Initialization and Displacement of the Particles in TRIBES, a Parameter-Free Particle Swarm Optimization Algorithm

Yann Cooren, Maurice Clerc, and Patrick Siarry

Laboratoire Images, Signaux et Systèmes Intelligents, LiSSi, E.A. 3956 Université de Paris XII, 61 avenue du Général de Gaulle, 94010 Créteil, France
{cooren,siarry}@univ-paris12.fr, maurice.clerc@writeme.com

Summary. This chapter presents two ways of improvement for TRIBES, a parameter-free Particle Swarm Optimization (PSO) algorithm. PSO requires the tuning of a set of parameters, and the performance of the algorithm is strongly linked to the values given to the parameter set. However, finding the optimal set of parameters is a very hard and time consuming problem. So, Clerc worked out TRIBES, a totally adaptive algorithm that avoids parameter fitting. Experimental results are encouraging but are still worse than many algorithms. The purpose of this chapter is to demonstrate how TRIBES can be improved by choosing a new way of initialization of the particles and by hybridizing it with an Estimation of Distribution Algorithm (EDA). These two improvements aim at allowing the algorithm to explore as widely as possible the search space and avoid a premature convergence in a local optimum. Obtained results show that, compared to other algorithms, the proposed algorithm gives results either equal or better.

Keywords: Particle swarm optimization, estimation of distribution algorithm, continuous optimization.

1 Introduction

Particle Swarm Optimization (PSO) is a population-based optimization technique proposed by Kennedy and Eberhart in 1995 [6]. Like ant colony algorithms or genetic algorithms, PSO is a biologically-inspired metaheuristic. The method is inspired from the social behavior of animals evolving in swarms, like birds or fishes. The principle is to use collaboration among a population of simple search agents to find the optimum in a function space. More precisely, a simple agent has basically the knowledge of the characteristics of its surroundings but, by communicating with other particles of the swarm, it also has a global knowledge of the search space, as it can be seen in a fish school which tries to find something to eat. PSO is also a particular case in the metaheuristic field because PSO was directly designed for solving continuous problems. This point has its importance because most of the applications deal with continuous problems. A state of the art of PSO and all the concepts which are linked to it is available in [2].

Like other metaheuristics, PSO shows the drawback of comprising many parameters which have to be defined. The difficulty is that the performances of the algorithm are strongly linked to the values given to these parameters. Such a remark implies that it is difficult and time consuming to find the optimal combination of parameter values. Moreover, in real problems, the parameters are often correlated, which makes the choice of parameters harder. So, researches are led to reduce the number of "free" parameters. The aim is to design algorithms which are as efficient as classical algorithms, but with a lower number of parameters. Such algorithms can be called "adaptive algorithms", because information gradually collected during the optimization process are used to compute the values of the parameters. The algorithm is "adaptive" in the way that its behavior, characterized by the values given to the parameters, is evolving all along the process. Several adaptive methods already exist for PSO [15,16,18]. All these algorithms are adaptive but not completely, i.e. there are still parameters to define, so the problem is admittedly easier, but still existing. The ideal would be to design a parameter-free algorithm. A parameter-free algorithm acts as a "black-box" and the user has just to define his problem and to run the process; no parameter has to be defined. Such an algorithm exists among genetic algorithms [10]. Clerc has created a parameter-free algorithm for PSO called TRIBES [3, 4]. In this chapter, we will describe the rules of adaptation which permit to avoid the definition of parameters in TRIBES and two ways of improvement will be explored.

Section 2 is dedicated to a brief presentation of the basic PSO algorithm. TRIBES is described in Section 3. In Section 4, we propose a new method to initialize TRIBES. A discussion of the strategies of displacement is presented in Section 5. Some numerical results are shown in Section 6. Finally, we conclude in Section 7.

2 Basic Particle Swarm Optimization

PSO is easy to be coded and implemented. In addition, its simplicity implies that the algorithm is inexpensive in terms of memory requirement and CPU time [4]. All these characteristics have made the popularity of PSO in the field of metaheuristics.

PSO starts with a random initialization of a swarm of particles in the search space. Each particle is modelled by its position in the search space and its velocity. At each time step, all particles adjust their positions and velocities, thus their trajectories, according to their best locations and the location of the best particle of the swarm, in the global version of the algorithm, or of their neighbors, in the local version. Here appears the social behavior of the particles. Indeed, each individual is influenced not only by its own experience but also by the experience of other particles.

In a D-dimensional search space, the position and the velocity of the ith particle can be represented as $X_i = [x_{i,1}, \cdots, x_{i,D}]$ and $V_i = [v_{i,1}, \cdots, v_{i,D}]$ respectively. Each particle has its own best location $p_i = [p_{i,1}, \cdots, p_{i,D}]$, which corresponds to the best location reached by the ith particle at time t. The global best location is named $g = [g_i, \cdots, g_D]$, which represents the best location reached by the entire swarm. From time t to time t+1, each velocity is updated using the following equation:

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1 r_1 (p_{i,j}(t) - v_{i,j}(t)) + c_2 r_2 (g_j(t) - v_{i,j}(t)) \tag{1}$$

where w is a constant called *inertia factor*, c_1 and c_2 are constants called *acceleration coefficients*, r_1 and r_2 are two independent random numbers uniformly distributed in [0,1]. Generally, the value of each component in V_i can be clamped in a range $[-V_{\max}, V_{\max}]$ to control excessive roaming of the particles outside the search space. If the computed velocity leads one particle out of the search space, two methods can be used:

- the particle goes out of the search space but its fitness is not computed.
- the particle is brought back in the search space either on the nearest bound or by applying a multiplicative coefficient chosen in]-1,0[.

The computation of the position at time $t+1$ is derived from Eq.(1) using:

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \tag{2}$$

The inertia weight w controls the impact of the previous velocity on the current one, so it ensures the diversity of the swarm, which is the main means to avoid the stagnation of particles at local optima. In the same way, c_1 controls the attitude of the particle of searching around its best location and c_2 controls the influence of the swarm on the particle's behavior. To summarize, we can say that w controls the diversification feature of the algorithm and c_1 and c_2 the intensification feature of the algorithm. In [5], Clerc and al show that the convergence of PSO may be insured by the use of a constriction factor. Using the constriction factor emancipates us to define $V_{\max}$. In this case, Eq (1) becomes:

$$v_{i,j}(t+1) = K\left(v_{i,j}(t) + \phi_1 r_1 (p_{i,j}(t) - v_{i,j}(t)) + \phi_2 r_2 (g_j(t) - v_{i,j}(t))\right) \tag{3}$$

with:

$$K = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|},\ \phi = \phi_1 + \phi_2,\ \phi > 4 \tag{4}$$

The convergence characteristic of the system can be controlled by ϕ. Namely, Clerc and al. [5] found that the system behavior can be controlled so that the system behavior has the following rules:

- the system does not diverge in a real value region and finally can converge,
- the system can search different regions efficiently by avoiding premature convergence.

