Soft Comput (2015) 19:1461–1475
DOI 10.1007/s00500-014-1283-z

FOCUS

A two-layer surrogate-assisted particle swarm optimization algorithm

Chaoli Sun · Yaochu Jin · Jianchao Zeng · Yang Yu

Published online: 30 April 2014

Abstract Like most evolutionary algorithms, particle swarm optimization (PSO) usually requires a large number of fitness evaluations to obtain a sufficiently good solution. This poses an obstacle for applying PSO to computationally expensive problems. This paper proposes a two-layer surrogate-assisted PSO (TLSAPSO) algorithm, in which a global and a number of local surrogate models are employed for fitness approximation. The global surrogate model aims to smooth out the local optima of the original multimodal fitness function and guide the swarm to fly quickly to an optimum or the global optimum. In the meantime, a local surrogate model constructed using the data samples near each particle is built to achieve a fitness estimation as accurate as possible. The contribution of each surrogate in the search is empirically verified by experiments on uni- and multi-modal problems. The performance of the proposed TLSAPSO algorithm is examined on ten widely used benchmark problems, and the experimental results show that the proposed algorithm is effective and highly competitive with the state-of-the-art, especially for multimodal optimization problems.

Keywords Particle swarm optimization · Surrogate-assisted optimization · Computationally expensive optimization problems

Communicated by Y. Jin.

C. Sun (✉) · J. Zeng
Complex System and Computational Intelligence Laboratory, Taiyuan University of Science and Technology, Taiyuan 030024, Shanxi, China
e-mail: clsun1225@live.cn

J. Zeng
e-mail: zengjianchao@263.net

Y. Jin
Department of Computing, University of Surrey, Guildford GU2 7XH, UK
e-mail: yaochu.jin@surrey.ac.uk

Y. Yu
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 200093, China
e-mail: yuy@lamda.nju.edu.cn

1 Introduction

As a population-based meta-heuristic search algorithm, particle swarm optimization (PSO) has achieved great success on many real-world application problems, such as mechanical design optimization (He et al. 2004), shop scheduling problem (Sha and Hsu 2008) and electric power systems (Abou El-Ela et al. 2008). However, many engineering design optimization problems involve the use of high fidelity simulation methods such as finite element analysis, computational fluid dynamics and computational electro magnetics for quality evaluations, which is often computationally expensive, ranging from several minutes to days of supercomputer time (Lim et al. 2010). Since PSO typically requires thousands of evaluations to achieve a global optimum solution, the application of PSO to this class of expensive problems becomes intractable. One promising approach to reduce computation time for optimization of highly time-consuming optimization problems is to employ computationally cheap approximation models (surrogates) to replace in part the computationally expensive exact function evaluations. Over recent years, surrogate model-assisted evolutionary algorithms have received increasing attention for addressing expensive optimization problems, because the computational effort required to build and use surrogates is usually much lower than that for expensive evaluations (Lim et al. 2010). A variety of surrogate models (also called metamodels or approximation models) have been proposed to be used in evolutionary algorithms (EAs), such as polynomial regression (also known as response surface method) (Lian and Liou 2005), artificial neural network (ANN) (Farina 2002), radial basis function (RBF) (Ong et al. 2006), and Gaussian process (GP) (also referred to Kriging) (Joseph et al. 2008).

In the context of EAs, various approaches for solving computationally expensive problems using surrogate models have been reported. Global-surrogate models are often proposed for EAs to approximate the expensive objective function in the early stage. Ratle (2001) examined strategies for integrating evolutionary search with global surrogate models based on Kriging. Jin et al. (2002) employed an artificial neural network to construct global surrogate models and an empirical criterion was proposed to switch between the expensive exact fitness function and the surrogate model during the search. Ulmer et al. (2003) and Buche et al. (2005) proposed different strategies using GP surrogate models. Liu et al. (2014) proposed a GP-assisted evolutionary algorithm, in which a high-quality global surrogate model was built using dimension reduction techniques for solving medium-scale computationally expensive optimization problems. However, since constructing accurate surrogate models is less likely due to the curse of dimensionality, building local surrogate models has only been more intensively explored recently. Ong et al. (2003, 2004) combined an evolutionary algorithm with a sequential quadratic programming solver in the spirit of Lamarckian learning, in which the trust-region method for interleaving exact models for the objective and constraint functions with computationally cheap surrogate models during local search was employed. Fitness inheritance, which was first proposed by Smith et al. (1995), can be seen as a special local surrogate technique, where the fitness of the individual is inherited from its parents or other individuals. Fonseca et al. (2012) introduced three inheritance surrogate models in genetic algorithms. Recently, many researchers proposed to ensemble different surrogate models (Goel et al. 2007; Acar and Rais-Rohani 2009; Jin and Sendhoff 2004; Lu et al. 2013) and it has been shown from these studies that ensemble models generally outperform most of the individual surrogates. Zhou et al. (2005) proposed a hierarchical surrogate-assisted evolutionary algorithm, in which GP and polynomial regression are used as global surrogate models, for solving computationally expensive optimization problems. The global surrogate model served to pre-screen the EA population for promising individuals, which will then undergo a local search in the form of Lamarckian learning using online local surrogate models. An extension of Zhou et al. (2005) was reported in Zhou et al. (2007), which presented a novel surrogate management framework for solving computationally expensive problems. Tenne and Armfield (2009) proposed a memetic algorithm using variable global and local surrogate-models for optimization of expensive functions. The method also employed the trust-region approach but replaced the quadratic models with the RBF network.

While they are widely used to assist evolutionary algorithms, surrogate models have relatively less often been used to assist PSO for computationally expensive problems. Praveen and Duvigneau (2009) used a RBF metamodel to reduce the cost of PSO in two 20-D aerodynamic shape optimization problems. Parno et al. (2012) used a Kriging surrogate to improve the efficiency of PSO for simulation-based problems and applied it to a 6-D groundwater management problem. Bird and Li (2010) incorporated a regression model into PSO algorithm to improve local convergence. Tang et al. (2013) used a hybrid global surrogate model consisting of a quadratic polynomial and an RBF model to develop a surrogate-based PSO method, which was applied to low-dimensional test problems and engineering design problems. Regis (2014) utilized an RBF surrogate model to identify the most promising trial position for each particle in the swarm. Hendtlass (2007) adopted the fitness inheritance strategies in PSO and added a reliability measure to enhance estimation accuracy. Reyes-Sierra and Coello (2005) incorporate 15 fitness inheritance techniques and four approximation techniques into a multi-objective PSO. Sun et al. (2012) proposed a new fitness inheritance strategy, called FESPSO, in which the fitness value of an individual was inherited not only from its parents, but also its progenitors and brothers. To reduce the evaluation times, Sun et al. (2013) subsequently added a similarity-based strategy for improving estimation quality.

In this paper, a two-layer surrogates-assisted particle swarm optimization (TLSAPSO) algorithm is suggested for solving computationally expensive problems. We believe such techniques are of great interest for further understanding surrogate-assisted optimization, as the search mechanisms and search dynamics of PSO are very different from those of local search methods and selection-based population metaheuristics, where a combination of global and local model has mostly been examined. In TLSAPSO, the surrogate model in the top layer is expected to smooth out local optima of the objective function and guide the swarm to fly to a region where the global optimum is potentially located. To this end, this surrogate should be able to learn a rough contour of the fitness landscape in a wide search space, which is therefore termed a global model. Meanwhile, the surrogate models in the bottom layer, built using data in the neighborhood of each particle, aim to approximate the local fitness landscape as accurately as possible. Therefore, these models are called local surrogate models. Note that the global model is shared by all particles in the swarm, whilst individual local surrogates are built for different particles. Different to the EAs assisted by global and local surrogate models reported in Lim et al. (2010), Zhou et al. (2005, 2007), in our proposed method, no local search is employed.

The paper is organized as follows. Section 2 provides a brief overview of the related techniques. In Sect. 3, the two-layer surrogate-assisted PSO is presented. The algorithm is evaluated empirically in Sect. 4 on ten widely used benchmark problems. Section 5 concludes the paper with a summary and some ideas for future work.

2 Related techniques

2.1 Particle swarm optimization

Consider the following optimization problem:

$$\begin{aligned} &\text{minimize}: f(\mathbf{x}) \\ &\text{subject to}: \mathbf{x}_l \leq \mathbf{x} \leq \mathbf{x}_u \end{aligned} \tag{1}$$

where $f(\mathbf{x})$ is a scalar-valued objective function, $\mathbf{x} \in \Re^D$ is a vector of continuous decision variables, $\mathbf{x}_l$ and $\mathbf{x}_u$ are vectors of the lower and upper bounds of search space, respectively.

