# A hybrid constrained optimization approach coupling PSO and adaptive constraint-handling technique

WEN LONG, SHAOHONG CAI, JIANJUN JIAO, WENZHUAN ZHANG
Guizhou Key Laboratory of Economics System Simulation
Guizhou University of Finance and Economics
No.276, Lu Chong Guan Road, Guiyang, Guizhou
CHINA
lw6822@163.com, gzcjdx_csh@163.com, jianjj73@163.com, zhangwz97@163.com

*Abstract:* - In this paper, we present a novel hybrid approach combining particle swarm optimization (PSO) and adaptive constraint-handling technique (ACT) for solving constrained numerical and engineering optimization problems. The proposed hybrid approach simultaneously adopts particle swarm optimizer and hybrid mutation operators to generate the offspring population. Additionally, the adaptive constraint-handling technique includes three main situations. In each situation, a constraint-handling mechanism is designed based on current population state. Our algorithm is validated using 15 well-known constrained numerical and engineering optimization problems reported in the literature. The experimental results demonstrate that the proposed method shows better performance in comparison to the state-of-the-art algorithms.

*Key-Words:* - Constrained optimization problem; Particle swarm optimization; Adaptive constraint-handling technique; Engineering optimization; mutation

## 1 Introduction

Constrained optimization problems are very important as they are encountered in many engineering applications. Conventional gradient-based optimization methods have difficulties to handle constrained optimization problems may usually lack an explicit mathematical formulation and have discrete definition domains. Compared with gradient-based optimization methods, evolutionary algorithms are population-based global search techniques, not sensitive to the characteristics of the problems and easy to implement. During the past decade, using evolutionary algorithm to solve constrained optimization problems has attracted a lot of research interest, and a large number of constrained optimization evolutionary algorithms have been proposed [1-3].

It is necessary to note that evolutionary algorithms are unconstrained search methods and lack an explicit mechanism to bias the search in constrained search space. In essence, constrained optimization evolutionary algorithms can be considered as constraint-handling techniques plus evolutionary algorithms, i.e., a proper constraint-handling technique needs to be considered in conjunction with an appropriate evolutionary algorithm, and these two aspects, thereby, should be mainly responsible for the performance of constrained optimization evolutionary algorithms.

As an important branch of evolutionary algorithms, particle swarm optimization (PSO) algorithm was originally introduced by Kennedy and Eberhart [4]. Similar to other evolutionary algorithms, PSO is a very simple population-based optimization technique which is at the same time very powerful and robust. PSO was originally designed for unconstrained optimization problems. However, being fascinated by the prospect and potential of PSO, many researchers have applied PSO to solve constrained optimization problems in the past decade [5-8].

Unlike the previous methods, this paper proposes a hybrid approach which combines a modified PSO algorithm with an adaptive constraint-handling technique to deal with constrained optimization problems. In the approach proposed, each particle in the population is applied to generate standard particle swarm optimizer and hybrid mutation operators. As a result, an offspring population is obtained. After combining the parent and offspring populations, an adaptive constraint-handling technique is designed. The proposed hybrid method is assessed on a total of fifteen constrained optimization problems to verify its performance. The experimental results show that it is very robust and effective for solving constrained optimization problems.

## 2 Particle Swarm Optimization

Particle swarm optimization is a population-based and derivative-free global optimization method based on the adaptation process observed in flocking organisms, such as birds, bees, fish, when searching for regions with food availability with the ability to return to promising regions that have previously been discovered [4]. With the standard particle swarm optimization, each particle of the swarm adjusts its trajectory according to its own flying experience and the flying experiences of other particles within its topological neighborhood in a *D*-dimensional space *S*. The velocity and position of particle $i$ are represented as $\vec{v}_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$ and $\vec{x}_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$, respectively. Its best historical position is recorded as $\vec{p}_i = (p_{i1}, p_{i2}, \cdots, p_{iD})$, which is also called $P_{best}$. The best historical position that the entire swarm has passed is denoted as $\vec{p}_g = (p_{g1}, p_{g2}, \cdots, p_{gD})$, which is also called $g_{best}$. The velocity and position of particle $i$ on dimension $d = (d = 1,2,\cdots,D)$ in iteration $t+1$ are updated as follows:

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) + c_2 r_2 (p_{gd}(t) - x_{id}(t)) \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (2)$$

where $\omega$ is a parameter called the inertia weight, $t$ is current iteration number, $c_1$ and $c_2$ are positive constants referred to as cognitive and social parameters, respectively, and $r_1$ and $r_2$ are random numbers generated from a uniform distribution in the region of [0,1].