Unlike other evolutionary computation methods, constricted PSO ensures the convergence of the search procedure based on the mathematical theory. The standard PSO procedure can be summarized in the algorithm in Figure 1.

Generally, the stopping criterion is either a predefined acceptable error or a maximum "reasonable" number of evaluations of the objective function.

```
**Initialize** a population of particles with random positions and velocities.
**Evaluate** the objective function for each particle and compute $g$.
For each individual, $p_i$ is **initialized** at $X_i$.
**Do**
  Update the velocities and the positions of the particles.
  Evaluate the objective function for each individual.
  Compute the new $p_i$ and $g$.
**While** the stopping criterion is not met
```

Fig. 1. Standard PSO procedure

3 TRIBES, a Parameter-Free PSO Algorithm

Like many other metaheuristics, PSO shows the drawback of having too many parameters which must be set by the user. According to the values given to these parameters the algorithm is more or less efficient. A first approach to adaptive PSO was proposed by Ye [16], whose method looks for inactive particles and replaces them by new particles, more able to explore new areas of the search space. Zhang et al. [18] proposed to modify swarm's size, constriction factor or neighborhood size through the use of an improvement threshold. Yasuda [15] worked out an algorithm in which parameters are defined according to the velocity information of the swarm. But the first parameter-free algorithm, called TRIBES, was proposed by Clerc [3]. TRIBES is an adaptive algorithm whose parameters change according to the swarm behavior. In TRIBES, the user only has to define the objective function and the stopping criterion. The method incorporates rules defining how the structure of the swarm must be modified and also how a given particle must behave, according to the information gradually collected during the optimization process.

3.1 Swarm, Tribes and Communication

PSO is based on the social behavior of animals evolving in swarms. Each individual of the swarm knows the direction of displacement and the velocity of its neighbors in the swarm and uses this information to decide its own direction of displacement and its own velocity. Considering that the swarm is an interconnected network, information collected by one of the individuals is propagated in the entire swarm. So, all the individuals modify their behavior according to the new interesting information. This implies a "global" behavior of the swarm, which allows the swarm to find regions of interest in the search space. These considerations form the framework of Standard PSO.

However, it can also be observed in real life that a swarm can be divided in "tribes" of individuals. Here, the behavior of the swarm is different from the one explained before. Each tribe acts as an independent swarm, i.e. each tribe has its own "global behavior" and explores a particular region of the search space. In addition to that, all the tribes exchange information about regions they are exploring.

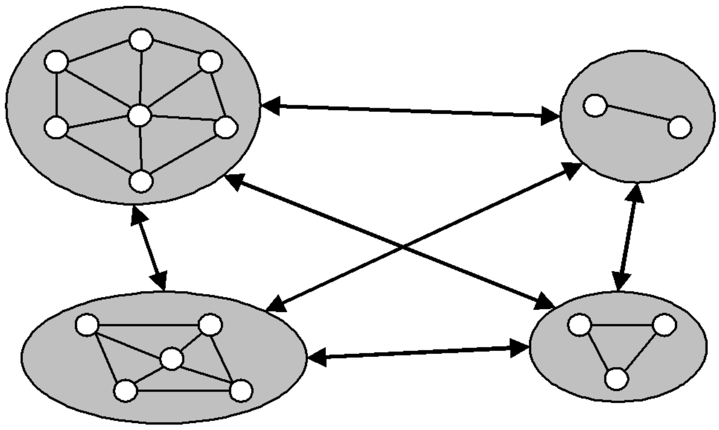

Fig. 2. Intra-tribe and inter-tribes communications

So, the swarm is an interconnected network of tribes, which are themselves interconnected networks of individuals. This implies two different types of communication: intra-tribe communication and inter-tribes communication. These considerations form the framework of TRIBES. In Figure 2, an example of a swarm of 17 particles (white spots) divided in 4 tribes is shown. Arrows symbolize inter-tribes communications and lines symbolize intra-tribe communications.

In TRIBES, the swarm is structured in different tribes of variable size. The aim is to simultaneously explore several promising areas, generally local optima, and to exchange results between all the tribes in view of finding the global optimum.

Each tribe is composed of a variable number of particles. Relations between particles in a tribe are similar to those defined in basic PSO. More precisely, each particle of the tribe stores the best location it has met and knows the best (and the worst) particle of the tribe, i.e. the particle which has met the best (and the worst) location in the search space. This is intra-tribe communication.

Even if each tribe is able to find a local optimum, a global decision must be taken to decide which of these optima is the best one. So, each tribe is linked to all the others to inform them on the best location found by its best particle. This is inter-tribes communication.

TRIBES is an adaptive algorithm, so the swarm must be generated and modified automatically, by means of creation, evolution, and removal of the tribes.

3.2 Structural Adaptations

Setting rules to modify the swarm's structure implies the definition of a quality qualifier for each particle and likewise for the tribes. In the case of particles, it is known that each particle has a current position and a best position. So, a particle is said to be a *good* particle if it has just improved its best performance, *neutral* if not. In addition to this qualitative (because not relative to values,

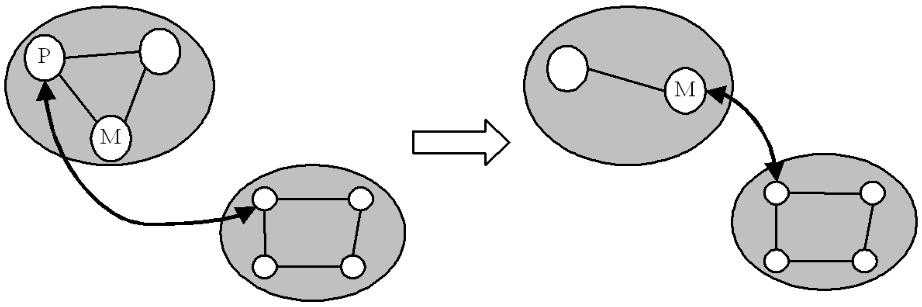

Fig. 3. Removal of a particle from a multiparticle tribe

but to improvement) qualification, the *best* and the *worst* particles are defined within the tribe framework.