The PSO algorithm, simulating the behavior of bird flocking or fish schooling, was originally proposed by Eberhart and Kennedy (1995) in 1995 for solving unconstrained optimization problems. It has been successfully applied to a wide range of problems because of its simplicity and attractive search efficiency. The algorithm starts with a population of particles randomly positioned in the search space, each of which has its own position and velocity. At each iteration, the position and velocity of a particle are updated as

$$\begin{aligned} v_{id}(t+1) = v_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) \\ + c_2 r_2 (p_{gd}(t) - x_{id}(t)) \end{aligned} \tag{2}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \tag{3}$$

where $\mathbf{v}_i(t) = (v_{i1}(t), v_{i2}(t), \ldots, v_{iD}(t))$ and $\mathbf{x}_i(t) = (x_{i1}(t), x_{i2}(t), \ldots, x_{iD}(t))$ are the velocity and position of particle i at iteration t, respectively. $\mathbf{p}_i(t) = (p_{i1}(t), p_{i2}(t), \ldots, p_{iD}(t))$ is the best historical position found by particle i (known as the personal best), $\mathbf{p}_g(t) = (p_{g1}(t), p_{g2}(t), \ldots, p_{gD}(t))$ is the best historical position of the swarm (the global best), r_1 and r_2 are two uniformly generated random numbers in the range [0,1], c_1 and c_2 are positive constants called acceleration coefficients.

A number of variants of PSO have been proposed to improve the convergence of the algorithm. Two most commonly used PSO variants modify the velocity updating rule in Eq. (2), one proposed by Shi and Eberhart (1998) (called the inertia weight model) and the other by Clerc and Kennedy (2002) (called the constriction factor model). In the inertia weight model, the velocity is updated as follows:

$$\begin{aligned} v_{id}(t+1) = \omega v_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) \\ + c_2 r_2 (p_{gd}(t) - x_{id}(t)) \end{aligned} \tag{4}$$

where ω is called inertia weight. Similarly, the constriction factor model uses following equation for updating the velocity

$$\begin{aligned} v_{id}(t+1) = \chi(v_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) \\ + c_2 r_2 (p_{gd}(t) - x_{id}(t))) \end{aligned} \tag{5}$$

with

$$\chi = \frac{2k}{2 - \phi - \sqrt{\phi^2 - 4\phi}} \tag{6}$$

where $\phi = c_1 + c_2$. In general, $\phi > 4$, and therefore, c_1 and c_2 are usually set to 2.05. k is a real number in the range (0,1].

Eberhart and Shi (2000) compared the performance of PSO using the inertia weight model and the constriction factor model, and their experimental results showed a PSO using constriction factor while limiting the maximum velocity $v_{\max}$ to the maximum position $x_{\max}$ on each dimension performed the best. So in this paper, we use Eq. (5) to update the velocity.

2.2 Radial basis function network (RBFN)

RBFN is one of the most commonly used approximation models, which has successfully been used for function approximation, time series prediction and control (Kattan and Galvan 2012). It can be seen as a variant of an ANN that uses radial basis functions as the activation function. RBFN is conceptually simple and can perform both interpolation and extrapolation from the known data-points (Kattan and Galvan 2012). So in this paper, we adopt RBFNs both for global and local surrogate models.

A RBF is a real-valued function $\varphi : \Re^n \to \Re$, with its value depending only on the distance from some point $\mathbf{c}$, called center, so that $\varphi(\mathbf{x}) = \varphi(\mathbf{x}_q - \mathbf{c})$. The point $\mathbf{c}$ is a parameter of the function and the point $\mathbf{x}_q$ is a query point to be estimated. The norm is usually Euclidean, so $\mathbf{x}_q - \mathbf{c}$ is the Euclidean distance between $\mathbf{c}$ and $\mathbf{x}_q$. There are several types of RBF functions, including Gaussian, multiquadric, inverse quadratic and inverse multiquadric. In this paper, the following Gaussian function is used.

$$\varphi(\mathbf{x}) = \exp\left(-\frac{(\mathbf{x}_q - \mathbf{c})^2}{2\sigma^2}\right) \tag{7}$$

where $\sigma > 0$ is the width of the Gaussian. RBFs are typically used to build function approximation of the following form:

$$y(\mathbf{x}) = \omega_0 + \sum_{i=1}^{N} \omega_i \phi(\mathbf{x}_q - \mathbf{c}_i) \tag{8}$$

where N is the number of RBFs, each associated with a different center $\mathbf{c}_i$, a width β_i, and weighted by a coefficient ω_i, plus a bias term ω_0. In principle, an RBFN can approximate any continuous function with an arbitrary accuracy, if a sufficiently large number N of radial basis function is used. The bias ω_0 can be set to the mean of the values of the known data-points from the training set that are used to train the surrogate model, or set to 0.

3 Two-layer surrogate-assisted PSO (TLSAPSO)

As suggested in Lim et al. (2010), approximation errors introduced by surrogate models in evolutionary algorithms can have both negative and positive impacts. The negative impact, called 'curse of uncertainty' indicates the phenomenon that inaccurate surrogates may lead to EA to a false optimum. By contrast, the positive impact, called 'bless of uncertainty', refers to the potential benefit achieved by the use of surrogates in removing local optimums. To mitigate the 'curse of uncertainty' and benefit from the 'bless of uncertainty', the authors suggested to conduct local search using both a local surrogate and well as a global surrogate.

Inspired by the idea in Lim et al. (2010), in this paper, a two-layer surrogate model is proposed to assist the search in PSO. The surrogate model in the top layer serves to smooth out local optimums, thus speeding up the search. Figure 1 gives an example to show the positive impact using a global surrogate. Due to the smoothing effect of the global surrogate, the search on the surrogate can be much faster than on the exact fitness function. Note that for constructing the global surrogate, data samples distributed in a large search should be used, however, with relatively lower approximation accuracy. On the other hand, the local surrogate models are constructed to approximate fitness landscape locally but more accurately. For an accurate local approximation, many data points that lie in the vicinity of the concerned particle are required, as illustrated in Fig. 2. By properly combining the global surrogate model with local surrogate models, we hope that PSO can find the global optimum quickly and accurately.

Algorithm 1 gives an overview of the proposed TLSAPSO. In Algorithm 1, $\mathbf{p}_i$ is the personal best historical position of particle i, $\mathbf{p}_g$ is the best historical position of the swarm, and $\tilde{f}(\mathbf{x}_i)$ represents the approximated value on position $\mathbf{x}_i$. A global database is used to store all particles evaluated using the original fitness function, including positions and

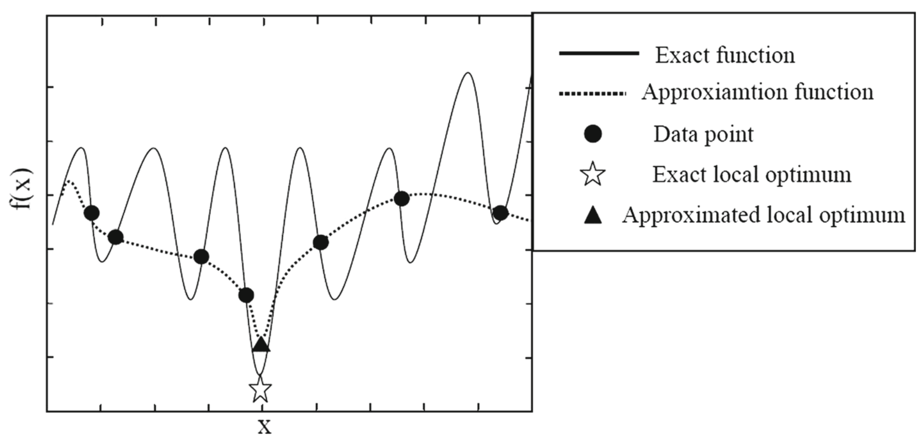

Fig. 1 An example to show the smoothing effect of a global surrogate model. Training data should spread in a wide search space

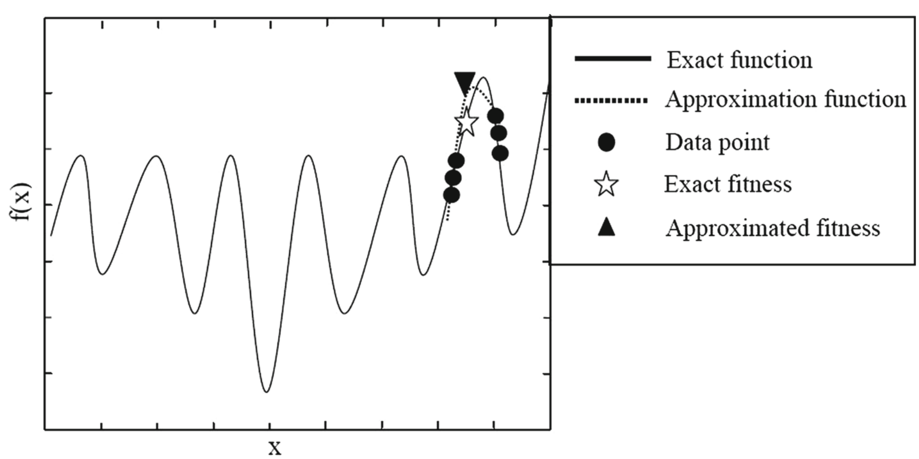

Fig. 2 An example to show the local approximation of a local surrogate model. Many data samples near the concerned particle are needed

corresponding fitness values. The data samples in the global database that are most relevant to the position of the current swarm will be used to build the global surrogate. A local database of a fixed memory size is set up for each particle to store the neighboring particles that have been evaluated using the original fitness function in recent generations, part of which will be used to build a local surrogate for each particle. The global and local surrogates are embedded in the PSO with a constriction factor (CPSO). In the following, we present the details of the TLSAPSO.