## 3 Description of Proposed Approach

### 3.1 Algorithm framework

In this paper, the degree of constraint violation of a particle $\vec{x}$ on the *j*th constraints is calculated using the following expression:

$$G_j(\vec{x}) = \begin{cases} \max\{0, g_j(\vec{x})\}, & 1 \le j \le p \\ \max\{0, |h_j(\vec{x})| - \delta\}, & p+1 \le j \le m \end{cases} \quad (3)$$

$G(\vec{x}) = \sum_{j=1}^{m} G_j(\vec{x})$ reflects the degree of constraint violation of the particle $\vec{x}$.

The proposed hybrid method (is called PSO-ACT) procedure is shown in Fig. 1.

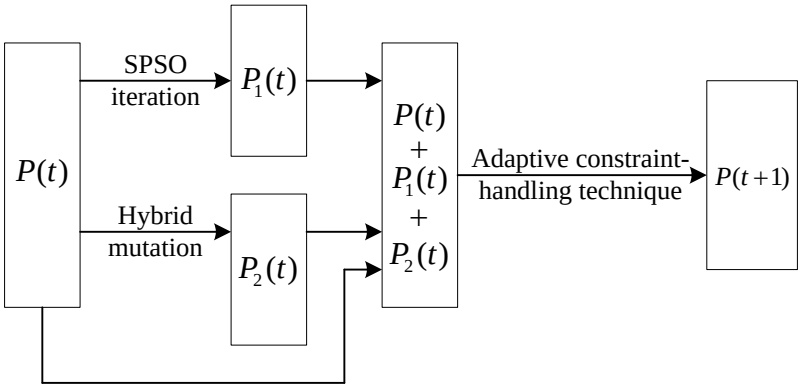


Fig.1. The framework of PSO-ACT

Coupling PSO algorithm and adaptive constraint-handling technique is (is called PSO-ACT) therefore given as:

Step 1 Set $t := 0$. Set parameter. Initialize a set of particle position and velocities by good point-set method from the decision space. Evaluate the objective function value $f$ and the degree of constraint violations $G$ for each particle in the initialization population.

Step 2 Generate new population $P_1(t)$ by update the velocity and position of each particle according to Eqs. (1) and (2). Generate new population $P_2(t)$ by hybrid mutation operators. Note that the PSO iteration and the hybrid mutation operators are applied in parallel rather than sequentially. Evaluate the $f$ value and $G$ value for each particle in $P_1(t) \cup P_2(t)$.

Step 3 Select $N$ (denote population size) particles from $P(t) \cup P_1(t) \cup P_2(t)$ to form the next population $P(t+1)$ based on adaptive constraint-handling technique that will be specified later.

Step 4 Set $t := t+1$.

Step 5 If stopping criterion is met, stop and return the optimal solution in $P(t)$, else go to Step 2.

### 3.2 Hybrid mutation operators

According to the update equations (1) and (2), the personal best position of the particle will gradually move closer to the global best position. Therefore, all the particles will converge onto the global best particle's position. Because of this mechanism, PSO algorithm can't guarantee to find the global minimal value of a function. In fact, the particles usually converge to local optima. To overcome the weakness of PSO algorithm, the hybrid mutation operators is introduced to generate new offspring particles. The reason for using such a mutation strategy is to increase the probability of escaping from a local optimum. It is worth noting that the hybrid mutation operators for the global best particle contain two components, i.e., Cauchy mutation and diversity mutation. A particle takes part in Cauchy mutation or diversity mutation with a probability of 0.5. Thus, no particle is subject to both mutation operators in the same generation.

The Cauchy mutation operator is described as follows [9]:

$$v'_{gd} = v_{gd} \exp(\mu) \tag{4}$$

$$x'_{gd} = x_{gd} + v'_{gd} \mu_g \tag{5}$$

where $x_{gd}$ and $v_{gd}$ represent the position and velocity of the global best particle. $\mu$ and $\mu_g$ denote Cauchy random numbers with the scale parameter of 1.

The diversity mutation operator is described as follows [10]:

To perform diversity mutation on a chosen particle $\vec{x} = (x_1, x_2, \cdots, x_n)$, randomly generate an integer $i_{rand}$ between 1 and $n$ with probability $1/n$ and a real number between $l_i$ and $u_i$, and then replace the $i_{rand}$ th component of the chosen particle by the real number to get a new particle $\vec{x}' = (x'_1, x'_2, \cdots, x'_n)$. The above procedure can be denoted by the following expression:

$$x'_i = \begin{cases} l_i + \beta(u_i - l_i) & i = i_{rand} \\ x_i & \text{otherwise} \end{cases}, i = 1,2,\cdots,n \tag{6}$$

where $\beta \in U(0,1)$ and $U(0,1)$ is a uniform random number generator in the range [0,1].

### 3.3 Adaptive constraint-handing technique

In general, for constrained optimization, the combined population $P(t) \cup P_1(t) \cup P_2(t)$ may inevitably experience three situations [10]: 1) the population contains in-feasible particles only (in-feasible situation); 2) the population consists of both feasible and in-feasible particles (semi-feasible situation); and 3) the population is entirely composed of feasible particles (feasible situation). In this paper, one constraint-handling mechanism is designed for one situation.

