3.64e+00(5.39e+00)</td>
<td>2.57e+01(1.37e+01)</td>
<td>8.76e+01(2.30e+01)</td>
<td>8.79e-01(8.51e-01)</td>
</tr>
<tr>
<td>Ellipsoid</td>
<td>20</td>
<td>6.88e+00(8.53e+00)</td>
<td>4.71e+01(1.71e+01)</td>
<td>8.42e+02(2.73e+01)</td>
<td>1.58e+00(4.83e-01)</td>
</tr>
<tr>
<td>Ellipsoid</td>
<td>30</td>
<td>3.56e+00(1.52e+00)</td>
<td>1.02e+02(5.87e+01)</td>
<td>1.02e+03(1.46e+02)</td>
<td>4.02e+00(1.08e+00)</td>
</tr>
<tr>
<td>Rosenbrock</td>
<td>10</td>
<td>1.29e+01(6.18e+00)</td>
<td>1.90e+01(6.80e+01)</td>
<td>4.57e+01(1.28e+01)</td>
<td>1.59e+01(3.80e-01)</td>
</tr>
<tr>
<td>Rosenbrock</td>
<td>20</td>
<td>2.33e+01(8.53e+00)</td>
<td>4.60e+01(1.23e+01)</td>
<td>2.65e+02(8.09e+01)</td>
<td>3.59e+01(3.32e-01)</td>
</tr>
<tr>
<td>Rosenbrock</td>
<td>30</td>
<td>4.76e+01(1.02e+01)</td>
<td>7.99e+01(1.63e+01)</td>
<td>6.16e+02(1.13e+02)</td>
<td>5.10e+01(1.15e+01)</td>
</tr>
<tr>
<td>Ackley</td>
<td>10</td>
<td>6.88e+00(1.87e+00)</td>
<td>1.51e+01(2.31e+00)</td>
<td>1.78e+01(1.09e+00)</td>
<td>2.01e+01(2.44e-01)</td>
</tr>
<tr>
<td>Ackley</td>
<td>20</td>
<td>5.24e+00(2.57e+00)</td>
<td>1.62e+01(1.97e+00)</td>
<td>1.87e+01(5.55e+01)</td>
<td>2.01e+01(0.00e+00)</td>
</tr>
<tr>
<td>Ackley</td>
<td>30</td>
<td>7.19e+00(2.94e+01)</td>
<td>1.57e+01(1.94e+00)</td>
<td>1.85e+01(4.71e-01)</td>
<td>1.62e+01(4.13e-01)</td>
</tr>
<tr>
<td>Griewank</td>
<td>10</td>
<td>1.46e+00(3.94e-01)</td>
<td>1.36e+01(1.09e+01)</td>
<td>5.55e+01(1.71e+01)</td>
<td>1.62e+00(1.21e-01)</td>
</tr>
<tr>
<td>Griewank</td>
<td>20</td>
<td>1.09e+00(8.93e-02)</td>
<td>1.12e+01(6.71e+00)</td>
<td>1.51e+02(2.68e+01)</td>
<td>1.62e+00(3.66e-02)</td>
</tr>
<tr>
<td>Griewank</td>
<td>30</td>
<td>1.04e+00(7.66e-02)</td>
<td>1.73e+01(8.43e+00)</td>
<td>2.37e+02(3.99e+01)</td>
<td>9.95e-01(3.99e-02)</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>10</td>
<td>7.57e+00(1.40e+01)</td>
<td>8.68e+01(1.58e+01)</td>
<td>8.86e+01(1.36e+01)</td>
<td>8.88e+01(2.26e+01)</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>20</td>
<td>1.65e+02(3.60e+01)</td>
<td>1.84e+02(2.10e+01)</td>
<td>2.13e+02(1.86e+01)</td>
<td>7.51e-01(1.44e+01)</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>30</td>
<td>2.76e+02(2.99e+01)</td>
<td>2.86e+02(2.49e+01)</td>
<td>3.19e+02(2.05e+01)</td>
<td>8.78e+01(1.65e+01)</td>
</tr>
</tbody>
</table>

Figure 8: Convergence profiles on Ellipsoid

Figure 9: Convergence profiles on Rosenbrock

Figure 10: Convergence profiles on Ackley

Figure 11: Convergence profiles on Griewank

A management strategy is proposed to determine which individuals should be evaluated using the expensive objective function and which should be added to the training data as the unlabelled. Experimental results comparing a few PSO variants including a state-of-the-art algorithm on five 30-dimensional benchmark problems demonstrate that proposed method is promising in finding an acceptable solution on a limited computational budget.
ACKNOWLEDGMENTS

This work was supported in part by National Natural Science Foundation of China (Grant No. 61403272), an EPSRC grant (No. EP/M017869/1), and the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, China.

REFERENCES