Algorithm 1 The two-layer surrogate-assisted PSO algorithm

```
Initialize a population,t=0;
Evaluate the fitness of all particles using the real objective function;
Archive all positions and fitness values into both the global database
  and the local one;
Determine the personal best historical position p_i(t) for each particle
  i;
Determine the best historical position p_g(t) of the swarm
while the stopping criterion is not met do
  Update velocity and position of each particle i using Eq. (5) and
  Eq. (3);
  Approximate fitness values using surrogate models for each particle;
  if there exists at least a particle that f~(x_i(t+1)) < f(p_i(t)) then
    Calculate the fitness value of each particle i that f~(x_i(t+1)) <
    f(p_i(t)) using the real objective function;
  else
    Calculate fitness values using the real objective function for all
    particles in the current swarm;
  end if
  if the global database or the local database is needed to be updated
  then
    Update the global database and/or the local one;
  end if
  Determine the personal best historical position p_i(t+1) for each
  particle i;
  Determine the best historical position p_g(t+1) of the swarm;
  t=t+1;
end while
```

3.1 Fitness approximation strategy

The initial population of the PSO is generated using the Latin hypercube sampling method. All particles are evaluated using the original fitness function to create the initial samples, which are stored in the global database. Then, an RBFN for the global surrogate model is trained using all data in the global database. Note however, that from the second generation onward, it is likely that only part of the data in the global database will be employed for training the global surrogate to reduce the computation time on the one hand, and to ensure the global nature of the surrogate on the other hand. For example, in Fig. 3, particles of the current swarm are located in a sub-region of the whole search space, while the data in the global database are distributed in a much wider space. Even though a global surrogate is targeted, it is not necessary to use all data to build the surrogate. In the proposed TLSAPSO approach, the subspace in which the data samples are used for training is adapted according to the current positions of the swarm. Let

$$\mathrm{max}d_d = \max(\{x_{id}(t+1), \quad i = 1, 2, \ldots, n\}) \tag{9}$$

$$\mathrm{min}d_d = \min(\{x_{id}(t+1), \quad i = 1, 2, \ldots, n\}) \tag{10}$$

where $\mathrm{max}d_d$ and $\mathrm{min}d_d$ refer to the maximum and minimum values of the current swarm on dth dimension at iteration $t+1$. n is the swarm size. Then, we define a subspace where

$$\mathrm{sp\_}x\mathrm{max}_d(t+1) = \min\{\mathrm{max}d_d + \alpha(\mathrm{max}d_d - \mathrm{min}d_d), x\mathrm{max}_d\} \tag{11}$$

$$\mathrm{sp\_}x\mathrm{min}_d(t+1) = \max\{\mathrm{min}d_d - \alpha(\mathrm{max}d_d - \mathrm{min}d_d), x\mathrm{min}_d\} \tag{12}$$

where $x\mathrm{max}_d$ and $x\mathrm{min}_d$ are the maximum and minimum values of the whole search space on dimension d, $\mathrm{sp\_}x\mathrm{max}_d(t+1)$ and $\mathrm{sp\_}x\mathrm{min}_d(t+1)$ are the maximum and minimum values of the subspace on dth dimension at iteration $t+1$. α is a spread coefficient between 0 and 1 to allow the surrogate to use data samples that are outside the space occupied by the current swarm, as illustrated in Fig. 3. As a result, the global surrogate model is constructed using the data samples in the whole decision space in the beginning and as the search proceeds only part of the samples in the global database that are near the location of the current swarm will be used for training.

Once it is available, the global surrogate model is used to estimate the fitness of all particles in the swarm. The fitness values of the particles estimated by the global surrogate model are denoted as $\tilde{f}_g$. As previously discussed, the global surrogate model may have a large approximation error and cannot accurately approximate the fitness values of all particles due to the curse of dimensionality or poor distribution and limited number of training samples (Jin 2005). However, if there are adequate data samples around the particle, it is desirable to create a local surrogate for more accurate fitness estimation. Given the local surrogate model, the fitness of this particle can also be estimated by its local surrogate model, which is denoted as $\tilde{f}_l$. The size of a particle's neighborhood is also adaptively set according to the size of the current swarm as follows:

$$\mathrm{local\_size}_d = \beta(\mathrm{max}d_d - \mathrm{min}d_d) \tag{13}$$

Note that the local surrogate models of different particles may often differ with each other because data samples around each particle are usually not the same.

The question now is, if a local surrogate is available to a particle, whether the fitness estimated using the global sur-

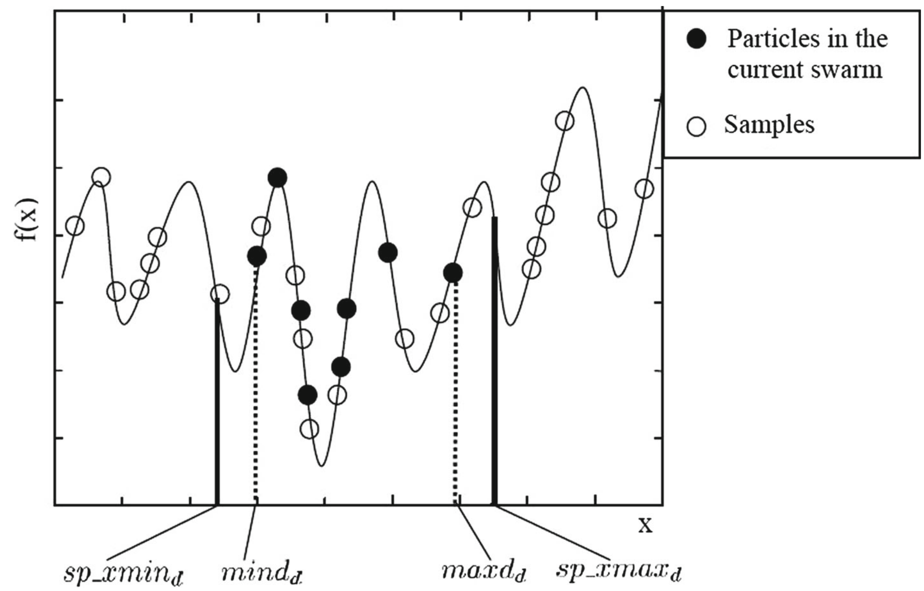

Fig. 3 Determination of the range of data samples for training the global surrogate

rogate or the one using the local surrogate should be used. Algorithm 2 shows the surrogate management strategy used in TLSAPSO. As we can see, if there are not enough historical data around a particle, the fitness value of this particle can be approximated by the global surrogate model, that is $\tilde{f}(\mathbf{x}) = \tilde{f}_g(\mathbf{x})$. In case both estimates are available, we choose the lower one (smaller one for minimization problem) to be the final fitness value of the particle in order not to leave out any potentially promising position. From Algorithm 2, we can also see that the surrogate models can be categorized into layers, where the global surrogate model is on the top layer and the local surrogate models are on the bottom layer. Consequently, we call our proposed method a TLSAPSO algorithm.

Algorithm 2 The management of the two-layer surrogate

```
Construct a global surrogate model;
Approximate a fitness value for each individual in the swarm using the global surrogate model;
for each particle i in the swarm do
  Find its neighbors in the local database;
  if there are enough samples to construct a local surrogate then
    Construct a local surrogate model;
    Approximate the fitness of particle i using the local surrogate model;
    f~(x_i(t+1)) = min{f~_g(x_i(t+1)), f~_l(x_i(t+1))}
  else
    f~(x_i(t+1)) = f~_g(x_i(t+1))
  end if
end for
```

3.2 Surrogate update and database management

To prevent the PSO from converging to a false optimum, the surrogate models need to be used together with the original fitness function. In PSO, the personal best positions as well as the global best positions play a central role in ensuring the whole swarm to converge to a true optimum. Therefore, TLSAPSO always computes the fitness of all personal best and global best particles using the real fitness function to guarantee a correct convergence. Algorithm 3 describes the overall strategies for updating the surrogates, the global database as well as the local archives, which corresponds to lines 9–16 in Algorithm 1.

In Algorithm 3, $\mathbf{x}_i(t+1)$ is the current position of particle i, $f(\mathbf{x}_i(t+1))$ and $\tilde{f}(\mathbf{x}_i(t+1))$ are its real and approximate fitness values, respectively. $\mathbf{x}' = (x_1', x_2', \ldots, x_D')$ is a position the swarm has visited and has obtained its real fitness value and $f(\mathbf{x}')$ is its corresponding fitness value. Both $\mathbf{x}'$ and $f(\mathbf{x}')$ are saved in the local database of particle i. δ_1 and δ_2 are two predefined thresholds used to judge whether the data of a new position should be added into the local database or the global database.

In Algorithm 3, it can happen that no particle will be evaluated using the real fitness function because all approximated fitness value of the swarm are worse than the current personal best position. In this case, all particles will be re-evaluated using the real fitness function to guarantee the correct convergence of the TLSAPSO.

4 Experimental studies

To evaluate the performance of our proposed algorithm, ten widely used benchmark problems suggested in Suganthan et al. (2005) are adopted. The dimension of all the problems is set to $D = 30$. The characteristics of these test problems are listed as in Table 1.