#### 3.3.1 The in-feasible situation

In this situation, since the combined population $P(t) \cup P_1(t) \cup P_2(t)$ contains no feasible particles, the aim is to promptly motivate the population toward the feasible region. In general, for constrained optimization, it does not make sense if a particle is far away from the boundaries of the feasible region. In addition, finding feasible particles is the most important objective in this situation. Thus, we only concerned with those particles with fewer constraint violations in the population. Firstly, the particles in the combined population $P(t) \cup P_1(t) \cup P_2(t)$ are ranked based on their constraint violations in ascending order, and then some excellent particles with least constraint violations are selected and form the offspring population. This scheme tends to guide the population toward feasibility from various directions.

#### 3.3.2 The semi-feasible situation

In this situation, since the combined population $P(t) \cup P_1(t) \cup P_2(t)$ consists of a combination of feasible and in-feasible particles, the aim is to maintain a reasonable proportion between feasible and in-feasible particles in the current population. Additionally, to keep the diversity and balance the exploration and exploitation ability of the population, some important feasible particles (feasible particles with small objective function values) and in-feasible particles (in-feasible particles with low constraint violation) should survive into the next generation since such particles are very promising for searching the optimal solution.

Firstly, the particles in the combined population $P(t) \cup P_1(t) \cup P_2(t)$ are ranked based on their constraint violations in ascending order and their

objective function values in ascending order, respectively, and then randomly generate a real number $\varepsilon$ between 0 and 1, if real number $\varepsilon \geq 0.5$, some excellent particles with least constraint violations are selected and form the offspring population; otherwise, some excellent particles with small objective function values are selected and form the offspring population. By this scheme, some potential feasible and in-feasible particles may survive into the next generation.

#### 3.3.3 The feasible situation

In this situation, since the combined population $P(t) \cup P_1(t) \cup P_2(t)$ is composed of feasible particles only, constrained optimization problems can be considered as unconstrained optimization problems. Furthermore, since the ultimate goal of constrained optimization evolutionary algorithms are to find the feasible optimal solution. Therefore, the comparison of particles is based only on the objective function values. Afterward, some excellent particles with the small objective function values are selected for the next generation.

## 4 Experimental Study

### 4.1 Benchmark test functions

Numerical simulations were executed based on 13 well-known benchmark test functions [3] to evaluate the performance of the proposed hybrid method. Note that test functions g02, g03, g08, and g12 are maximization problems, and the others are minimization problems. In this study, the maximization problems are transformed into minimization using $-f(\vec{x})$. In addition, only test functions g03, g05, g11, and g13 contain equality constraints.

The following parameters are established experimentally for the best performance of PSO-ACT: the population size $N = 200$, the acceleration constants $c_1 = c_2 = 1.496180$, the inertia weight $\omega = 0.729844$, the tolerant value $\delta = 10e-06$. In addition, as the Cauchy mutation or the diversity mutation occurs with a probability of 0.5 for each particle in the population. The above parameter settings are kept for all experiments. The number of fitness function evaluations (FFEs) is fixed to 240,000. In this paper, 30 independent runs are performed for each test function.

### 4.2 Experimental results

Table 1 exhibits the overall performance of our PSO-ACT. This table shows the "known" optimal solution for each test function and the "best", "median", "mean", "worst", and standard deviations of the objective function values achieved by PSO-ACT.

Table 1. Experimental results obtained by PSO-ACT for 13 test functions over 30 independent runs

| Function | Known optimal | Best | Median | Mean | Worst | St.dev |
|---|---|---|---|---|---|---|
| g01 | −15.0000000 | **−15.0000000** | −15.0000000 | −15.0000000 | −15.0000000 | 6.44E−09 |
| g02 | −0.80361910 | −0.80359815 | −0.77683374 | −0.76928989 | −0.71849287 | 2.56E−02 |
| g03 | −1.00050010 | −1.00000000 | −1.00000000 | −1.00000000 | −1.00000000 | 7.32E−14 |
| g04 | −30665.53867 | **−30665.53867** | −30665.53867 | −30665.53867 | −30665.53867 | 1.82E−12 |
| g05 | 5126.496714 | 5126.498110 | 5126.498110 | 5126.498110 | 5126.498110 | 1.36E−12 |
| g06 | −6961.813876 | **−6961.813876** | −6961.813876 | −6961.813876 | −6961.813876 | 1.02E−12 |
| g07 | 24.30620907 | **24.30620907** | 24.30620907 | 24.30620907 | 24.30620907 | 8.38E−10 |
| g08 | −0.09582504 | **−0.09582504** | −0.09582504 | −0.09582504 | −0.09582504 | 0.00E+00 |
| g09 | 680.6300574 | **680.6300574** | 680.6300574 | 680.6300574 | 680.6300574 | 5.68E−14 |
| g10 | 7049.248021 | 7049.248096 | 7049.248173 | 7049.248168 | 7049.248283 | 7.33E−05 |
| g11 | 0.749900000 | 0.749999000 | 0.749999000 | 0.749999000 | 0.749999000 | 0.00E+00 |
| g12 | −1.00000000 | **−1.00000000** | −1.00000000 | −1.00000000 | −1.00000000 | 0.00E+00 |
| g13 | 0.053941514 | 0.053949848 | 0.053949848 | 0.053949848 | 0.053949848 | 1.97E−15 |