In addition, good and bad statuses are also defined for the tribes. These statuses are related to the amount of good particles inside the tribes. It is postulated that: "The larger the number of good particles in the tribe, the better the tribe itself". In practice, a random integer number p between 0 and the swarm's size is generated according to a uniform distribution. Then, if the number of good particles in the tribe is strictly larger than p, the tribe is said *good*, if not, the tribe is said *bad*.

These qualifiers allow us to define the two following rules.

Removal of a Particle

In most common uses of PSO, the most time consuming part of the algorithm is the objective function evaluation. So, it is interesting to carry out the least number of evaluations of the objective function. Consequently, it will be tried to remove a particle of the swarm as soon as possible, on condition that the removal does not affect the final result. That is why a removal of particle must occur in a good tribe and the removed particle is obviously the worst. In Figure 3, the particle P is the worst of its tribe and the tribe was declared good. In this case P is removed and the redistribution of its external links (here, only one symmetrical link) is done on M, the best particle of the tribe. The information links that each particle has with itself are not represented, because they do not play any role here.

In the case of a monoparticle tribe, the removal is only made if one of the "informers", i.e. a particle of another tribe by which the inter-tribes communication is made, has a better performance (see Figure 4). This ensures us to keep the better quality of information. In Figure 4, the monoparticle tribe was declared good, thus the single particle P, which is necessarily the worst of the tribe, even if it is at the same time good, should be removed. But it will be removed only if its best external informer M_P is better than itself. The assumption is indeed that the information carried by P is then less valuable than that carried by M_P.

The removal of a particle implies a change in the information network. All the particles linked to the removed particle are redirected to the best particle of

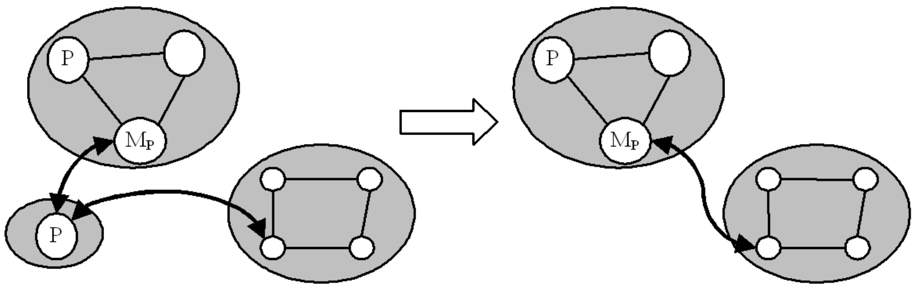

Fig. 4. Removal of a particle from a multiparticle tribe

the removed particle's tribe. In the particular case of a monoparticle tribe, all these links are redirected to the best informer of the removed particle, because removing the particle implies removing the tribe.

Obviously, these structural adaptations must not occur at each iteration, because time must be let for the information propagation. In practice, if NL is information links number at the moment of the last adaptation, the next adaptation will occur after NL/2 iterations.

Generation of a Particle

The process of adding a particle is quite similar to the removal. A bad tribe generates particles which will form a new tribe. The bad tribe will keep the contact with the new tribe and will try to use it to improve its best location.

Three types of particles are generated:

- Free particle: The particle is randomly generated according to a uniform distribution in the whole search space, on a side of the search space or on a vertex of the search space. The idea is to rely on the future course of the particle to cross a promising area.
- Confined particle: If x is the best particle of the generating tribe and i_x the best informer of x, p_x and p_{i_x} are the best locations of x and i_x. The new particle will be generated in the D-sphere of center p_{i_x} and radius $||p_x - p_{i_x}||$. The idea is here to intensify research inside an interesting area.
- Isolated particle: The particle is generated in the biggest "terra incognita", i.e. as far as possible from the existing particles and from the boundaries of the search space. The idea is to explore areas which have not been explored yet and, then, to discover new regions of interest. Figure 5 shows an example of where a "terra incognita" can be located. The * symbolize the particles and the gray regions indicate possible "terra incognita".

Swarm Evolution

At the beginning, the swarm is composed of only one particle which represents a single tribe. If, at the first iteration, this particle does not improve its location, a new one is created, forming a second tribe. At the second iteration, the same process is applied and so on.

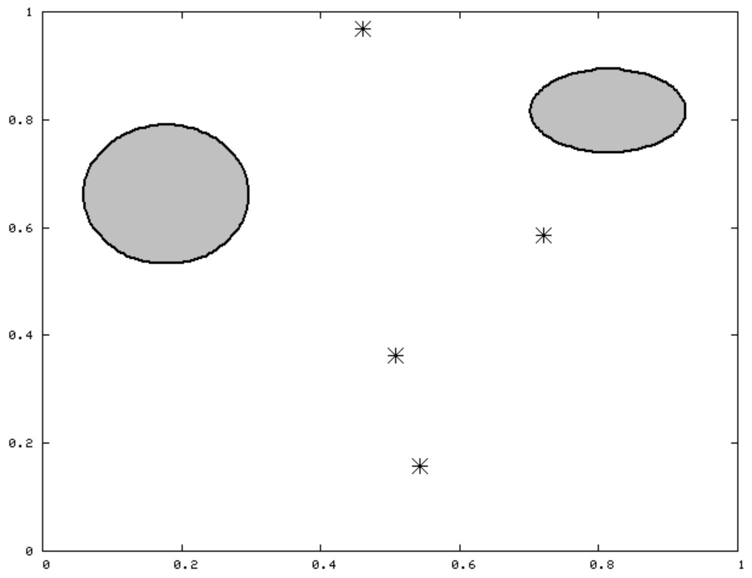

Fig. 5. Location of possible "terra incognita"

The swarm's size will grow up until promising areas are found. The more the swarm grows, the longer the time between two adaptations will be. By this way, the swarm's exploratory capacity will grow up but the adaptations will be more and more spaced in time. Therefore, the swarm has more and more chances to find a good solution between two adaptations.

Contrarily, once a promising area is found, each tribe will gradually remove its worst particle, possibly until it disappears. Ideally, when convergence is confirmed, each tribe will be reduced to a single particle.

3.3 Behavioral Adaptations

In the previous section, the first way of adaptation of the algorithm was described. The second way in view of adapting the swarm to the results found by the particles is to choose the strategy of displacement of each particle according to its recent past. As in the case of the evolution of tribes, it will enable a particle with a good behavior to have an exploration of greater scope, with a special strategy for very good particles, which can be compared to a local search. According to this postulate, the algorithm will choose to call the best displacement's strategy in view of moving the particle to the best possible location it can reach.