Algorithm 3 Fitness evaluation and updating of database

for each particle i in the population **do**
  if $\tilde{f}(\mathbf{x}_i(t+1)) < f(\mathbf{p}_i(t))$ **then**
    Calculate the fitness of particle i using the real objective function;
    Update personal best historical position: $\mathbf{p}_i(t+1) = \min\{f(\mathbf{x}_i(t+1)), f(\mathbf{p}_i(t))\}$;
  end if
end for
if none fitness of particles in the current swarm is calculated using the real objective function **then**
  Calculate the fitness of each particle i using real objective function;
  Update the personal best historical position for each particle i: $\mathbf{p}_i(t+1) = \min\{f(\mathbf{x}_i(t+1)), f(\mathbf{p}_i(t))\}$;
end if
for each particle i in the population **do**
  if the fitness is calculated with real objective function **then**
    if on all dimension d, $|x_{id}(t+1) - x'_d| < \delta_1$ and $|\frac{f(\mathbf{x}_i(t+1)) - f(\mathbf{x}')}{f(\mathbf{x}_i(t+1))}| > \delta_2$ **then**
      Store the position and corresponding fitness into the local database;
    end if
    if $|\frac{\tilde{f}(\mathbf{x}_i(t+1)) - f(\mathbf{x}_i(t+1))}{f(\mathbf{x}_i(t+1))}| > \delta_2$ **then**
      Archive the positional information and corresponding fitness into the global database;
    end if
  end if
end for

The parameters of the TLSAPSO used in our experiments are set as follows: the size of the swarm is 60, the cognitive and social parameters are both set to 2.05. The maximum velocity $v_{\max}$ is set to the maximum position $x_{\max}$ on each dimension. The maximum number of real fitness evaluations is set to 10,000, which is the same as used in Lu et al. (2011). All compared algorithms perform ten independent runs on each test problem in Matlab®2009. In TLSAPSO, the surrogate models are built using the "newrb" function provided in the toolbox of Matlab. The desired mean squared errors of the global and the local surrogate models are set to 0.1 and 0.01, respectively, the maximum number of hidden neurons in the RBFNs is set to 20 for both kinds of surrogate models. The number of hidden nodes is adapted up to the maximum according to the desired accuracy. The parameter SPREAD in "newrb' function, which corresponds to the parameter σ in Eq. (7), is important, as too large a spread requires a lot of neurons to fit a rugged fitness function, while too small a spread means many neurons will be required to fit a smooth function and the network may not be able to generalize well. Considering the above factors, the SPREAD is set adaptively according to the samples used to construct a surrogate model.

$$\mathrm{SPREAD} = \min\left\{\max\{\max\{\mathrm{samples}_d\} - \min\{\mathrm{samples}_d\}, 0\},\; d = 1, 2, \ldots, D\right\} \tag{14}$$

where $\mathrm{samples}_d$ represents the values on the dth dimension of the data set in a given range. Parameters α , β, δ_1 and δ_2 are all empirically set in our experiments. After many tries, we found that with the values of α and β increased, the time for training the global or local surrogates considerably increased without improving the quality of the surrogates. On the other hand, if these parameters are set too small, the surrogate

Table 1 Characteristics of ten benchmark problems

	Benchmark problems	Characteristics	Decision space	Fitness value of global optimum
F1	Shifted Sphere Function	Unimodal	$\mathbf{x} \in [-100, 100]^D$	$f_1(\mathbf{x}^*) = -450$
F2	Shifted Schwefels Problem 1.2	Unimodal	$\mathbf{x} \in [-100, 100]^D$	$f_2(\mathbf{x}^*) = -450$
F3	Shifted Rotated High Conditioned Elliptic Function	Unimodal	$\mathbf{x} \in [-100, 100]^D$	$f_3(\mathbf{x}^*) = -450$
F4	Shifted Schwefels Problem 1.2 with Noise in Fitness	Unimodal	$\mathbf{x} \in [-100, 100]^D$	$f_4(\mathbf{x}^*) = -450$
F5	Schwefels Problem 2.6 with Global Optimum on Bounds	Unimodal	$\mathbf{x} \in [-100, 100]^D$	$f_5(\mathbf{x}^*) = -310$
F6	Shifted Rosenbrocks Function	Multimodal, having a very narrow valley from local optimum to global optimum	$\mathbf{x} \in [-100, 100]^D$	$f_6(\mathbf{x}^*) = 390$
F7	Shifted Rotated Griewanks Function without Bounds	Multimodal, no bounds for variables	Initialize population in $[0, 600]^D$, global optimum is outside of initialization range	$f_7(\mathbf{x}^*) = -180$
F8	Shifted Rotated Ackleys Function with Global Optimum on Bounds	Multimodal, global optimum on the bound	$\mathbf{x} \in [-32, 32]^D$	$f_8(\mathbf{x}^*) = -140$
F9	Shifted Rastrigins Function	Multimodal, local optima's number is huge	$\mathbf{x} \in [-5, 5]^D$	$f_9(\mathbf{x}^*) = -330$
F10	Shifted Rotated Rastrigins Function	Multimodal, local optima's number is huge	$\mathbf{x} \in [-5, 5]^D$	$f_{10}(\mathbf{x}^*) = -330$

Table 2 Comparative results on F1 and F6

	Opt.	Approach	Best	Mean	Worst	Std.
F1	−4.50e+02	CPSO	4.2760e+02	2.7140e+03	7.1566e+03	2.1724e+03
		CPSO_L	7.2905e+01	2.7903e+03	7.3201e+03	2.2028e+03
		CPSO_G	−4.5000e+02	−2.5742e+02	2.2923e+02	2.5074e+02
		TLSAPSO	−4.5000e+02	−4.5000e+02	−4.4999e+02	3.9000e−03
F6	3.90e+02	CPSO	1.2944e+06	2.6445e+08	7.4144e+08	2.5924e+08
		CPSO_L	2.5734e+06	1.6365e+08	6.3167e+08	2.2317e+08
		CPSO_G	1.0673e+03	4.6523e+06	2.5662e+07	8.8134e+06
		TLSAPSO	5.8234e+02	1.5715e+03	6.4199e+03	1.7562e+03

models cannot reflect the properties of the overall or part of the problem. Therefore, in our experiments, α in Eqs. (11) and (12) is set to 0.25, β in Eq. (13) is set to 0.5. Parameters δ_1 and δ_2 determine whether the position calculated with the real objective function should be added in the local or global databases. If δ_1 is too small and δ_2 is too big, important data points that have potential important influence on training will be lost. If δ_1 is too big and δ_2 is too small, many redundant training data will be added into the database. So δ_1 and δ_2 are both set to 10^{-3} in our experiments.

To verify our hypothesis that in TLSAPSO, the global surrogate model is able to smooth out the local optimums, while the local ones are expected to accurately approximate the local fitness landscape, we at first conducted experiments on two selected test problems, one unimodal (F1) and the other multimodal (F6) using the above settings. Table 2 gives the comparative results of the four algorithms: CPSO is the particle swarm optimization algorithm with a constriction factor without using surrogate, CPSO_L is the CPSO algorithm with local surrogate models only, CPSO_G is the CPSO algorithm with the global surrogate model only, TLSAPSO is the proposed PSO using two-layer surrogate models. “Opt.” represents the optimal solution currently known for each problem, “Best”, “Mean” and “Worst” represent the best, the mean and the worst values of optimal solutions achieved in ten independent runs. “Std.” stands for the standard deviation of the obtained optimal solutions in the 10 runs. Figures 4 and 5 present the convergence profile of the compared algorithms on these two functions.

From the results presented in Table 2 and Figs. 4, 5, we can draw the following conclusions. First, use of local surrogates only cannot effectively speed up the PSO search, neither on unimodal nor on multimodal optimization problems. Second, use of a global surrogate only can accelerate the search both on unimodal and multimodal functions, although in the multimodal case, a global surrogate only may fail to find the global optimum. Third, a combination of a global surrogate and local surrogates can take advantage of the benefits brought by both the global and local models, therefore can work well on both unimodal and multimodal optimization problems.

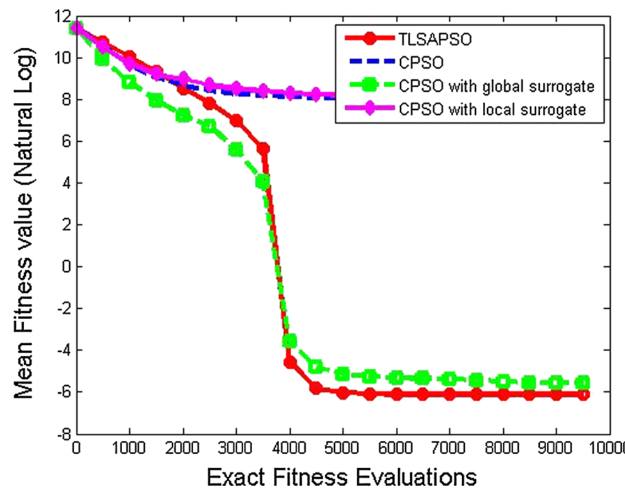

Fig. 4 The convergence profile on F1

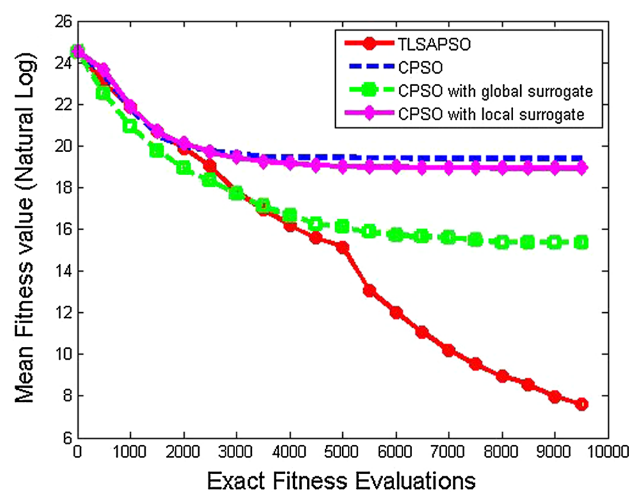

Fig. 5 The convergence profile on F6

To gain deeper insight into the individual contributions of the global and local surrogates to the improvement of the fitness during the search, we again use functions F1 and F6