As shown in Table 1, PSO-ACT has the ability to consistently converge to the global optima for seven test functions, i.e. g01, g04, g06, g07, g08, g09, and g12. With respect to the rest of the test functions (i.e. g02, g03, g05, g10, g11, and g13), although the optimal solutions are not consistently found, the "best" results achieved are fairly close to the global optimal values known. The test function g02 has many local optima with high peak near the global optimal solution. The resulting solutions achieved for test function g02 have been exhibited in Fig. 2.

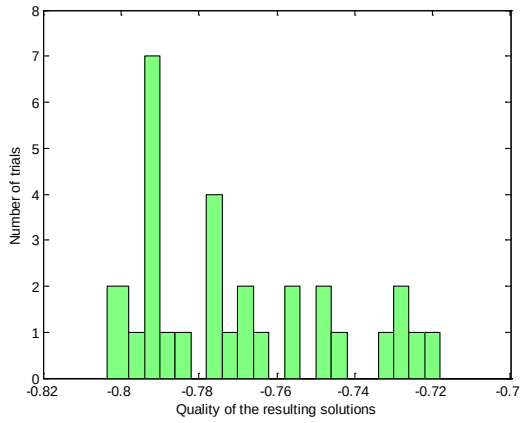


Fig.2. Plots show the number of trials versus the quality of the resulting solutions achieved for function g02

It is noteworthy that the standard deviations over 30 independent runs for all test functions are fairly small. In particular, the standard deviation for test functions g08, g11, and g12 are equal to 0. In addition, the "mean" and "worst" results provided by PSO-ACT are very near the "best" results obtained, which indicates that the performance of PSO-ACT is very stable in producing consistent results for all test functions. Moreover, the computational cost of PSO-ACT is 240000 FFEs for all test functions. The above discussion validates that PSO-ACT is an effective and efficient method for constrained optimization.

### 4.3 Comparison with the other methods

In order to further verify the performance of PSO-ACT, the results of our algorithm are compared against five typical state-of-the-art approaches from the literatures, including the self-adaptive velocity particle swarm optimization (SAVPSO) [2], the constrained multi-swarm particle swarm optimization (CPSO) [5], the improved vector particle swarm optimization (IVPSO) [7], the constraint- handling mechanism for particle swarm optimization (CHMPSO) [11], and the hybrid particle swarm optimization and differential evolution (PSO-DE) [12]. Among these algorithms, CPSO is one of the most competitive PSO algorithms known to date. The comparison results are listed in Table 2-4.

Table 2. Comparing of the proposed PSO-ACT with respect to other algorithms on the best solutions (NA means not available)

| Function | Optimal | SAVPSO[2] | IVPSO[7] | CHMPSO[11] | PSO-DE[12] | CPSO[5] | PSO-ACT |
|---|---|---|---|---|---|---|---|
| g01 | −15.00000000 | −15 | −15 | −15 | −15.000000 | −15.00000000 | −15.0000000 |
| g02 | −0.803619104 | −0.803443 | −0.803619 | −0.803432 | −0.8036145 | −0.803619104 | −0.80359815 |
| g03 | −1.000500100 | −1.0048 | −1.005010 | −1.004720 | −1.0050100 | −1.000500100 | −1.00000000 |
| g04 | −30665.53867 | −30665.539 | −30665.539 | −30665.5 | −30665.539 | −30665.53867 | −30665.53867 |
| g05 | 5126.4967140 | 5126.4841 | 5126.492646 | 5126.64 | NA | 5126.4967140 | 5126.498110 |
| g06 | −6961.813876 | −6961.81388 | −6961.813876 | −6961.81 | −6961.8139 | −6961.813876 | −6961.813876 |
| g07 | 24.306209068 | 24.319 | 24.306497 | 24.3511 | 24.306209 | 24.306209068 | 24.306209068 |
| g08 | −0.095825042 | −0.095825 | −0.095825 | −0.095825 | −0.095826 | −0.095825042 | −0.095825042 |
| g09 | 680.63005737 | 680.632 | 680.630 | 680.638 | 680.63006 | 680.63005737 | 680.63005737 |
| g10 | 7049.2480205 | 7054.1256 | 7049.351110 | 7057.5900 | 7049.2480 | 7049.2480205 | 7049.248096 |
| g11 | 0.7499000000 | 0.749000 | 0.749000 | 0.749999 | 0.749999 | 0.7499000000 | 0.749999000 |
| g12 | −1.000000000 | −1.000000 | −1.000000 | −1.000000 | −1.000000 | −1.000000000 | −1.00000000 |
| g13 | 0.0539415140 | 0.0538666 | 0.053988 | 0.068665 | NA | 0.0539415140 | 0.053949848 |