There are three possibilities of variation for a particle: deterioration, status quo and improvement, i.e. the current location of the particle is worse, equal or better than its last position. These three statuses are denoted by the following symbols: - for deterioration, = for status quo and + for improvement. The history of a particle includes the two last variations of its performance. For example, an improvement followed by a deterioration is denoted by (+ -). So, there are nine possibilities of history. The strategy of displacement of a particle will be determined by its couple of variations. The different strategies of displacement will be discussed in Section 5.

To sum up, it can be said that TRIBES is an algorithm which tries to solve one of the main problems of metaheuristics: the fitting of parameters. By adapting the swarm's form and particles' strategies of displacement, TRIBES frees users of defining parameters and acts as a "black box". The particles use their own history and swarm's history to decide how they must move and how the swarm must be organized in view of approaching as efficiently as possible the global optimum. The algorithm in Figure 6 shows a pseudo-code which summarizes TRIBES process. g is the best location reached by the swarm and the p's are the best locations for each particle. NL is the number of information links at the last swarm's adaptation and n is the number of iterations since the last swarm's adaptation.

```
Initialize a population of particles with random positions and velocities.
Evaluate the objective function for each particle and compute $g$.
For each individual, $p_i$ is initialized at $X_i$.
Do
  Determination of statuses for each particle.
  Choice of the displacement strategies.
  Update the velocities and the positions of the particles.
  Evaluate the objective function for each individual.
  Compute the new pi and g.
  If $n < NL$
      Determination of tribes qualities
      Swarm's adaptations
      Computation of NL
  End if
While the stopping criterion is not met
```

Fig. 6. TRIBES procedure

4 Initialization of TRIBES

The efficiency of every PSO-inspired algorithm is linked to the initialization of the particles. In [1], it is proved that, in basic PSO, the position of a particle at the time step t can be decomposed in two vectors, one which only depends on the initial configuration and one which does not depend on the initial configuration. The trajectory of a given particle is then linearly dependent from its initial position and its initial velocity. So, it clearly appears here that a good choice of initial positions and velocities can lead to better results. The ideal case would be to assess the initial points in the way that the search space is explored as widely as possible by the trajectories of the particles.

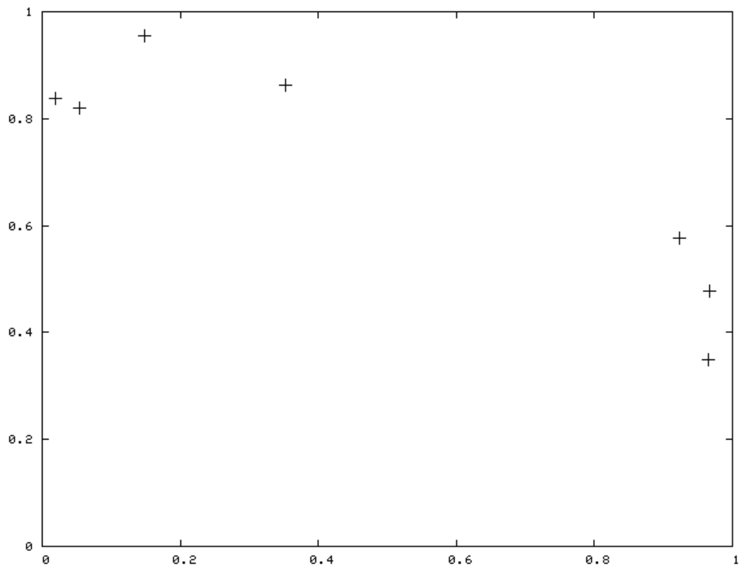

Fig. 7. Random initialization

Commonly, particles are randomly initialized in the search space. But, such an initialization can lead to a bad diversity of the particles. On Figure 7, a random initialization of seven particles in dimension D=2 is shown. It can be seen that the particles are confined in the lower-right part of the search space and in the upper-left part. In this way, the particles would first explore the upper-left and the lower-right corners of the search space and, then, there is a possibility that they would be trapped in a local optimum situated in this region without having explored the other regions of the search space.

In view of avoiding this problem, a new way of initialization is proposed. The idea is to fill as widely as possible the search space. To summarize, the particles will be initialized so that each particle will be as far as possible from the others and as far as possible from the boundaries of the search space.

In dimension D, TRIBES will be initialized with $D+1$ particles. The initialization is made using a standard particle swarm optimization using the objective function of Eq.(5):

$$f = \sum_{i=1}^{D+1} \sum_{j \neq i} \frac{1}{d_{ij}} + \sum_{i=1}^{D+1} \frac{1}{\min_{d \in [1..D]}(d(x_i, bound_d))} \tag{5}$$

where d_{ij} is the distance between particle i and particle j and $d(x_i, bound_d)$ is the distance between particle i and the boundary of the dth dimension.

Figure 8 shows that this new way of initialization leads to a better diversity of the particles in the search space. The particles fill well the search space and no region will be preferred to others during the exploration process.

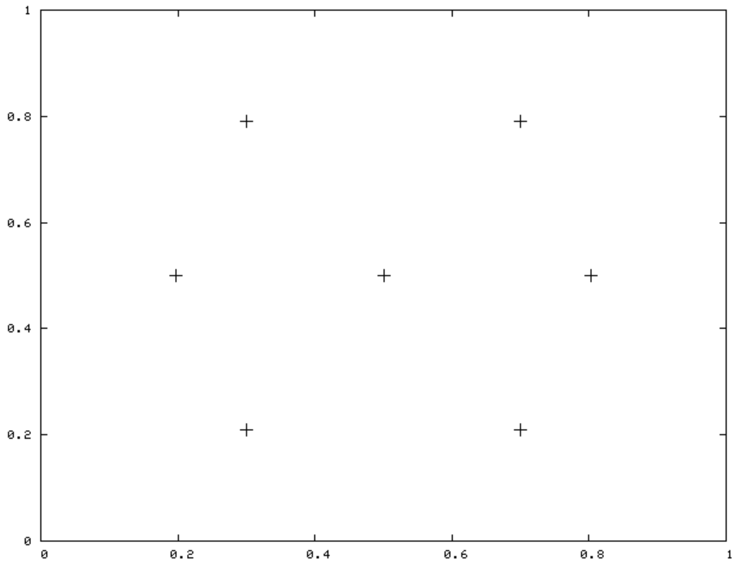

Fig. 8. Regular initialization

5 Strategies of Displacement

It was seen in Section 4 that the way of initializing the particles has its importance in view of making PSO as effective as possible. Once the particles are initialized, it must be decided how they will move. In basic PSO, the strategy of displacement is the same for each particle. Eqs.(1,2) model this strategy.

