J Glob Optim (2008) 41:427–445
DOI 10.1007/s10898-007-9255-9

Self-adaptive velocity particle swarm optimization for solving constrained optimization problems

Haiyan Lu · Weiqi Chen

Received: 11 June 2006 / Accepted: 8 October 2007 / Published online: 31 October 2007

Abstract Particle swarm optimization (PSO) is originally developed as an unconstrained optimization technique, therefore lacks an explicit mechanism for handling constraints. When solving constrained optimization problems (COPs) with PSO, the existing research mainly focuses on how to handle constraints, and the impact of constraints on the inherent search mechanism of PSO has been scarcely explored. Motivated by this fact, in this paper we mainly investigate how to utilize the impact of constraints (or the knowledge about the feasible region) to improve the optimization ability of the particles. Based on these investigations, we present a modified PSO, called self-adaptive velocity particle swarm optimization (SAVPSO), for solving COPs. To handle constraints, in SAVPSO we adopt our recently proposed dynamic-objective constraint-handling method (DOCHM), which is essentially a constituent part of the inherent search mechanism of the integrated SAVPSO, i.e., DOCHM + SAVPSO. The performance of the integrated SAVPSO is tested on a well-known benchmark suite and the experimental results show that appropriately utilizing the knowledge about the feasible region can substantially improve the performance of the underlying algorithm in solving COPs.

Keywords Constrained optimization · Particle swarm optimization · Stochastic optimization · Evolutionary algorithms · Nonlinear programming · Constraint-handling mechanism

H. Lu (✉)
School of Science, Jiangnan University, Wuxi 214122, P.R. China
e-mail: kangting88@hotmail.com

H. Lu
Department of Mathematics, Zhejiang University, Hangzhou 310027, P.R. China

W. Chen
School of Information Technology, Jiangnan University, Wuxi 214122, P.R. China

W. Chen
China Ship Scientific Research Center, Wuxi 214082, P.R. China

1 Introduction

Many real-world applications, such as engineering design, VLSI design, structural optimization, economics, location and allocation problems [1], involve difficult constrained optimization problems (COPs) that must be solved efficiently and effectively. Due to the complex nature of many of these problems, deterministic optimization approaches such as Feasible Direction and Generalized Gradient Descent are often unable to provide even a feasible solution, since these approaches make strong assumptions on the continuity and differentiability of the objective function [1,2]. This has provided an opportunity for evolutionary algorithms such as Genetic Algorithm, Evolutionary Strategies, Evolutionary Programming and Particle swarm optimization (PSO), which have been successfully applied for tackling COPs during the past few years [3–7].

PSO is population-based, global, and stochastic optimization algorithm developed by Kennedy and Eberhart in 1995 [8,9]. It has gaining an increasing popularity due to its simplicity and effectiveness in performing difficult optimization tasks. However, like other aforementioned stochastic algorithms, PSO lacks an explicit constraint-handling mechanism. A number of constraint-handling mechanisms have been proposed for evolutionary algorithms [3,10]. In recent years, several studies have been devoted to incorporated some constraint-handling mechanisms into PSO algorithm for solving COPs [4,6,11–14]. Nonetheless, most of these studies are focused on the methods for "handling constraints" of COPs, little attention has been paid on investigating the influence of constraints on the particles' flying modes. Motivated by this fact, in this work we not only incorporate appropriate constraint-handling technique into the algorithm proposed, but also investigate how to improve its inherent algorithmic mechanism under the influence of constraints.

PSO emulates the flying pattern (or search mechanism) of particles in search space without constraints (except the bound constraints), hence it is not directly applicable to COPs. Since PSO does not take into accounts the impact of constraints on the search mechanism, it is usually difficult to concentrate the particle in the approximate region of the feasible region (especially when the feasible region is very small), impairing the optimization ability of the algorithm. Therefore, we imagine that if we could appropriately incorporate the impact of constraints into the search mechanism of PSO algorithm, then the optimization ability of PSO would be improved. For this, in this paper we primarily investigate the potential impact of constraints on the flying pattern (or the search mechanism) of particles, explore the methods for incorporating this impact into the inherent algorithmic mechanism of PSO, and thereby propose a simple incorporation strategy, which we refer to as self-adaptive velocity particle swarm optimization (SAVPSO). This name is derived from the property that in our proposed algorithm, each particle has the ability to self-adaptively adjust its velocity according to some characteristics of feasible region.