Table 3. Comparing of the proposed PSO-ACT with respect to other algorithms on the mean solutions (NA means not available)

| Function | Optimal | SAVPSO[2] | IVPSO[7] | CHMPSO[11] | PSO-DE[12] | CPSO[5] | PSO-ACT |
|---|---|---|---|---|---|---|---|
| g01 | −15.00000000 | −14.7151 | −15 | −15 | −15.000000 | −15.00000000 | −15.0000000 |
| g02 | −0.803619104 | −0.740577 | −0.769889 | −0.790406 | −0.756678 | −0.801653204 | −0.76928989 |
| g03 | −1.000500100 | −1.0034 | −1.005010 | −1.003814 | −1.0050100 | −1.000500100 | −1.00000000 |
| g04 | −30665.53867 | −30665.539 | −30665.539 | −30664.5 | −30665.539 | −30665.53867 | −30665.53867 |
| g05 | 5126.4967140 | 5202.3627 | 5126.492646 | 5461.081333 | NA | 5126.4967140 | 5126.498110 |
| g06 | −6961.813876 | −6961.81388 | −6961.81388 | −6961.81 | −6961.8139 | −6961.813876 | −6961.813876 |
| g07 | 24.306209068 | 24.989 | 24.429646 | 25.355771 | 24.306210 | 24.306209068 | 24.30620907 |
| g08 | −0.095825042 | −0.095825 | −0.095825 | −0.095825 | −0.0958259 | −0.095825042 | −0.09582504 |
| g09 | 680.63005737 | 680.655 | 680.630 | 680.852393 | 680.63006 | 680.63005737 | 680.6300574 |
| g10 | 7049.2480205 | 7173.2661 | 7069.885232 | 7560.047857 | 7049.2480 | 7049.2480205 | 7049.248168 |
| g11 | 0.7499000000 | 0.749000 | 0.749000 | 0.750107 | 0.749999 | 0.7499000000 | 0.749999000 |
| g12 | −1.000000000 | −1 | −1.000000 | −1.000000 | −1.000000 | −1.000000000 | −1.00000000 |
| g13 | 0.0539415140 | 0.552753 | 0.462385 | 1.716426 | NA | 0.0539415188 | 0.053949848 |

Table 4. Comparing of the proposed PSO-ACT with respect to other algorithms on the worst solutions (NA means not available)

| Function | Optimal | SAVPSO[2] | IVPSO[7] | CHMPSO[11] | PSO-DE[12] | CPSO[5] | PSO-ACT |
|---|---|---|---|---|---|---|---|
| g01 | −15.00000000 | −12.4531 | −15 | −15 | −15.000000 | −15.00000000 | −15.0000000 |
| g02 | −0.803619104 | −0.631598 | −0.703477 | −0.750393 | −0.6367995 | −0.784076104 | −0.71849287 |
| g03 | −1.000500100 | −0.9976 | −1.005010 | −1.002490 | −1.0050100 | −1.000500100 | −1.00000000 |
| g04 | −30665.53867 | −30665.539 | −30665.539 | −30665.5 | −30665.539 | −30665.53867 | −30665.53867 |
| g05 | 5126.4967140 | 5520.1467 | 5126.492646 | 6104.75 | NA | 5126.4967140 | 5126.498110 |
| g06 | −6961.813876 | −6961.81388 | −6961.81388 | −6104.75 | −6961.8139 | −6961.813876 | −6961.813876 |
| g07 | 24.306209068 | 26.194 | 25.004855 | 24.374 | 24.3062 | 24.306209068 | 24.30620907 |
| g08 | −0.095825042 | −0.095825 | −0.095825 | −0.095825 | −0.0958259 | −0.095825042 | −0.09582504 |
| g09 | 680.63005737 | 680.699 | 680.630139 | 681.553000 | 680.6301 | 680.63005737 | 680.6300574 |
| g10 | 7049.2480205 | 7335.2477 | 7251.241769 | 8104.310000 | 7049.2482 | 7049.2480205 | 7049.248283 |
| g11 | 0.7499000000 | 0.749021 | 0.749000 | 0.752885 | 0.750001 | 0.7499000000 | 0.749999000 |
| g12 | −1.000000000 | −1 | −1.000000 | −1.000000 | −1.000000 | −1.000000000 | −1.00000000 |
| g13 | 0.0539415140 | 1.856102 | 0.996901 | 13.669500 | NA | 0.0539415825 | 0.053949848 |