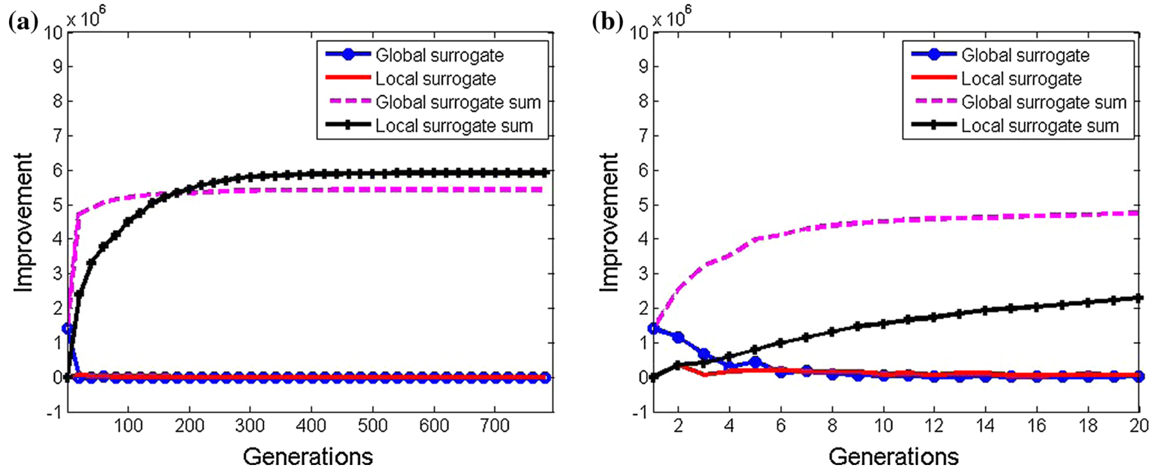

Fig. 6 Results on F1. **a** Contributions to fitness improvement in the whole search process; **b** contributions in the first 20 generations

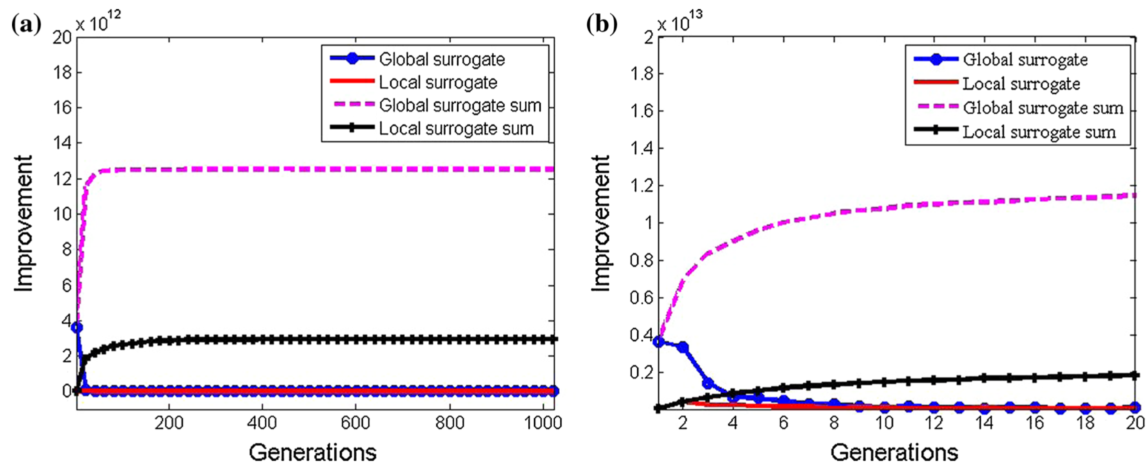

Fig. 7 Results on F6. **a** Contributions to fitness improvement in the whole search process; **b** contributions in the first 20 generations

as two representative examples to examine how much fitness improvement has been achieved when the global surrogate or local surrogates are used for fitness evaluation. The fitness gain of a surrogate is calculated in the following way: If an estimated fitness is better than the current pbest, and the fitness after re-evaluation using the real fitness function is indeed better, this fitness improvement is attributed to the surrogate. If this better fitness was predicted by the global surrogate, this fitness improvement is attributed to the global model. If this better fitness was predicted by the local surrogate, the fitness improvement will be attributed to the local surrogate, refer to Algorithm 3. Figures 6, 7 plot the individual as well as the aggregated contributions of the local and global surrogates for function F1 and F6, respectively. From Fig. 6, we can see for the unimodal function F1, the global surrogate contributes more than local one in the early stage of the search, refer to Fig. 6b. However, as the search proceeds, the local surrogate contributes more than the global one, and the total contribution of the local models is larger than the global model, as shown in Fig. 6a. Interestingly, the global surrogate contributes more than the local surrogates in the whole search process for the multimodal function F6, as shown in Fig. 7a. These results agree with our conjecture that global surrogate may speed up search for multimodal functions by smoothing out the local optimums, in particular in the early stage of the search. On the other hand, local surrogate may be more important for unimodal functions or when the search is approaching the optimum.

In the following, we compare the TLSAPSO with the CPSO without using surrogates, and the FESPSO (Sun et al. 2012) on the ten benchmark problems listed in Table 1. FESPSO is a PSO algorithm assisted by fitness estimation using the inheritance strategy. Based on the results presented above, PSO with the global and local surrogates only will be left out from the comparison in the following experiments.

Table 3 shows the comparison of results obtained by CPSO, FESPSO and TLSAPSO on the ten test problems using the same experimental settings. Figures 8, 9, 10, 11,

Table 3 Comparative results from TLSAPSO, FESPSO (Sun et al. 2012) and CPSO

	Opt.	Approach	Best	Mean	Worst	Std.
F1	−4.50e+02	CPSO	4.2760e+02	2.7140e+03	7.1566e+03	2.1724e+03
		FESPSO	4.7899e+02	2.4010e+03	5.1174e+03	1.7986e+03
		TLSAPSO	−4.5000e+02	−4.5000e+02	−4.4999e+02	3.9000e−03
F2	−4.50e+02	CPSO	4.8460e+03	7.9951e+03	1.1362e+04	2.2039e+03
		FESPSO	9.8467e+02	3.0824e+03	6.0068e+03	1.7129e+03
		TLSAPSO	3.5734e+03	5.7497e+03	7.9880e+03	1.4364e+03
F3	−4.50e+02	CPSO	1.0616e+07	3.0398e+07	9.9308e+07	2.5642e+07
		FESPSO	8.3308e+06	5.5926e+07	2.2536e+08	6.7418e+07
		TLSAPSO	5.6473e+06	1.5712e+07	3.0107e+07	7.7189e+06
F4	−4.50e+02	CPSO	8.5185e+03	1.6597e+04	2.7170e+04	5.3545e+03
		FESPSO	9.0492e+03	1.8508e+04	2.9075e+04	7.2387e+03
		TLSAPSO	1.0451e+04	1.7458e+04	2.5537e+04	3.8397e+03
F5	−3.10e+02	CPSO	6.1334e+03	1.1940e+04	2.0313e+04	3.9721e+03
		FESPSO	7.9465e+03	1.2036e+04	1.6727e+04	2.8412e+03
		TLSAPSO	5.2499e+03	1.0082e+04	1.5392e+04	2.9308e+03
F6	3.90e+02	CPSO	1.2944e+06	2.6445e+08	7.4144e+08	2.5924e+08
		FESPSO	3.7162e+06	5.3199e+08	1.4675e+09	4.7445e+08
		TLSAPSO	5.8234e+02	1.5715e+03	6.4199e+03	1.7562e+03
F7	−1.80e+02	CPSO	−1.7649e+02	−1.7348e+02	−1.6622e+02	3.1381e+00
		FESPSO	−1.7893e+02	−1.7713e+02	−1.7407e+02	1.5909e+00
		TLSAPSO	−1.7879e+02	−1.7765e+02	−1.7528e+02	1.0289e+00
F8	−1.40e+02	CPSO	−1.1900e+02	−1.1891e+02	−1.1881e+02	8.2500e−02
		FESPSO	−1.1956e+02	−1.1937e+02	−1.1907e+02	1.4340e−01
		TLSAPSO	−1.1900e+02	−1.1892e+02	−1.1885e+02	4.9300e−02
F9	−3.30e+02	CPSO	−2.5433e+02	−2.1893e+02	−1.8760e+02	2.4030e+01
		FESPSO	−2.8216e+02	−2.3747e+02	−1.9489e+02	2.9339e+01
		TLSAPSO	−2.7336e+02	−2.2924e+02	−2.0043e+02	2.3965e+01
F10	−3.30e+02	CPSO	−2.1180e+02	−1.5538e+02	−9.9515e+01	3.4173e+01
		FESPSO	−2.1098e+02	−1.5694e+02	−6.0503e+02	5.0590e+01
		TLSAPSO	−2.6029e+02	−1.9114e+02	−1.1391e+02	4.9602e+01

12, 13, 14, 15, 16, 17 plot the convergence profiles of the compared algorithms over the number of real fitness evaluations. A t test of the optimization results with a significance of 5 % has been performed and listed in Table 4. To make a fair comparison, the initial population of the FESPSO is also generated using the Latin hypercube sampling method, and the velocity is updated using Eq. (5).