In addition, to deal with constraints directly, some proper constraint-handling mechanism is necessary when applying SAVPSO to COPs. As aforementioned, there have been several methods for handling constraints with PSO. One commonly used and straightforward approach is via penalty function, which converts the COP into a unconstrained optimization problem [15]. This method may work quite well for some problems, but it requires a careful tuning of the penalty parameters, and which turns out to be a difficult optimization problem itself [16]. Another constraint-handling method with PSO is preserving feasible solutions, which was adapted from [17] by Hu and Eberhart [12]. This method requires an initialization of particles inside the feasible region, which may need a long time initialization process and may be hard to achieve for some problems. Recently, some researchers have proposed some hybrid PSO algorithms combined with some constraint-handling mechanisms [13,18]. In

this paper, we adopt our recently proposed dynamic-objective constraint-handling method (DOCHM) to handle constraints [19]. DOCHM operates on the inherent search mechanism of SAVPSO and reflects in some aspect the impact of the constraints. The main aim of DOCHM is to compel the particles to search for the feasible region, and this is accomplished by minimizing a distance function which is treated as the first objective to be optimized, and optimizing the original objective (of the original problem) which is treated as the second objective of DOCHM.

The rest of this paper is organized as follows. Section 2 introduces the problem of interest to us. Section 3 describes the standard PSO and analyze the the aspects of its inapplicability to COPs. In Sect. 4, we present our proposed SAVPSO, including a brief description of DOCHM. In Sect. 5, experiments were performed on 13 well-known benchmark functions to evaluate the performance of our proposed SAVPSO, and a comparison of results with respect to some other algorithms are provided. Finally, our conclusion and future work are given in Sect. 6.

2 Problem statement

The constrained optimization problems (COPs) or the general nonlinear programming problems (NLPs) can be formulated as follows:

$$\text{minimize} \quad f(x) \tag{1}$$

subject to

$$g_i(x) \leq 0, \quad i = 1, \ldots, q \tag{2}$$

$$h_j(x) = 0, \quad j = q+1, \ldots, m \tag{3}$$

where $x = (x_1, x_2, \ldots, x_n)$ is the vector of solutions such that $x \in S \subseteq R^n$, q is the number of inequality constraints. The *search space* S is defined as an n-dimensional space bounded by *parametric constraints*

$$x_d^l \leq x_d \leq x_d^u, \quad d = 1, \ldots, n \tag{4}$$

and the *feasible region* $F \subseteq S$ is the region of S for which the inequality and equality constraints are satisfied. For an inequality constraint that satisfies $g_i(x) = 0$, then we will say that is active at x. All equality constraints h_j (regardless of the values of x used) are said to be active at all points of F. As a common practice in the specialized literature on evolutionary algorithms, equality constraints are transformed into inequalities of the form

$$|h_j(x)| - \delta \leq 0, \quad \text{for } j = q+1, \ldots, m \tag{5}$$

where δ is the tolerance allowed (a very small positive value). A solution x is regarded as *feasible* if $g_i(x) \leq 0$, for $i = 1, \ldots, q$ and $|h(x)| - \delta \leq 0$, for $j = q+1, \ldots, m$.

3 PSO and its inapplicability to COPs

Since its introduction in 1995 [8,9], the PSO algorithm has experienced many changes, several of which have turned out to cause genuine improvements in performance [20]. In this section, we take the following typical variant of PSO as an example, which we refer to as the standard PSO (SPSO for short), to analyze the aspects of inapplicability of SPSO algorithms to COPs.

In a n-dimensional search space, $S \subseteq R^n$, assume that the swarm consists of N particles. The i-th particle is in effect an n-dimensional vector $x_i = (x_{i1}, x_{i2}, \ldots, x_{in}) \in S$. The velocity of this particle is also a n-dimensional vector $v_i = (v_{i1}, v_{i2}, \ldots, v_{in}) \in S$. The best previous position visited by the i-th particle is a point in S, denoted as $p_i = (p_{i1}, p_{i2}, \ldots, p_{in})$. Let g be the index of the particle that attained the best previous position among the entire swarm, and t be the iteration counter. Then in SPSO, the swarm is manipulated according to the following update equations [21–23]:

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) + c_2 r_2 (p_{gd}(t) - x_{id}(t)) \quad (6)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (7)$$

where $i = 1, 2, \ldots, N$ is the particle's index, $d = 1, 2, \ldots, n$ indicates the particle's d-th component, ω is a parameter called the *inertia weight*, 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 uniformly distributed in [0, 1], denoted as $r_1, r_2 \in U[0, 1]$.