As shown in Tables 2-4, the performance of the proposed PSO-ACT is compared in detail with the five algorithms in terms of the selected performance metrics. Results found by all algorithms for functions g01, g08, and g12 were largely in accord with each other. With respect to SAVPSO, the proposed method finds better "best" results in seven problems (g02, g04, g06, g07, g08, g09 and g10) and similar "best" results in two problems (g01 and g12). Also, the proposed PSO-ACT reaches better "mean" and "worst" results in nine problems (g01, g02, g04, g05, g06, g07, g09, g10 and g13). Similar "mean" and "worst" results are found in two problems (g08 and g12). Compare with IVPSO, the proposed PSO-ACT provides similar "best", "mean", and "worst" results for three problems (g01, g06 and g12). Better "best" and "mean" results are found by IVPSO in four problems (g02, g03, g05 and g11). Also, the proposed PSO-ACT reaches better "best" results in five test functions (g04, g07, g08, g09 and g10) and "mean" and "worst" results in eight problems (g02, g04, g06, g07, g08, g09, g10 and g13). With respect to CHMPSO, the proposed PSO-ACT provides better "best", "mean" and "worst" results in ten problems (g02, g04, g05, g06, g07, g08, g09, g10, g11 and g13) and similar results in two test functions (g01 and g12). Compare with PSO-DE, PSO-ACT finds better results in eight problems (g04, g05, g06, g07, g08, g09, g10 and g13) and similar results in three test functions (g01, g11 and g12). A better "best" result is found by PSO-DE in test function g02; however, PSO-ACT reaches better "mean" and "worst" results. CPSO is one of the most competitive constrained PSO algorithms known to date. With respect to CPSO, PSO-ACT finds similar results in eight problems (g01, g04, g06, g07, g08, g09, g11 and g12) and worse results in five test functions (g02, g03, g05, g10 and g13).

As a general remark on the comparison above, PSO-ACT shows a very competitive performance with respect to five state-of-the-art algorithms in terms of the quality, the robustness, and the efficiency of search.

## 4.4 Engineering design optimization

In order to further study the performance of PSO-ACT on real-world engineering constrained optimization problems, two well-studied engineering design optimization cases are solved using this approach. The parameter settings are the same as those used by the previous experiments for 13 benchmark test functions except for the number of FFEs.

### 4.4.1 Pressure vessel design

In this case, a cylindrical pressure vessel with two hemispherical heads is designed for minimum fabrication cost. Four variables are identified: thickness of the pressure vessel $T_s(x_1)$, thickness of the head $T_h(x_2)$, inner radius of the vessel $R(x_3)$, and length of the vessel without heads $L(x_4)$ (see Fig. 3).

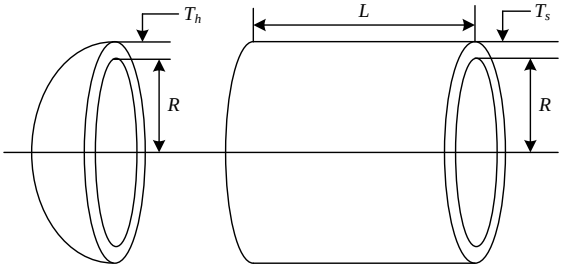


Fig.3. Schematic view of pressure vessel design

This problem has been solved previously using GA3, Gaussian quantum-behaved particle swarm optimization method (GQPSO), CEPSO, HPSO, hybrid Nelder-Mead simplex search particle swarm optimization (NMPSO) and CDE [13]. Table 5

shows the comparisons of the best solutions obtained by the proposed PSO-ACT and other compared approaches.

Table 5. Comparison of the best solution obtained from various studies for the pressure vessel design

| Design variables | GA3[13] | CEPSO[13] | GQPSO[13] | NMPSO[13] | HPSO[13] | CDE[13] | PSO-ACT |
|---|---|---|---|---|---|---|---|
| $x_1(T_s)$ | 0.8125 | 0.8125 | 0.8125 | 0.8036 | 0.8125 | 0.8125 | 0.8125 |
| $x_2(T_h)$ | 0.4375 | 0.4375 | 0.4375 | 0.3972 | 0.4375 | 0.4375 | 0.4375 |
| $x_3(R)$ | 42.0974 | 42.0913 | 42.0984 | 41.6392 | 42.0984 | 42.0984 | 42.098445 |
| $x_4(L)$ | 176.6540 | 176.7465 | 176.6372 | 182.4120 | 176.6366 | 176.6376 | 176.636595 |
| $g_1(x)$ | −2.01E−03 | −1.37E−06 | −8.79E−07 | 3.65E−05 | −8.80E−07 | −6.67E−07 | −1.15E−08 |
| $g_2(x)$ | −3.58E−02 | −3.59E−04 | −3.58E−02 | 3.79E−05 | −3.58E−02 | −3.58E−02 | −3.59E−02 |
| $g_3(x)$ | −24.7593 | −118.7687 | −0.2179 | −1.5914 | 3.1226 | −3.705123 | 4.58E−02 |
| $g_4(x)$ | −63.3460 | −63.2535 | −63.3628 | −57.5879 | −63.3634 | −63.3623 | −63.3634 |
| $f(x)$ | 6059.9463 | 6061.0777 | 6059.7208 | 5930.3137 | 6059.7143 | 6059.7340 | 6059.714215 |