From Tables 3 and 4, we can see that TLSAPSO can obtain competitive or better results than CPSO on all the 10 benchmark problems when a maximum of 10,000 fitness evaluations is used. Although TLSAPSO does not improve much after 10,000 fitness evaluations, it can converge much faster in the early stage of the search, as seen from Figs. 8, 9, 10, 11, 12, 13, 14, 15, 16, 17. Specially, compared to CPSO, the 'bless of uncertainty' brought by the global surrogate model can be highlighted in the optimization of the multi-model problems

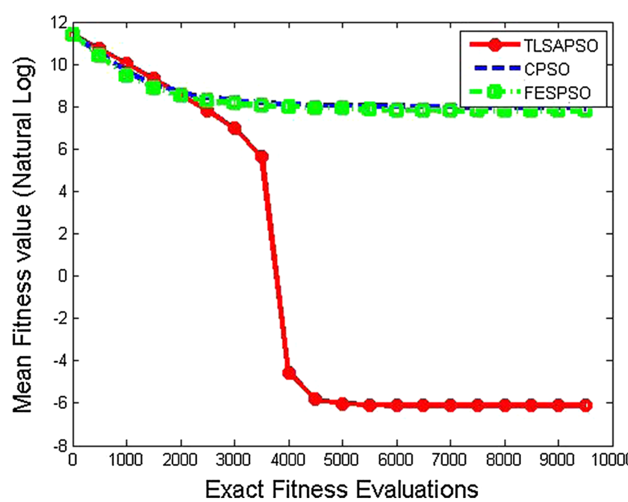

Fig. 8 The convergence profile on F1

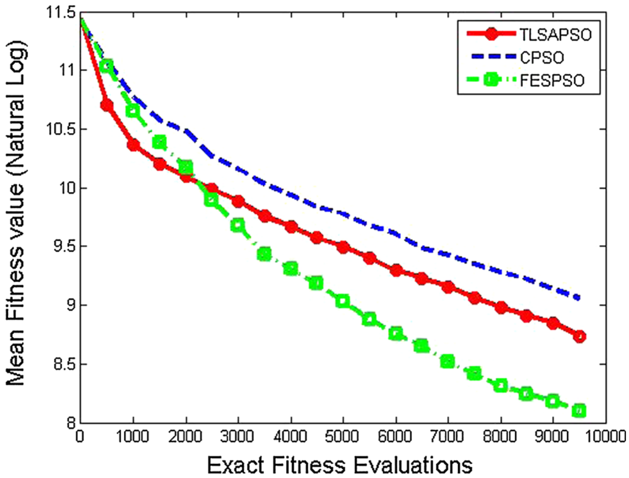

Fig. 9 The convergence profile on F2

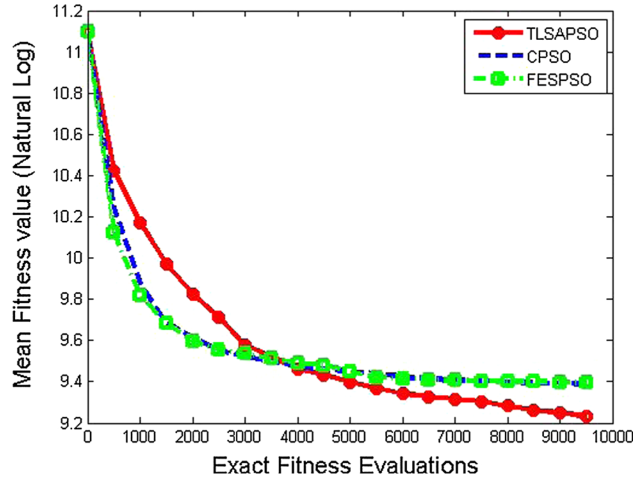

Fig. 12 The convergence profile on F5

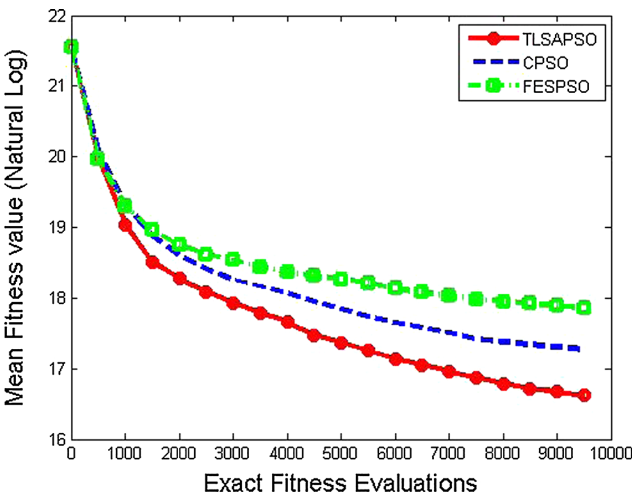

Fig. 10 The convergence profile on F3

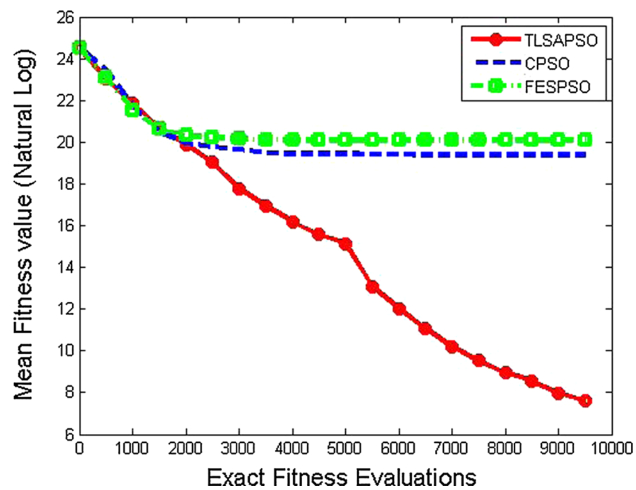

Fig. 13 The convergence profile on F6

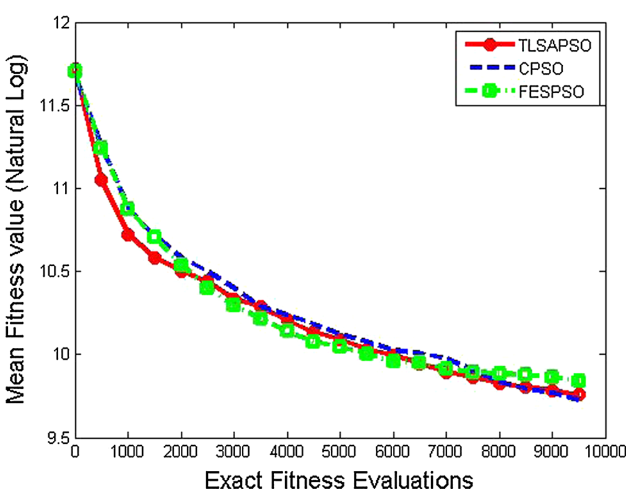

Fig. 11 The convergence profile on F4

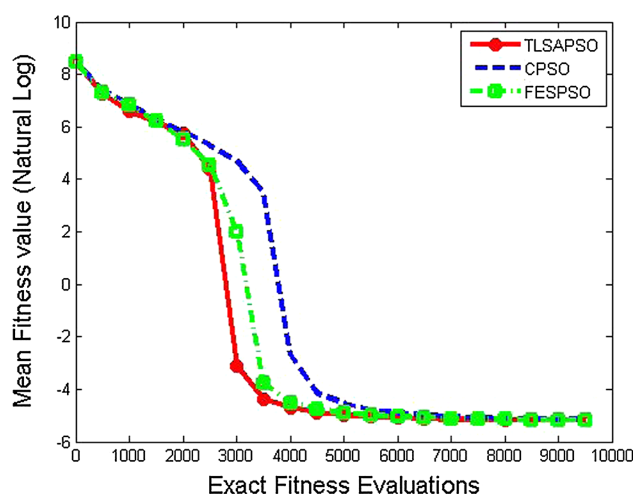

Fig. 14 The convergence profile on F7

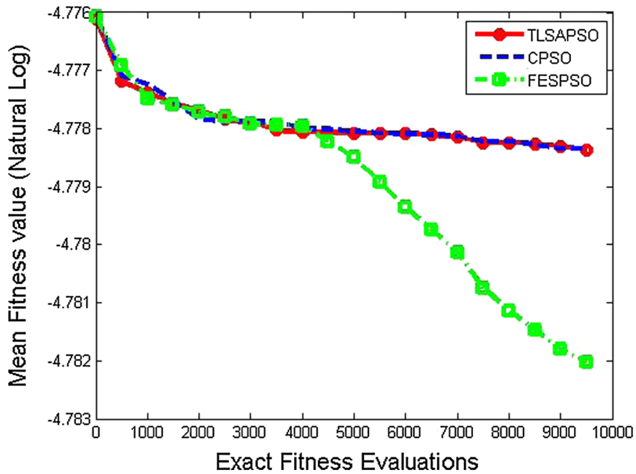

Fig. 15 The convergence profile on F8

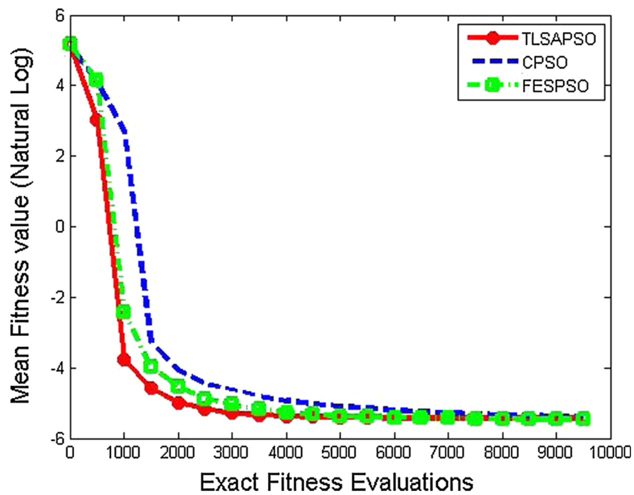

Fig. 16 The convergence profile on F9

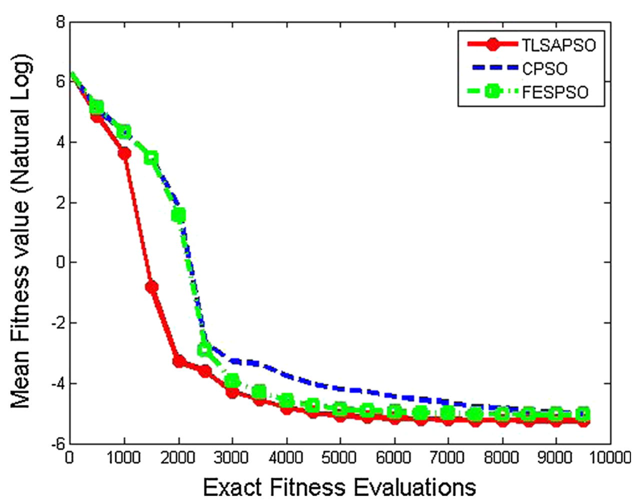

Fig. 17 The convergence profile on F10

Table 4 Results of a t test with 5 % significance level comparing the results listed in Table 3

	CPSO	FESPSO
F1	w	w
F2	w	l
F3	d	d
F4	d	d
F5	d	d
F6	w	w
F7	w	d
F8	d	l
F9	d	d
F10	d	d

"w", "d" and "l" stand for "win", "draw" and "lose", respectively

except for F8. To understand why TLSAPSO failed to perform well on F8, let us take a closer look into this optimization problem. We find that the global optimum of F8 is on the boundary of the search space and is located in a very narrow region, while most local optimums are almost equally good. We also find that in the search, neither CPSO nor TLSAPSO can find the global optimum. Instead, they oscillate between different local optimums. By contrast, TLSAPSO converges much faster than CPSO on other multimodal problems, refer to Figs. 13, 14, 16 and 17.

Comparing TLSAPSO with FESPSO on the multimodal problems, we can find from Tables 3 and 4 that the former performed much better than the latter on F6, comparable on F7, F9, and F10, and worse on F8. This is an interesting observation, and the reason might be attributed to the fact that the fitness approximation strategy in FESPSO may not be local, when two particles are similar (close to each other) in the search space but have very different fitness values, e.g., F8 near the global optimum. In this case, FESPSO may outperform TLSAPSO. On the other hand, though TLSAPSO and FESPSO obtained comparable results on F7, F8 and F10, the former converged much faster to an optimum very close to the global optimum than the latter as shown in Figs. 14, 16 and 17.

Different from multimodal problems, the global surrogate cannot accelerate search by smoothing out local optimums for unimodal problems. Presumably, if the global fitness landscape of a unimodal fitness function is easy to approximate using a small number of samples, e.g., if the function is symmetric, use of a global model will also help locate the global optimum more quickly. This has also been empirically confirmed by our empirical results. For example, TLSAPSO performed much better than CPSO on F1 (sphere function), refer to Fig. 8. However, if the fitness landscape of a unimodal function becomes more complicated, it will be very difficult to approximate in a high-dimensional space using a small