Consider the d-th dimension of the search space, $d = 1, 2, \ldots, n$. The right-hand-side of (6) consists of three parts [24]. The first part $\omega v_{id}(t)$ is the momentum part. The second part is the "cognitive" part which represents personal thinking of itself—learning from its own flying experience. The third part is the "social" part which represents the collaboration among particles—learning from group flying experience. In fact, the sum of the last two parts, i.e., $c_1 r_1 (p_{id}(t) - x_{id}(t)) + c_2 r_2 (p_{gd}(t) - x_{id}(t))$, can be considered as, for the time being, the newly gained velocity term towards a potential position p' in the promising region around $p_{gd}(t)$ and $p_{id}(t)$. While the first part $\omega v_{id}(t)$ can be viewed as the credibility on the previous velocity of the particle, or the modification term on the newly gained velocity, which would impel the particle to deviate the potential position p'. Consequently, summing these two terms results in the current velocity $v_{id}(t+1)$. However, these two terms do not take into account the influence of the feasible region. For example, each or both of these two components might be so large that the corresponding particle would leave far away from the feasible region. This poses one of the great difficulties of PSO in solving COPs. Therefore, we expect that if the impact of constraints could be appropriately exploited, then the optimization ability of the algorithm might be substantially improved.

4 Our approach

In this section, we present in detail our approach to incorporating the impact of constraints into the inherent search mechanism of PSO.

4.1 Impact of constraints on the search mechanism of PSO

Recall that in Sect. 3, we divide the velocity expression (6) into two terms and explain the implicit meaning of them respectively. Our discussion shows that the search mechanism of PSO takes no account of the impact of constraints or the knowledge about the feasible region. In this paper we will investigate how to incorporate this impact or utilize this kind of knowledge appropriately so as to improve the performance of the resulting algorithm for solving COPs. For this reason, we first analyze the knowledge (about the feasible region) that might have impact upon the search behavior of the particles. We conjecture that the following three characteristics of the feasible region, which can be considered as some kind of knowledge about the feasible region, are responsible for the impact on the search behavior of the particles:

(1) The position of the feasible region with respect to the search space;
(2) The connectivity and the shape of the feasible region;
(3) The ratio $|F|/|S|$ of feasible region to the search space.

Characteristic (1) implies that the particles should first of all search for the feasible region so as to find the optimal solution within the feasible region. Therefore, the feasible region can be viewed as the objective of the particle lying outside the feasible region. In our algorithm discussed later, this kind of impact is reflected and realized by DOCHM (see Sect. 4.4), a constraint-handling method recently proposed by us in [19].

Characteristic (2) suggests that the particles should have relatively strong exploration ability so as to improve their ability to jump out of the local optimum. As we will see later in Sects. 4.2, 4.3 and 5, we may achieve this goal by selecting appropriate values of parameters.

The ratio $|F|/|S|$ described in characteristics (3) reflects the relative size of the feasible region. We conjecture that if we can effectively utilize this kind of knowledge in our algorithm to appropriately modify the velocity of each particle, then the optimization ability of our algorithm might be improved.

4.2 Self-adaptive velocity particle swarm optimization

Based on the analysis above, we modify the update equation of velocity in the following way. In (6), we set $c_1 = c_2 = 1$ and $r_1 = 1 - r_2$, thus the newly gained velocity term would cause particle i to fly toward a position between p_{gd} and p_{id}. Therefore, particle i will not hurtle too far away from the feasible region. At the same time, we replace the modification term ωv_{id}^t by $\omega|p_{i'd}(t) - p_{id}(t)|\text{sign}(v_{id}(t))$, where $\text{sign}(v_{id}(t))$ represents the sign of $v_{id}(t)$, which indicates the flying direction of $v_i(t)$ in the d-th dimension. In other words, the previous experience in velocity of particle i is only restricted to the flying direction, while the magnitude of it is determined by $\omega|p_{i'd}(t) - p_{id}(t)|$ according to the effect of feasible region, where ω is a parameter, and i' is a uniform random integer in the range $[1, N]$, denoted as $i' \in intU[1, N]$. Note that from the above assumption, both p_{id} and $p_{i'd}$ are close to or within the feasible region, thus $|p_{i'd}(t) - p_{id}(t)|$ roughly reflects the size of the feasible region. Therefore, with $\omega|p_{i'd}(t) - p_{id}(t)|$, particle i will not deviate too far from the feasible region; moreover, the value of $|p_{i'd}(t) - p_{id}(t)|$ can self-adaptively vary with the changes of the search scope of the swarm.