Table 6. Statistical results of different methods for pressure vessel design

| Feature | GA3[13] | CEPSO[13] | GQPSO[13] | NMPSO[13] | HPSO[13] | CDE[13] | PSO-ACT |
|---|---|---|---|---|---|---|---|
| Best | 6059.9463 | 6061.0777 | 6059.7208 | 5930.3137 | 6059.7143 | 6059.7340 | 6059.714215 |
| Mean | 6177.2533 | 6147.1332 | 6440.3786 | 5946.7901 | 6099.9323 | 6085.2303 | 6059.714215 |
| Worst | 6469.3220 | 6363.8041 | 7544.4925 | 5960.0557 | 6288.6770 | 6371.0455 | 6059.714215 |
| Std. | 130.9297 | 86.4500 | 448.4711 | 9.1610 | 86.2000 | 43.0130 | 1.01E−12 |

The comparison of obtained statistical results for the proposed PSO-ACT with previous studies including GA3, CEPSO, GQPSO, NMPSO, HPSO, and CDE is presented in Table 6. As can be seen from Table 6, the proposed PSO-ACT obtained the best solution in 35000 FFEs which is superior to other considered algorithms except for NMPSO.

### 4.4.2 Tension/compression string design

The tension/compression string design problem is described in Eskandar [13] for which the objective is to minimize the weight ( $f(x)$ ) of a tension/ compression string (as shown in Fig. 4) subject to constraints on minimum deflection, shear stress, surge frequency, limits on outside diameter and on design variables. The independent variables are the wire diameter $d(x_1)$, the mean coil diameter $D(x_2)$, and the number of active coils $P(x_3)$.

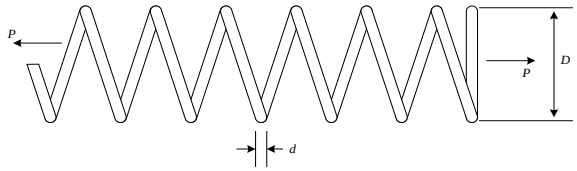


Fig.4. Tension/compression string design

The approaches applied to this problem for comparisons include GA3, CEPSO, HPSO, NMPSO, QPSO, GQPSO, DEDS, and PSO-DE. The best solutions obtained by the above approaches and PSO-ACT are listed in Table 7, and the statistical results are shown in Table 8.

Table 7. Comparison of the best solution obtained from various studies for tension/compression string design

| Design variables | GA3[13] | CEPSO[13] | GQPSO[13] | NMPSO[13] | HPSO[13] | PSO-DE[13] | PSO-ACT |
|---|---|---|---|---|---|---|---|
| $x_1(d)$ | 0.051989 | 0.051728 | 0.051515 | 0.051620 | 0.051706 | 0.0516888 | 0.05168906 |
| $x_2(D)$ | 0.363965 | 0.357644 | 0.352529 | 0.355498 | 0.357126 | 0.3567117 | 0.35671774 |
| $x_3(P)$ | 10.890522 | 11.244543 | 11.538862 | 11.333272 | 11.265083 | 11.289320 | 11.2889656 |
| $g_1(x)$ | −1.26E−03 | −8.25E−04 | −4.83E−05 | 1.01E−03 | NA | NA | −1.98E−13 |
| $g_2(x)$ | −2.54E−05 | −2.52E−05 | −3.58E−05 | 9.94E−04 | NA | NA | 1.51E−13 |
| $g_3(x)$ | −4.061337 | −4.051306 | −4.0455 | −4.061859 | NA | NA | −4.0537856 |
| $g_4(x)$ | −0.722697 | −0.727085 | −0.73064 | −0.728588 | NA | NA | −0.7277288 |
| $f(x)$ | 0.0126810 | 0.0126747 | 0.012665 | 0.012630 | 0.0126652 | 0.0126652 | 0.01266523 |

Table 8. Statistical results of different approaches for tension/compression string design

| Feature | GA3[13] | CEPSO[13] | GQPSO[13] | NMPSO[13] | HPSO[13] | PSO-DE[13] | PSO-ACT |
|---|---|---|---|---|---|---|---|
| Best | 0.012681 | 0.012674 | 0.012665 | 0.012630 | 0.012665 | 0.012665 | 0.01266523 |
| Mean | 0.012742 | 0.012730 | 0.013524 | 0.012631 | 0.012707 | 0.012665 | 0.01266523 |
| Worst | 0.012973 | 0.012924 | 0.017759 | 0.012633 | 0.012719 | 0.012665 | 0.01266523 |
| Std. | 5.90E−05 | 5.20E−04 | 1.27E−03 | 8.47E−07 | 1.58E−05 | 1.20E−08 | 2.74E−10 |

From Table 7, it can be observed that the proposed PSO-ACT is robust and find solutions in 24000 FFEs which are better than the "best" solution found by GA3 and CEPSO. With respect to GQPSO, HPSO, and PSO-DE, the proposed PSO-ACT finds similar "best" results. A better "best" solution is obtained by NMPSO. As can be seen from Table 8, compare with GA3, CEPSO, GQPSO, and HPSO, the proposed PSO-ACT provides better results for tension/compression string design problem. With respect to PSO-DE, the proposed PSO-ACT reaches similar results.