Table 5 The comparative results fromTLSAPSO and surrogate-assisted DE (Lu et al. 2011)

	Opt.	Approach	*Best*	*Mean*	*Worst*	*Std.*
F1	−4.50e + 02	TLSAPSO	−4.5000e + 02	−4.5000e + 02	−4.4999e + 02	3.9000e−03
		SVR-DE	4.56e−01	6.32e−01	8.58e−01	9.19e−02
		SVC-DE	6.37e−02	1.07e−01	2.24e−01	3.67e−02
F2	−4.50e + 02	TLSAPSO	3.5734e + 03	5.7497e + 03	7.9880e + 03	1.4364e + 03
		SVR-DE	6.72e + 03	1.64e + 04	2.45e + 04	4.87e + 03
		SVC-DE	1.72e + 03	3.54e + 03	7.12e + 03	1.33e + 03
F3	−4.50e + 02	TLSAPSO	5.6473e + 06	1.5712e + 07	3.0107e + 07	7.7189e + 06
		SVR-DE	5.82e + 07	1.10e + 08	1.68e + 08	2.75e + 07
		SVC-DE	7.38e + 06	1.80e + 07	3.42e + 07	5.75e + 06
F4	−4.50e + 02	TLSAPSO	1.0451e + 04	1.7458e + 04	2.5537e + 04	3.8397e + 03
		SVR-DE	1.20e + 04	2.70e + 04	3.83e + 04	7.06e + 03
		SVC-DE	3.67e + 03	7.71e + 03	1.27e + 04	2.77e + 03
F5	−3.10e + 02	TLSAPSO	5.2499e + 03	1.0082e + 04	1.5392e + 04	2.9308e + 03
		SVR-DE	7.30e + 02	2.24e + 03	3.28e + 03	5.69e + 02
		SVC-DE	1.49e + 03	2.39e + 03	3.27e + 03	5.71e + 02
F6	3.90e + 02	TLSAPSO	5.8234e + 02	1.5715e + 03	6.4199e + 03	1.7562e + 03
		SVR-DE	5.11e + 06	2.32e + 07	7.16e + 07	1.43e + 07
		SVC-DE	1.08e + 02	2.54e + 03	1.04e + 004	3.11e + 03
F7	−1.80e + 02	TLSAPSO	−1.7879e + 02	−1.7765e + 02	−1.7528e + 02	1.0289e + 00
		SVR-DE	1.02e + 00	1.06e + 00	1.12e + 00	2.35e−02
		SVC-DE	1.17e−01	4.03e−02	4.40e−03	3.15e−02
F8	−1.40e + 02	TLSAPSO	−1.1900e + 02	−1.1892e + 02	−1.1885e + 02	4.9300e−02
		SVR-DE	2.09e + 01	2.11e + 01	2.12e + 01	6.39e−02
		SVC-DE	2.09e + 01	2.08e + 01	2.12e + 01	6.61e−02
F9	−3.30e + 02	TLSAPSO	−2.7336e + 02	−2.2924e + 02	−2.0043e + 02	2.3965e + 01
		SVR-DE	1.79e + 02	2.01e + 02	2.17e + 02	1.14e + 01
		SVC-DE	1.84e + 02	2.09e + 02	2.27e + 02	1.31e + 01
F10	−3.30e + 02	TLSAPSO	−2.6029e + 02	−1.9114e + 02	−1.1391e + 02	4.9602e + 01
		SVR-DE	1.80e + 02	2.15e + 02	2.34e + 02	1.29e + 01
		SVC-DE	1.93e + 02	2.15e + 02	2.38e + 02	1.37e + 01

number of samples. One consequence is that the optimum of the surrogate is different from that of the real fitness function. As a result, TLSAPSO assisted by a global surrogate only may fail to locate the real global optimum of unimodal problem. For example, it is difficult to build a correct global surrogate model for F5, especially in the early stage of the evolution, because its global optimum is on the boundary. In this case, the achieved performance enhancement by the global surrogate is less significant than on other unimodal functions, such as F1.

Comparing the results in Figs. 4 and 5 with those in Figs. 8 and 13, we can see that the contribution of the surrogates in FESPSO is similar to that in CPSO-L, because the fitness estimation strategy suggested in FESPSO is in some certain sense a local estimation strategy. Among the five unimodal test functions, TLSAPSO outperformed FESPSO on F1, F3 and F5. However, FESPSO performed better than TLSAPSO on F2. The three compared algorithms performed similarly on F4.

To further demonstrate the effectiveness of TLSAPSO, a second set of experiments has been conducted comparing TLSAPSO with state-of-the-art surrogate-assisted differential evolution (DE) algorithms (Lu et al. 2011), one is a regression-assisted DE and the other is a classification-assisted DE. DE is chosen for comparison here because DE has been demonstrated to be an efficient method for optimizing continuous multimodal function (Storn 1996). The parameter settings for TLSAPSO are the same as in the first set of experiments. The comparative results are shown in Table 5.