In TRIBES, the strategy of displacement is different for each particle and can be modified at each time step. For a given particle, the choice of its strategy of displacement is made according to its recent history. In this section, the original strategies defined by Clerc in [4] are exposed and a new strategy is defined.

5.1 Basic Strategies of TRIBES

It was said in Section 3.3 that the strategy of displacement of a given particle depends on its two last variations. So, there are nine possibilities of history. Clerc [4] gathered them in three groups. So, only three strategies are needed. The three used strategies are the following:

- Pivot strategy: This method is inspired from [11]. Let us denote by p the best location of the particle, g the best position of the informers of the particle and f the objective function. The movement is done as follows:

$$X = c_1 U(H_p) + c_2 U(H_g) \tag{6}$$

  with $c_1 = f(p)/(f(p) + f(g))$, $c_2 = f(g)/(f(p) + f(g))$, $U(H_p)$ a point uniformly chosen in the hyper-sphere of center p and radius $||p - g||$ and $U(H_g)$ a point uniformly chosen in the hyper-sphere of center g and radius $||p - g||$.

- Disturbed pivot strategy: Here we have

$$X = c_1U(H_p) + c_2U(H_g) \tag{7}$$

$$b = N(0, \frac{f(p) - f(g)}{f(p) + f(g)} \tag{8}$$

$$X_j = (1 + b)X_j \tag{9}$$

- Local strategy by independent gaussians: If g is the best particle of the swarm, we have

$$X_j = g_j + N(g_j - X_j, ||g_j - X_j||) \tag{10}$$

where $N(g_j - X_j, ||g_j - X_j||)$ is a point randomly chosen with a monodimensional gaussian distribution of mean $g_j - X_j$ and standard deviation $||gj - Xj||$.

So, (= +) and (+ +) represent the best particles and it will be chosen for them a local strategy by "independent gaussians". (+ =) and (- +) represent the neither bad nor good particles and it will be chosen a "disturbed pivot strategy". At last, (- -), (= -), (+ -), (- =) and (= =) represent feeble particles. A "pivot strategy" is chosen for them. If the movement implies one particle to go out the search space, the particle will be brought back to its closest position in the search space. This is particle's confinement.

It can be seen that the strategies of displacement defined in this paragraph are different from the one of Standard PSO, especially because there is no need to define velocity vectors, but the philosophy is almost the same: a particle moves according to its own best performance and according to the best performance of the swarm.

5.2 A New Strategy of Displacement

In the previous paragraph, the three strategies of displacement employed in the first version of TRIBES were defined. Results obtained with the original algorithm can be found in [19]. In view of evaluating the efficiency of the strategies of displacement defined above, a study of the behavior of the particles must be done. No complete theoretical study already exists because the interactions between particles made the problem very hard to solve although a few theoretical results about the dynamics of PSO [5, 13, 14] are available in the literature. In consequence, only experimentation can inform us about the dynamics of the particles during the process.

Figure 9 shows us examples of "convergence graphs" of TRIBES. Such graph shows the median performance of the total runs, here 25 runs, as a function of the number of evaluations of the objective function (Fes). A semilog scale is used. These graphs are made with Griewank (F7), Rastrigin (F9) and Weierstrass (F11) functions [12] in dimension D=10. It can be seen that TRIBES converges quickly at the beginning of the process and seems to stagnate after. It can

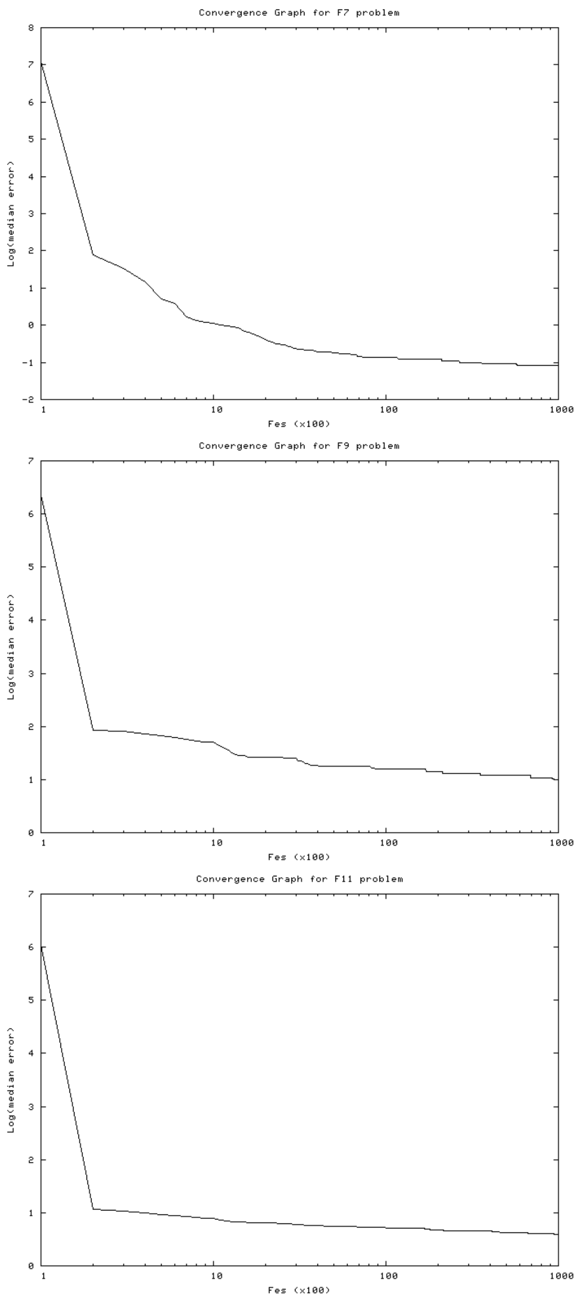

Fig. 9. Convergence graphs for Griewank (top), Rastrigin (middle) and Weierstrass (bottom) functions