According to the discussion above, we derive our improved PSO algorithm, which we refer to as self-adaptive velocity PSO (SAVPSO for short). In SAVPSO, the swarm is manipulated according to the following update equations:

$$\begin{aligned} v_{id}(t+1) &= \omega|p_{i'd}(t) - p_{id}(t)|\text{sign}(v_{id}(t)) \\ &\quad + r(p_{id}(t) - x_{id}(t)) + (1-r)(p_{gd}(t) - x_{id}(t)) \qquad (8) \\ x_{id}(t+1) &= x_{id}(t) + v_{id}(t+1) \qquad (9) \end{aligned}$$

where $r \in U[0, 1]$, $i' \in intU[1, N]$, ω is a scaling parameter, and $\text{sign}(v_{id}(t))$ is the sign of $v_{id}(t)$.

4.3 Parameter analysis

In this subsection, we make an intuitive analysis of the parameter ω or c.

Comparing (6) with (8), we see that in the right hand of (8), the term $r(p_{id}(t) - x_{id}(t)) + (1-r)(p_{gd}(t) - x_{id}(t))$ is designed such that the particle can focus its search in the approximate region between p_i and p_g, while the term $\omega|p_{i'd}(t) - p_{id}(t)|\text{sign}(v_{id}(t))$ is designed

such that the corresponding velocity reflects the impact of constraints or utilizes the knowledge about the feasible region. Note that at certain stage in the search process, under the impact of DOCHM, the best positions of the particles will lie within and/or around the feasible region, then $|p_{i'd}(t) - p_{id}(t)|$ may roughly reflect the size of the feasible region from a point-of-view of statistics, and ω is a scaling parameter. Therefore, if we take $\omega = 1$, then speed $\omega|p_{i'd}(t) - p_{id}(t)|$ roughly matches the size of the feasible region. If we take $\omega > 1$, then speed $\omega|p_{i'd}(t) - p_{id}(t)|$ is expanded and thus the search scope of the swarm is enlarged, hence the exploration ability of the swarm is improved, but the convergence speed is lowered. If we take $\omega < 1$, then speed $\omega|p_{i'd}(t) - p_{id}(t)|$ is reduced and thus the search scope of the swarm shrinks, the exploitation ability of the swarm is improved, and the algorithm converges fast but is prone to get trapped into local optimum.

To obtain the global optimum, the particles need not only strong exploration ability to jump out of the local optimum, but also good exploitation ability so as to pinpoint the global optimum within the feasible region. Therefore, according to the above analysis of the meaning of ω, we think that it is reasonable to strike a balance between exploitation and exploration ability of SAVPSO. To do so, we may simply take $\omega = 1$. Or alternatively, we may set $\omega = cr_3$, where c is a parameter and $r_3 \in U[0, 1]$. Note that if $c = 2$, then $\omega = 2r_3$, and the mean value of ω is 1. If $c < 2$, then the mean value of $\omega < 1$.

As we will see Sect. 5, our algorithm with $c = 2$ can achieve substantially good results for those problems with relatively large value of $|F|/|S|$. However, this is not the case for the problems with relatively small $|F|/|S|$. Therefore, according to the analysis of parameter ω, we propose that if we take a smaller value of c, say $c = 1$, then the performance of our algorithm would be improved for the problems with relatively small value of $|F|/|S|$. In fact, additional experimental results (see Table 4 in Sect. 5) indicate that $c = 1$ (in this case the mean value of $\omega = 1/2 < 1$) indeed improves the performance of our algorithm for these problems, which is consistent with our parameter analysis.