From Table 5, it can be clearly seen that for all multimodal problems, the optima found by TLSAPSO are much better than both surrogate-assisted DE algorithms presented in Lu et al. (2011), which further confirms that our TLSAPSO algorithm is highly suited for solving multimodal optimization problems. For unimodal problems, TLSAPSO can find better or competitive results compared to the surrogate-assisted DE algorithms except for F5, whose global optimum is located on the boundary of the search space. As previously, surrogates are less helpful for such problems.

5 Conclusion and future work

A TLSAPSO algorithm is proposed to solve computationally expensive optimization problems. In TLSAPSO, a global surrogate model is used to smooth out the local optimums of the original multimodal fitness function and a local surrogate modal is employed to achieve accurate local fitness estimations. The contributions of these surrogates to the performance improvement on uni- and multi-modal problems are empirically examined and the experimental results agree with our conjecture. Our experimental results show the effectiveness of the proposed TLSAPSO compared with the CPSO, FESPSO and two state-of-the-art surrogate-assisted DE algorithms, especially for multimodal problems.

However, much work remains for future study. For example, TLSAPSO contains a few parameters to be specified, including those for updating the RBFN models and for maintaining the databases. An optimal set up of these parameters may be challenging, although our results indicate that the performance of TLSAPSO is relatively insensitive to these parameters. Second, selecting data sampling for training the global model as well as the local surrogates is critical for the success of surrogate-assisted PSO algorithms. Therefore, integration of advanced learning techniques such as semi-supervised learning (Sun et al. 2013) into surrogate-assisted PSO is another promising topic of our future work.

Acknowledgments This work was supported in part by Youth Foundation of Shanxi Province of China under Grant No. 2011021019-3, the Doctoral Foundation of Taiyuan University of Science and Technology under Grant No. 20122010, and the State Key Laboratory of Software Engineering, Nanjing University, China, Project No. KFKT2013A05.

References

Abou El-Ela A, Fetouh T, Bishr M, Saleh R (2008) Power systems operation using particle swarm optimization technique. Electr Power Syst Res 78(11):1906–1913

Acar E, Rais-Rohani M (2009) Ensemble of metamodels with optimized weight factors. Struct Multidiscip Optim 37(3):279–294

Bird S, Li X (2010) Improving local convergence in particle swarms by fitness approximation using regression. In: Computational intelligence in expensive optimization problems. Adaptation learning and optimization, vol 2. Springer, Berlin, Heidelberg, New York, pp 265–293

Buche D, Schraudolph NN, Koumoutsakos P (2005) Accelerating evolutionary algorithms with gaussian process fitness function models. IEEE Trans Syst Man Cybern Part C Appl Rev 35(2):183–194

Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6(1):58–73

Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, pp 39–43

Eberhart RC, Shi Y (2000) Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 congress on evolutionary computation, vol 1, pp 84–88

Farina M (2002) A neural network based generalized response surface multiobjective. In: Proceedings of the 2002 congress on evolutionary computation, vol 1, pp 956–961

Fonseca LG, Lemonge AC, Barbosa HJ (2012) A study on fitness inheritance for enhanced efficiency in real-coded genetic algorithms. In: Proceedings of the 2012 IEEE congress on evolutionary computation (CEC), pp 1–8

Goel T, Haftka RT, Shyy W, Queipo NV (2007) Ensemble of surrogates. Struct Multidiscip Optim 33(3):199–216

He S, Prempain E, Wu Q (2004) An improved particle swarm optimizer for mechanical design optimization problems. Eng Optim 36(5):585–605

Hendtlass T (2007) Fitness estimation and the particle swarm optimisation algorithm. In: Proceedings of the IEEE congress on evolutionary computation, pp 4266–4272

Jin Y, Sendhoff B (2004) Reducing fitness evaluations using clustering techniques and neural network ensembles. In: Proceedings of the genetic and evolutionary computation (GECCO 2004). Lecture notes in computer science, vol 3102. Springer, New York, pp 688–699

Jin Y, Olhofer M, Sendhoff B (2002) A framework for evolutionary optimization with approximate fitness. IEEE Trans Evol Comput 6(5):481–494

Jin Y (2005) A comprehensive survey of fitness approximation in evolutionary computation. Soft Comput 9(1):3–12

Joseph VR, Hung Y, Sudjianto A (2008) Blind kriging: a new method for developing metamodels. J Mech Des 130(3):031102.1–031102.8

Kattan A, Galvan E (2012) Evolving radial basis function networks via gp for estimating fitness values using surrogate models. In: Proceedings of the 2012 IEEE congress on evolutionary computation (CEC), pp 1–7

Lian Y, Liou M-S (2005) Multiobjective optimization using coupled response surface model. AIAA J 43(6):1316–1325

Lim D, Jin Y, Ong Y-S, Sendhoff B (2010) Generalizing surrogate-assisted evolutionary computation. IEEE Trans Evol Comput 14(3):329–355

Liu B, Zhang Q, Gielen G (2014) A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems. IEEE Trans Evol Comput 18(2):180–192

Lu J, Li B, Jin Y (2013) An evolution strategy assisted by an ensemble of local gaussian process models. In: Proceedings of the fifteenth annual conference on genetic and evolutionary computation conference, ACM, pp 447–454

Lu X, Tang K, Yao X (2011) Classification-assisted differential evolution for computationally expensive problems. In: Proceedings of the 2011 IEEE congress on evolutionary computation (CEC), pp 1986–1993

Ong YS, Nair PB, Keane AJ, Wong KW (2004) Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems. In: Jin Y (ed) Knowledge incorporation in evolutionary computation. Studies in fuzziness and soft computing series. Springer, pp 307–331

Ong YS, Nair PB, Keane AJ (2003) Evolutionary optimization of computationally expensive problems via surrogate modeling. AIAA J 41(4):687–696

Ong Y-S, Nair PB, Lum KY (2006) Max-min surrogate-assisted evolutionary algorithm for robust design. IEEE Trans Evol Comput 10(4):392–404

Parno M, Hemker T, Fowler K (2012) Applicability of surrogates to improve efficiency of particle swarm optimization for simulation-based problems. Eng Optim 44(5):521–535

Praveen C, Duvigneau R (2009) Low cost pso using metamodels and inexact preevaluation: application to aerodynamic shape design. Comput Methods Appl Mech Eng 198(9):1087–1096

Ratle A (2001) Kriging as a surrogate fitness landscape in evolutionary optimization. AI EDAM 15(01):37–49

Regis RG (2014) Particle swarm with radial basis function surrogates for expensive blackbox optimization. J Comput Sci 5(1):12–23

Reyes-Sierra M, Coello CAC (2005) A study of fitness inheritance and approximation techniques for multi-objective particle swarm optimization. In: Proceedings of the 2005 IEEE congress on evolutionary computation, vol 1, pp 65–72

Sha D, Hsu C-Y (2008) A new particle swarm optimization for the open shop scheduling problem. Comput Oper Res 35(10):3243–3261

Shi Y, Eberhart R (1998) A modified particle swarm optimizer. In: Proceedings of the 1998 IEEE international conference on evolutionary computation, IEEE world congress on computational intelligence, pp 69–73

Smith RE, Dike BA, Stegmann SA (1995) Fitness inheritance in genetic algorithms. In: Proceedings of the 1995 ACM symposium on applied computing, pp 345–350

Storn R (1996) On the usage of differential evolution for function optimization. In: Proceedings of the 1996 biennial conference of the North American Fuzzy Information Processing Society, NAFIPS, pp 519–523

Suganthan PN, Hansen N, Liang JJ, Deb K, Chen Y, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the cec 2005 special session on realparameter optimization. Technical Report, Nanyang Technological University, Singapore and KanGAL Report #2005005, IIT Kanpur, India

Sun C, Zeng J, Pan J, Jin Y (2013) Similarity-based evolution control for fitness estimation in particle swarm optimization. In: Proceedings of the 2013 IEEE symposium on computational intelligence in dynamic and uncertain environments (CIDUE), pp 1–8

Sun C, Zeng J, Pan J, Xue S, Jin Y (2012) A new fitness estimation strategy for particle swarm optimization. Inf Sci 221:355–370

Sun X, Gong D, Jin Y, Chen S (2013) A new surrogate-assisted interactive genetic algorithm with weighted semisupervised learning. IEEE Trans Cybern 43(2):685–698

Tang Y, Chen J, Wei J (2013) A surrogate-based particle swarm optimization algorithm for solving optimization problems with expensive black box functions. Eng Optim 45(5):557–576

Tenne Y, Armfield SW (2009) A framework for memetic optimization using variable global and local surrogate models. Soft Comput 13(8–9):781–793

Ulmer H, Streichert F, Zell A (2003) Evolution strategies assisted by Gaussian processes with improved preselection criterion. In: Proceedings of the 2003 congress on evolutionary computation (CEC'03), vol 1, pp 692–699

Zhou Z, Ong YS, Nguyen MH, Lim D (2005) A study on polynomial regression and gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm. In: Proceedings of the 2005 IEEE congress on evolutionary computation, vol 3, pp 2832–2839

Zhou Z, Ong YS, Nair PB, Keane AJ, Lum KY (2007) Combining global and local surrogate models to accelerate evolutionary optimization. IEEE Trans Syst Man Cybern Part C Appl Rev 37(1):66–76