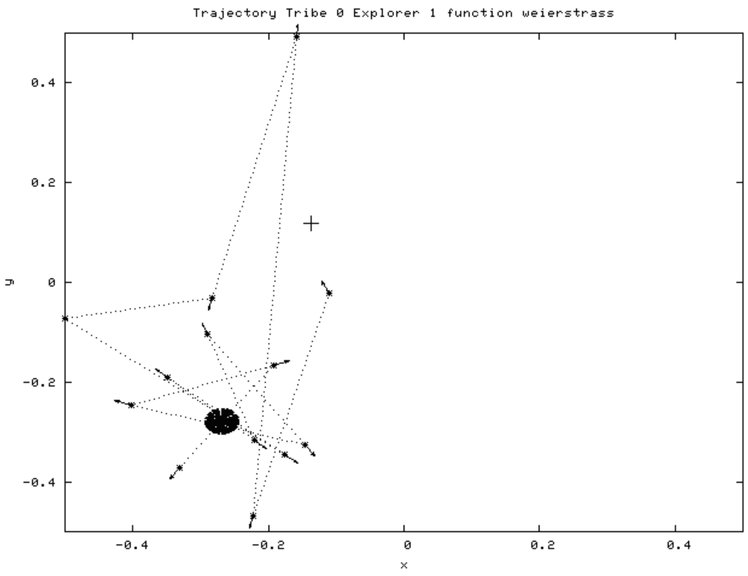

Fig. 10. Trajectory of a particle. The cross indicates the optimum.

```
**Initialize** the population $X^0$
**Build** the distribution $P^0$
$X^t = X^0$
$P^t = P^0$
**Do**
  **Select** a subgroup $X^t_{sub}$ of $X^t$
  **Build** $P^{t+1}$ according to $X^t_{sub}$
  **Sample** $P^{t+1}$ to create $X^{t+1}_{offspring}$
  **Replace** individuals of $X^t$ by individuals of $X^{t+1}_{offspring}$ for creating $X^{t+1}$
  $t = t + 1$
**While** the stopping criterion is not met
```

Fig. 11. Principle of an EDA

be concluded that the particles converge quickly to a local optimum and do not manage to escape from it. This fact is a problem common to every PSO-inspired algorithm. Considering that, in TRIBES, the position of a given particle is randomly chosen using two hyperspheres of centers g and p, the particle is attracted towards these two points. This implies that, if g or p is a local optimum, the particle would be attracted to the "valley" of this optimum, move around it and, so, it would not be possible for the particle to improve its performance. Figure 9 shows an example of this behavior. Figure 9 represents the trajectory of a particle for Weierstrass 2D problem [12]. Arrows are the velocities of the particle at each time step and the cross symbolizes the location of the global optimum. It is clear that the particle is trapped in a local optimum and moves around it. The observation of the velocity vectors confirms this remark.

Previous remarks imply that TRIBES is defective in matter of diversification, i.e. the search space is not widely explored by the particles. The objective of the new strategy of displacement is to improve the diversification capacity of TRIBES. To reach this goal, an Estimation of Distribution Algorithm (EDA) is used [7]. The principle of EDAs is to select a subgroup of a family of solutions, build a probabilistic model of this subgroup and construct new solutions by randomly sampling the constructed distribution. First, a family X^0 of solutions is randomly generated and its probabilistic model P^0 is constructed. For a given time t, the main loop is composed of four steps. First, a subgroup X^t_{sub} of X^t is chosen using a predefined criterion. Then, according to the element of X^t_{sub}, the probabilistic model P^{t+1} is constructed. At this moment, P^{t+1} is sampled to give the new family of solutions X^{t+1} offspring. Finally, individuals of X^t are replaced by individuals of X^{t+1} offspring. The process is iterated until the predefined stopping criterion is reached. Many different probabilistic models can be used considering or not that the variables are correlated [7]. Algorithm 11 summarizes this process.

Introducing a new strategy of displacement based on EDA can permit us to improve the diversification process of TRIBES. Indeed, the displacement of a given particle will not be driven only by three positions (the current position of the particle, p and g) but by a sub-family of the swarm. So, premature convergence, caused by the stagnation of p and g, is avoided. Moreover, sampling a probabilistic model implies that all the positions of the search space can be reached with a non-zero probability and, then, gives the possibility to a particle to escape from a local optimum in which it was trapped. By this way, the search space can be more widely explored.

The problem is now to choose the appropriate family of particles in view of building the distribution of probability. In a PSO-inspired algorithm like TRIBES, the most obvious choice is to choose the best position of each particle i.e. the p_i. Dimensions are supposed independent, thus a monodimensional probabilistic model is computed for each dimension of the search space. If, at the current time step, the size of the swarm is N and d is the current dimension, the distribution of probability is supposed normal and is modelled by its average and its standard deviation according to Eqs.(11, 12).

$$\mu_d = \frac{1}{N}\sum_{i=1}^{D} p_{i,D} \tag{11}$$

$$\sigma_d^2 = \frac{1}{N-1}\sum_{i=1}^{D} (p_{i,D} - \mu_d)^2 \tag{12}$$

Then, the new coordinate of the particle for the dimension d is randomly chosen according to the normal distribution of average μ_d and standard deviation σ_d.