4.4 Constraint-handling method

DOCHM is a PSO-based constraint-handling technique recently proposed by us in [19]. Through defining a distance function $\Phi(x)$, DOCHM converts the original problem into a bi-objective optimization problem $min(\Phi(x), f(x))$, where $\Phi(x)$ is treated as the first objective and $f(x)$ the second one. There are several ways to construct the distance function $\Phi(x)$ with respect to the feasible region, one simple way is as follows:

$$\Phi(x) = \sum_{i=1}^{q} \max\{0, g_i(x)\} + \sum_{j=q+1}^{m} \max\{0, |h_j(x)| - \delta\} \tag{10}$$

Clearly, $\Phi(x)$ is the sum of constraint violations, $\Phi(x) \geq 0$, and thus $\Phi(x) = 0$ for $\forall\, x \in F$. Moreover, all the optimal solutions of $\Phi(x)$ constitute the feasible region F of the original problem. Clearly, flying the particles towards the feasible region is equivalent to optimizing $\Phi(x)$, and thus the aforementioned characteristic (1) can be reflected by DOCHM.

It is worth noting that although $\Phi(x)$ takes on a form of penalty function, it is not used as it is defined in the conventional penalty function method. $\Phi(x)$ is merely used to determine wether or not a particle is within the feasible region and how close a particle is to the feasible region. It can be seen from (10) that no additional parameter is involved in DOCHM. To be specific, the dynamic-objective constraint-handling method works in the following way. The auxiliary objective function $\Phi(x)$ and the real objective function $f(x)$ constitute the two functions to be optimized. If a particle lies outside the feasible region, the particle will

Table 1 Pseudocode of DOCHM

Procedure for calculating p_i and p_g

set $\Phi_{ibest} = \Phi(p_i)$, $f_{ibest} = f(p_i)$, $\Phi_i = \Phi(x_i^k)$
If $\Phi_i < \Phi_{ibest}$ **Then** $p_i \leftarrow x_i^k$, $\Phi_{ibest} \leftarrow \Phi_i$ **End**
If $\Phi_i = 0$ and $\Phi_{ibest} = 0$ **Then**
  $f_i = f(x_i^k)$
  If $f_i \le f_{ibest}$ **Then** $p_i \leftarrow x_i^k$, $f_{ibest} \leftarrow f_i$
End
End
set $\Phi_{gbest} = \Phi(p_g)$, $f_{gbest} = f(p_g)$
If $\Phi_i < \Phi_{gbest}$ **Then** $p_g \leftarrow x_i^k$, $\Phi_{gbest} \leftarrow \Phi_i$ **End**
If $\Phi_i = 0$ and $\Phi_{gbest} = 0$ **Then**
  If $f_i \le f_{gbest}$ **Then** $p_g \leftarrow x_i^k$, $f_{gbest} \leftarrow f_i$
End

take $\Phi(x)$ as its optimization objective. Otherwise, the particle will instead optimize the real objective function $f(x)$. During the optimization process, if a particle leaves the feasible region, then it will once again optimize $\Phi(x)$. Therefore, the particle have the ability to dynamically adjust their optimization objectives independently of one another. Additionally, although the main aim of DOCHM is to drive the particles into the feasible region, it do not strictly confine the particles within the feasible region. This is advantageous to improving the particles' exploration ability. The procedure of DOCHM is described in Table 1.

DOCHM is a generic constraint-handling technique in the sense that it can be incorporated into various kinds of PSO-based algorithms. DOCHM mainly focus on efficiently reflecting the impact of characteristic (1) of the feasible region on the flying pattern of the particles, whereas SAVPSO aims at improving the optimization behavior within the feasible region. This can be seen from the experiments in Sect. 5.

4.5 The integrated SAVPSO for solving COPs

In some cases, the boundaries of the feasible region may locate very close to the parametric boundaries x_d^l and/or x_d^u of the search space. Therefore, during the solution process, it would be likely to occur that some particles near the boundaries of the feasible region violate the parametric constraints. In order to overcome this problem, we adopt the technique that we have used in [19], to randomly re-evaluate $x_{id}(t)$ such that $x_{id}(t)$ falls between the mean value $\bar{x}_d(t)$ of the d-th components of all particles and the parametric bounds on dimension d, that is

$$x_{id}(t) = \begin{cases} \bar{x}_d(t) + ar_4(x_d^l - \bar{x}_d(t)), & \text{if } x_{id}(t) < x_d^l \\ \bar{x}_d(t) + ar_4(x_d^u - \bar{x}_d(t)), & \text{if } x_{id}(t) > x_d^u \end{cases} \tag{11}$$

where $\bar{x}_d(t) = \left(\sum_{i=1}^{N} x_{id}(t)\right)/N$, $r_4 \in U[0, 1]$, and a is a constant number in the range $[0, 1]$. To make this technique applicable to general problems, we set $a = 1$ in the experiments conducted in this paper.