Based on the aforementioned simulation and comparison results validate that the proposed PSO-ACT has the substantial capability in handling various constrained optimization problems and its solution quality is quite stable. So, it can be concluded that the proposed PSO-ACT is a good alternative for constrained optimization.

## 5 Conclusion

In this paper, we have presented the hybrid approach coupling particle swarm optimization and adaptive constraint-handling technique for general nonlinear constrained optimization problems and engineering design optimization problems. In the approach proposed, each particle in the population is applied to generate standard particle swarm optimizer and hybrid mutation operators. As a result, an offspring population is obtained. It is worthwhile to note that the standard particle swarm optimizer and the hybrid mutation operation are implemented concurrently and that the Cauchy mutation or diversity mutation is applied to a particle in the population with a probability of 0.5. After combining the parent and offspring populations, an adaptive constraint-handling technique is designed. The approach has been tested experimentally based numerical and engineering constrained design optimization problems. The experimental results indicate that the proposed hybrid method is very suitable for constrained optimization problems with different types and that it is superior to or competitive with the compared approaches. The future work will be focused on extension of the approach to solve multi-objective problems.

## Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant No. 61463009 , the Humanities and Social Sciences Planning Foundation of Ministry of Education of China under Grant No. 12XJA910001, and the 125 Special Major Science and Technology of Department of Education of Guizhou Province under Grant No. [2012]011.

*References:*

[1] Long W, Jiao J, "Hybrid cuckoo search algorithm based on Powell search for constrained engineering design optimization". *WSEAS Transactions on Mathematics*, Vol. 13, 2014, pp: 431-440.

[2] Lu H Y, Chen W Q, "Self-adaptive velocity particle swarm optimization for solving constrained optimization problems". *Journal of Global Optimization*, Vol. 41, no. 3, 2008, pp: 427−445.

[3] Long W, Liang X M, Huang Y F, Y.X. Chen Y X, "A hybrid differential evolution augmented Lagrangian method for constrained numerical and engineering optimization". *Computer-Aided Design*, Vol. 45, no. 12, 2013, pp: 1562−1574.

[4] Kennedy J, Eberhart R, "Particle swarm optimization". *In: Proceedings of IEEE International Conference on Neural Networks*, Washington, USA, Vol. 4, 1995, pp: 1942−1948.

[5] Daneshyari M, Yen G, "Constrained multiple-swarm particle swarm optimization within a cultural framework". *IEEE Transactions on Systems, Man, Cybernetics*, Vol. 42, no. 2, 2012, pp: 475−490.

[6] Sedlaczek K, Eberhard P, "Using augmented Lagrangian particle swarm optimization for constrained problems in engineering". *Structural Multidisciplinary Optimization*, Vol. 32, no. 4, 2006, pp: 277−286.

[7] Sun C L, Zeng J C, Pan J, "An improved vector particle swarm optimization for constrained optimization problems". *Information Sciences*, Vol. 181, no. 6, 2011, pp: 1153−1163.

[8] Kou X L, Liu S Y, Zhang J K, Zheng W, "Co-evolutionary particle swarm optimization to solve constrained optimization problems". *Computers & Mathematics with Applications*, Vol. 57, no. 11−12, 2009, pp: 1776−1784.

[9] Li C, Yang S, Korejo I, "An adaptive mutation operator for particle warm optimization", *in: Proceedings of the UK Workshop on Computational Intelligence*, UK, 2008.

[10] Wang Y, Cai Z, Zhou Y, Zhun F, "Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique". *Structural Multidisciplinary Optimization*, Vol. 37, no. 4, 2009, pp: 395−413.

[11] Pulido G T, Coello C A C, "A constraint-handling mechanism for particle swarm optimization". *in: Proceedings of the IEEE Congress on Evolutionary Computation*, Portland, USA, 2004.

[12] Liu H, Cai Z X, Wang Y, "Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization". *Applied Soft Computing*, Vol. 10, no. 3, 2010, pp: 629–640.

[13] Eskandar H, Sadollah A, Bahreininejad A, et al., "Water cycle algorithm—A novel metaheuristic optimization method for solving constrained engineering optimization problems". *Computers & Structures*, Vol. 110–111, 2012, pp: 151–166.