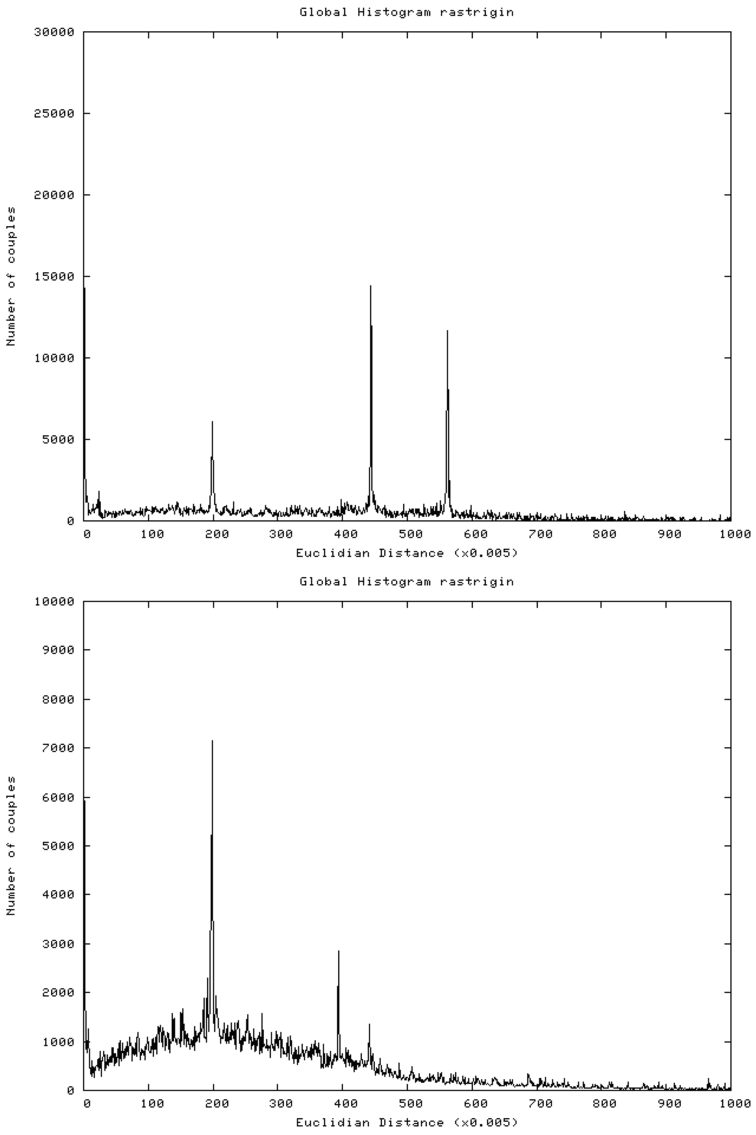

Fig. 12. Histograms of the distance to all reached positions

In real problems, it is current that the variables are correlated. In this case, a joint normal distribution is used instead of D monodimensional normal distributions.

Obviously, this method cannot be employed at any moment for each particle. It had been said above that there are nine possibilities of history for a particle. In basic TRIBES, histories are gathered in three groups. With the addition of our new strategy of displacement, a new group is created. The use of the new strategy of displacement is dedicated to the worst particles. So, particles with a history (- -), (- =) or (= -) will use the new strategy to compute their displacement.

Figure 12 shows histograms of the distances between all the positions reached by the particles for the Rastrigin 2D problem [12]. It can be seen that, with the utilization of the new strategy of displacement, the positions reached by the particles are less close than in the basic case. This remark implies than the diversification process is better with the utilization of the new strategy. In Figure 12(top), it can be seen that the histogram is mainly composed of three thin peaks. This implies that, at the end of the procedure, the swarm is composed of three tribes which have explored three different regions of the search space. The fact that the peaks are very thin implies that all the particles of a same tribe are concentrated on a local optimum and do not manage to escape from it. Contrarily, in Figure 9b, the histogram is also composed of three peaks but the peaks are, in this case, larger. This implies that the particles have explored more largely the search space and did not stay concentrated around a local optimum. So, it can be concluded that the diversification process of TRIBES is improved.

6 Numerical Results

The aim of this section is to compare Improved TRIBES, implemented with the regular initialization and the new strategy of displacement, with a Standard PSO algorithm [20], a simple continuous EDA [17], a real-coded Memetic Algorithm [8], a real-coded Differential Evolution Algorithm [9] and with Basic TRIBES. Tests are made in dimension D=10. For Standard PSO, the number of neighbors for each particle is 3, the size of the swarm is 20, $w = 1/(2\log 2)$ and $c_1 = c_2 = 1/2 + \log 2$. The process stops if the error is lower or equal to 10^{-6} for functions of Table 1 and 10^{-2} for functions of Table 2 or if the number of evaluations of the objective function exceeds 105. Functions used are extracted from the CEC'2005 benchmark functions set [12]. Table 1 and Table 2 give the mean error of each algorithm for 25 executions and the mean number of evaluations of the objective function, in brackets, respectively for unimodal functions and multimodal functions.

Table 1 and Table 2 show that Improved TRIBES, implemented with the regular initialization and the new strategy of displacement, gives competitive results. Ackley bounds function can be excluded from the comparison because it can be seen in Table 2 that all the algorithms are trapped in the same local optimum. Improved TRIBES solves 50% of the benchmark, the best algorithm being Differential Evolution with 60% of the benchmark solved.

Compared to Standard PSO, it can be seen that Improved TRIBES gives better results on most cases. Indeed, Standard PSO is better only on Schwefel and Schwefel noise, what can be explained by the simplicity of the function and by the fact that Standard PSO starts with more particles than Improved TRIBES. Improved TRIBES is better than TRIBES on 80% of the benchmark.

Table 1. Comparison between several algorithms for unimodal functions

	Sphere	Schwefel	Elliptic	Schwefel noise	Schwefel bounds
Standard PSO	0.00 (3375)	0.00 (8300)	71.28e3 (100000)	0.00 (11562)	18.86 (67610)
Simple EDA	0.00 (28732)	0.00 (28916)	0.00 (32922)	0.00 (32636)	233.50 (100000)
Memetic algorithm	0.00 (11737)	0.00 (36598)	4.77e4 (100000)	0.00 (71563)	0.02 (100000)
Differential Evolution	0.00 (7120)	0.00 (21800)	0.00 (8970)	0.00 (26100)	0.00 (25300)
Basic TRIBES	0.00 (1856)	0.00 (10452)	18.77e3 (100000)	0.00 (18966)	1.74 (60409)
Improved TRIBES	0.00 (1521)	0.00 (12011)	13.11e4 (100000)	0.00 (23783)	9.07e-4 (80832)

Table 2. Comparison between several algorithms for multimodal functions

	Rosen-brock	Grie-wank	Ackley bounds	Rastrigin	Rastrigin rotated	Weier-strass
Standard PSO	1.88 (100000)	0.08 (100000)	20.11 (100000)	4.02 (100000)	10.18 (100000)	4.72 (100000)
Simple EDA	0.00 (43500)	0.53 (10000)	20.33 (100000)	32.28 (100000)	32.28 (100000)	8.27 (100000)
Memetic algorithm	1.48 (100000)	0.19 (100000)	20.19 (100000)	0.43 (100000)	5.63 (100000)	4.55 (100000)
Differential Evolution	0.00 (26100)	0.15 (100000)	20.40 (100000)	0.00 (5110)	36.00 (100000)	4.67 (100000)
Basic TRIBES	0.06 (100000)	0.07 (100000)	20.32 (100000)	8.55 (100000)	12.11 (100000)	5.68 (100000)
Improved TRIBES	0.12 (100000)	0.02 (100000)	20.35 (100000)	0.19 (72584)	8.55 (100000)	3.55 (100000)

Compared to Basic TRIBES, Improved TRIBES is better excepted on Schwefel, Schwefel noise and Rosenbrock functions. On Schwefel and Schwefel noise, it can be deduced that the new strategy of displacement slows a little bit the algorithm mainly because the diversification process is larger. On Rosenbrock function, the result is a little better in favor of Basic TRIBES but the difference is not significant.