Table 2 Pseudocode of the Integrated SAVPSO

create and initialize an n-dimensional swarm
For $k = 0$ to I_{max}
  $\bar{x}_d(t) = \left(\sum_{i=1}^{N} x_{id}(t)\right)/N, d = 1, 2, \ldots, n$
  For $i = 1$ to N
    For $d = 1$ to n
      $x_{id}(t+1) = x_{id}(t) + v_{id}(t+1)$
      $v_{id}(t+1) = \omega|p_{i'd}(t) - p_{id}(t)|\text{sign}(v_{id}(t) + r(p_{id}(t) - x_{id}(t))+$
        $(1-r)(p_{gd}(t) - x_{id}(t))$
      If $x_{id}(t+1) > x_d^u$ **Then** $x_{id}(t+1) = \bar{x}_d + ar_4(x_d^u - \bar{x}_d)$
      End
      If $x_{id}(t+1) < x_d^l$ **Then** $x_{id}(t+1) = \bar{x}_d + ar_4(x_d^l - \bar{x}_d)$
      End
    End
    Call Procedure for Calculating $p_i(t+1)$ and $p_g(t+1)$
  End
End

In summary, Table 2 describes the pseudo-code of our integrated SAVPSO algorithm, where I_{max} is the maximum number of iterations. Note that the integrated SAVPSO incorporates DOCHM as a component of its search mechanism.

5 Experiments and discussions

To evaluate the performance of the proposed algorithm, we conducted a series of experiments on the well known Michalewicz' benchmark functions [10] extended by Runarsson and Yao [16]. These test functions selected include characteristics that are representative of what can be considered "difficult" global optimization problems for an evolutionary algorithm. Moreover, we compared our results with respect to some most recently proposed PSO-based algorithms enhanced with some additional operators and constraint-handling mechanisms [4,12,13,18,25].

The main characteristics of these benchmark functions are summarized in Table 3, for detailed function information see Appendix at the end of this paper. In Table 3, each value in the third column indicated by $|F|/|S|$ is an estimate of the ratio between the feasible region and the entire of search space, where $|F|$ is the number of feasible solutions and $|S|$ is the total number of solutions randomly generated. In this work, $|S| = 1{,}000{,}000$.

Problems `g02`, `g03`, `g08`, and `g12` are maximization problems. They were converted into minimization problems using $-f(x)$. All equality constraints $h_j(x) = 0$ have been transformed into inequality constraints $|h_j(x)| - \delta \leq 0$, using the degree of violation $\delta = 0.001$. A total of 50 particles were employed, the maximum number of iterations I_{max} were set to 1000 per run, and 30 independent runs of our algorithm were executed for each problem. The parameter c in Table 3 is the constant in the dynamic selection strategy $\omega = cr_3$ for the parameter ω in Eq. 8. The value of c is selected according to the method discussed in

Table 3 Main characteristics of the benchmark functions

Problem	n	Function	$\lvert F\rvert/\lvert S\rvert(\%)$	c	LI	NI	LE	NE	Active
g01	13	quadratic	0.0003	2	9	0	0	0	6
g02	20	nonlinear	99.9973	2	1	1	0	0	1
g03	10	polynomial	0.0026	1	0	0	0	1	1
g04	5	quadratic	27.0079	2	0	6	0	0	2
g05	4	cubic	0.0000	1	2	0	0	3	3
g06	2	cubic	0.0057	1	0	2	0	0	2
g07	10	quadratic	0.0000	2	3	5	0	0	6
g08	2	nonlinear	0.8581	2	0	2	0	0	0
g09	7	polynomial	0.5199	2	0	4	0	0	2
g10	8	linear	0.0020	2	3	3	0	0	3
g11	2	quadratic	0.0973	2	0	0	0	1	1
g12	3	quadratic	4.7697	2	0	9^3	0	0	0
g13	5	exponential	0.0000	1	0	0	0	3	3

LI: linear inequality, NI: nonlinear inequalities, LE: linear equalities, NE: nonlinear equality, active: the number of active constraints at optimum

Sect. 4, with g01 and g07 being two exceptions. Since these two functions tends to trap the algorithm into local minima, we set $c = 2$. All experiments were performed in MATLAB.

Table 4 summarizes the experimental results obtained using our SAVPSO algorithm with the above experimental settings, where "Opt" represents the known "optimal" solution for each problem, "Std" stands for "standard deviation" of the obtained statistics for the 30 independent runs. It should be pointed out that the obtained solutions are all feasible solutions, this in fact is guaranteed by the inherent search mechanism of our algorithm. It can be seen from Table 4 that SAVPSO algorithm generates considerably accurate solutions for most of the problems and the standard deviations are quite small. Note that all equality constraints have been converted into inequality constraints using a degree of violation $\delta = 0.001$. Due to this approximation, some solutions might be better than the known optimal solutions. For example, the best objective value of g05 given by SAVPSO is 5126.484153, which is better than the optimum 5126.4981. Other examples include problems g03, g11 and g13.

Problem g01 has two local minima -13.000 and -12.453125, only 3 out of 30 independent runs were trapped into the two local minima. Problems g05, g10 and g13 are among those difficult-to-solve problems for evolutionary algorithms. Specifically, g10 is a difficult problem for penalty function approach, while g05 and g13 involves equality constraints.

The above experiment have shown that our algorithm can achieve substantially good results for those problems with relatively large value of $|F|/|S|$. But this is not the case for the problems with relatively small $|F|/|S|$ (e.g. g03, g05 and g13). Therefore, according to the previous analysis of parameter ω, we conducted additional experiment on these problems with $c = 1$ (thus the mean value of ω is equal to 1/2), and the corresponding results are listed in Table 4. It can be seen from Table 4, this parameter setting indeed improves the performance of our algorithm for these problems, which is consistent with our previous parameter analysis.

Note from Table 3 that the size of feasible region of problems g05 and g13 is extremely small (practically zero), this will deteriorate the performance of the PSO-based algorithms.

Table 4 Experimental results on 13 benchmark functions using SAVPSO with $I_{max} = 1000$; 30 Independent runs were carried out; the row indicated by * lists the corresponding results with $\omega = 1$

Pro	c	Opt	Best	Median	Mean	Worst	Std
g01	2	−15	−15	−15	−14.7151	−12.4531	0.74
g02	2	−0.803619	−0.803443	−0.747502	−0.740577	−0.631598	0.042
g03	2	−1	−0.7706	−0.4534	−0.4964	−0.2413	0.14
	1		−1.0048	−1.0038	−1.0034	−0.9976	0.0017
g04	2	−30665.539	−30665.539	−30665.539	−30665.539	−30665.539	0
g05	2	5126.4981	5130.1575	5589.1126	5562.4528	6111.5135	324.42
	1		5126.4886	5224.6825	5363.5092	6051.1367	296.08
	*		5126.4841	5126.4842	5202.3627	5520.1467	112.97
g06	2	−6961.81388	−6961.81388	−6961.81388	−6961.81388	−6961.81387	0.0000013
	1		−6961.81388	−6961.81388	−6961.81388	−6961.81388	0
g07	2	24.306	24.319	24.887	24.989	26.194	0.55
g08	2	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	0
g09	2	680.630	680.632	680.653	680.655	680.699	0.018
g10	2	7049.3307	7087.9553	7400.7132	7439.7345	8165.1973	246.67
	*		7054.1256	7174.8991	7173.2661	7335.2477	84.53
g11	2	0.75	0.749000	0.749000	0.749002	0.749021	0.0000039
g12	2	−1	−1	−1	−1	−1	0
g13	2	0.0539498	0.3092	0.7722	1.0639	10.1580	1.72
	1		0.300316	0.851943	1.542039	19.486249	3.43
	*		0.0538666	0.461471	0.552753	1.856102	0.42

To overcome this difficulty, we carried out additional experiments with $\omega = 1$ and other experimental settings remaining unchanged, generating significantly better results, which are included in the rows indicated by symbol * in Table 4. The results suggests that if the feasible region is very small, then a search mechanism with a constant value of ω may perform better that with a random value of ω. This is because randomized ω imposes so much randomness to the movement of particles that it is difficult to focus the search on the feasible region, thus impairing the optimization ability of the algorithm used.