Table 1 shows that, on unimodal problems, Improved TRIBES is faster than non-PSO methods except on Elliptic problem. PSO-inspired methods fail totally on this problem whereas Simple EDA and Differential Evolution solve it quite simply. Globally, PSO-inspired methods seem to be more efficient, i.e. quicker, than the others on simple problems easy to solve (ex: Sphere, Schwefel,...).

Table 3. Time comparison between Basic TRIBES and Improved TRIBES for unimodal functions

	Sphere	Schwefel	Elliptic	Schwefel noise	Schwefel bounds
Basic TRIBES	12.72	32.12	21.96	24.18	65.7
Improved TRIBES	26.57	25.28	27.72	34.85	74.12

Table 4. Time comparison between Basic TRIBES and Improved TRIBES for multimodal functions

	Rosen-brock	Grie-wank	Ackley bounds	Rastrigin	Rastrigin rotated	Weier-strass
Basic TRIBES	26.5	24.48	36.6	22.14	32.11	123.55
Improved TRIBES	32.19	32.32	47.2	38.74	46.84	147.66

Table 2 does not permit to make strong conclusions. However, Improved TRIBES appears to be better than PSO-inspired methods and competitive with non-PSO methods.

Table 3 and Table 4 present execution times, in seconds, of 10^5 evaluations of the objective functions for Basic TRIBES and Improved TRIBES. Table 3 and Table 4 show that Improved TRIBES is slower than Basic TRIBES. This fact is easily understandable since the new strategy of displacement needs more computation time than the others. Moreover, the regular initialization process is also slower than a simple random initialization.

7 Conclusions

This chapter has shown that TRIBES uses structural and behavioral rules to avoid the fitting of parameters. Having no parameters implies no lost of time to fit the parameters, which is a very hard problem, but also implies to give more information to the particles, because the process is not "driven" by the values of the parameters.

Two methods are proposed to try to help the particles to explore as widely as possible the search space. The first one is a regular initialization of the particles, which aims at filling as regularly as possible the search space. The second one is a new strategy of displacement based on an estimation of distribution algorithm. This new strategy permits to the particles to have more various displacements, thus avoiding to be trapped into local optima.

Numerical results show that Improved TRIBES is equivalent or better than Standard PSO and Basic TRIBES. Improved TRIBES also appears to be competitive with non-PSO methods. Numerous improvements on TRIBES still are possible. The rules of adaptation are quite simple and use very few information compared with all the available information. New rules can be defined to try to have the best adaptation of the choices made to the specificity of the problem.

References

1. Campana, E.F., Fasano, G., Peri, D., Pinto, A.: Particle Swarm Optimization: Efficient Globally Convergent Modifications. In: III European Conference on Computational Mechanics, Solids and Coupled Problems in Engineering, Lisbon, Portugal, June 5-8 (2006)
2. Eberhart, R.C., Kennedy, J., Shi, Y.: Swarm Intelligence. In: Evolutionary Computation. Morgan Kaufmann, San Francisco (2001)
3. Clerc, M.: TRIBES - Un exemple d'optimisation par essaim particulaire sans paramtres de contrle. In: OEP 2003, Paris, France (2003)
4. Clerc, M.: Particle Swarm Optimization, International Scientific and Technical Encyclopedia (2006)
5. Clerc, M., Kennedy, J.: The particle swarm: explosion, stability, and convergence in multi-dimensional complex space. IEEE Transactions on Evolutionary Computation 6, 58–73 (2002)
6. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimisation. In: Proceedings of the IEEE International Conference On Neural Networks, WA, Australia, pp. 1942–1948 (1995)
7. Larrañaga, P., Lozano, J.A.: Estimation of Distribution Algorithms, a new tool for evolutionary computation. Kluwer Academic Publishers, Dordrecht (2001)
8. Molina, D., Herrera, F., Lozano, M.: Adaptive Local Search Parameters for Real-Coded Memetic Algorithms. In: Proceedings of the 2005 Conference on Evolutionary Computation, Edinburgh, Scotland, September 2-5, 2005, pp. 888–895 (2005)
9. Rnkknen, J., Kukkonen, S., Price, K.V.: Real-Parameter Optimization with Differential Evolution. In: Proceedings of the 2005 Conference on Evolutionary Computation, Edinburgh, Scotland, September 2-5, 2005, pp. 888–895 (2005)
10. Sawai, H., Adachi, S.: Genetic Algorithm Inspired by Gene Duplication. In: Proceedings of the 1999 Congress on Evolutionary Computing, Washington DC, USA, July 6-9, 1999, pp. 480–487 (1999)
11. Serra, P., Stanton, A.F., Kais, S.: Method for global optimization. Physical Review, tome 55, 1162–1165 (1997)
12. Suganthan, P.N., et al.: Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, Technical Report, Nanyang Technological University, Singapore, May 2005, AND KanGAL Report #2005005, IIT Kanpur, India (2005), `http://www.dcs.ex.ac.uk/~dwcorne/cec2005/`
13. Trelea, I.C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters 85, 317–325 (2003)
14. Van Den Bergh, F.: An Analysis of Particle Swarm Optimizers. Department of Computer Science, University of Pretoria, South Africa (2002)
15. Yasuda, K., Iwasaki, N.: Adaptive particle swarm optimization using velocity information of swarm. In: IEEE Conference on System, Man and Cybernetics, October 10-13, 2004, pp. 3475–3481. The Hague, Netherlands (2004)
16. Ye, X.F., Zhang, W.J., Yang, Z.L.: Adaptive Particle Swarm Optimization on Individual Level. In: International Conference on Signal Processing (ICSP), Beijing, China, August 26-30, 2002, pp. 1215–1218 (2002)

17. Yuan, B., Gallagher, M.: Experimental results for the special session on real-parameter optimization at CEC 2005, a simple continuous EDA. In: Proceedings of the 2005 Congress on Evolutionary Computation, Edinburgh, Scotland, September 2-5, 2005, pp. 1792–1799 (2005)
18. Zhang, W., Liu, Y., Clerc, M.: An adaptive PSO algorithm for real power optimization. In: APSCOM (Advances in Power System Control Operation and Management), S6: Application of Artificial Intelligence Technique (part I), Hong Kong, November 11-14, 2003, pp. 302–307 (2003)
19. `http://clerc.maurice.free.fr/pso/`
20. `http://www.particleswarm.info`