Our comparison results gainst the aforementioned PSO-based algorithms is presented in Tables 5 and 6.

As is shown in Table 5, in comparison with Toscano and Coello's algorithm [13], denoted CHMPSO in the rest of this paper, our results are much better for all but two problems (`g01` and `g02`) in terms of best, mean, and worst results. Even in the two exceptions, our algorithm found optimal solution of one problem (`g01`) and "better" best result for the other problem (`g02`) than that obtained by CHMPSO. Regarding the computational cost measured in the number of fitness (or objective) function evaluations (FFE), we have used a maximum of only $50 \times 1000 = 50,000$ FFE for each problem, while $40 \times 8500 = 340,000$ (40 particles runing for 8500 iterations) FFE were performed in CHMPSO.

For problems `g04, g08, g11, g12` and even `g06`, our algorithm has consistently found the optimal solution for all 30 runs, whereas CHMPSO did not find any for problems `g04` and `g06`, and consistently found optimal solutions only for problems `g08` and `g12`.

For problems `g05, g10` and `g13`, our algorithm significantly outperformed CHMPSO.

Table 5 Comparison between SAVPSO and CHMPSO [13]

Pro	Opt	SAVPSO Best	CHMPSO	SAVPSO Mean	CHMPSO	SAVPSO Worst	CHMPSO
g01	−15	−15	−15	−14.715104	−15	−12.453125	−15
g02	−0.803619	−0.803443	−0.803432	−0.740577	−0.790406	−0.631598	−0.750393
g03	−1	−1.004814	−1.004720	−1.003367	−1.003814	−0.997588	−1.002490
g04	−30665.539	−30665.538672	−30665.500	−30665.538672	−30665.500	−30665.538672	−30665.500
g05	5126.4981	5126.484153	5126.6400	5202.362681	5461.081333	5520.146710	6104.750000
g06	−6961.81388	−6961.813875	−6961.8100	−6961.813875	−6961.8100	−6961.813869	−6961.8100
g07	24.306	24.318980	24.351100	24.988731	25.355771	26.194272	27.316800
g08	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
g09	680.630	680.632332	680.638000	680.655378	680.852393	680.699042	681.55300
g10	7049.3307	7054.125620	7057.5900	7173.266104	7560.047857	7335.247682	8104.310000
g11	0.75	0.749000	0.749999	0.749002	0.750107	0.749021	0.752885
g12	−1	−1	−1.000000	−1	−1.000000	−1	−1.000000
g13	0.0539498	0.0538666	0.068665	0.552753	1.716426	1.856102	13.66950

Table 6 Comparison of our SAVPSO with PSO [12], DEPSO [18], CPSO [4], and PESO [25] in terms of best results

Pro	Opt	SAVPSO	PSO	DEPSO	CPSO	PESO
g01	−15	−15	−15	−15.000	/	15.000000
g02	−0.803619	−0.803443	/	−0.7868	/	−0.792608
g03	−1	−1.004814	/	−1.0050	/	−1.005010
g04	−30665.539	−30665.538672	−30665.5	−30665.5	−30664.7	−30665.538672
g05	5126.4981	5126.484153	–	5126.484	/	5126.484154
g06	−6961.81388	−6961.813875	−6961.7	−6961.81	/	−6961.813876
g07	24.306	24.318980	24.44201	24.586	24.80818	24.306921
g08	−0.095825	−0.095825	−0.095825	−0.095825	/	−0.095825
g09	680.630	680.632332	680.657	680.641	680.667	680.630057
g10	7049.3307	7054.125620	7131.01	7267.4	7114.84	7049.459452
g11	0.75	0.749000	/	0.74900	/	0.749000
g12	−1	−1	−1.000000	/	/	−1.000000
g13	0.0539498	0.0538666	/	/	/	0.081498

"/" indicates that the corresponding problem was not tested, "–" indicates that the problem was tested but no valid results were obtained

For all the problems involving equality constraints (`g03, g05, g11,` and `g13`) in the benchmark, our algorithm has found even better results than the optimal solutions. This is resulted from approximating equalities by inequalities, as previously mentioned.

Table 6 summarizes the comparison of our algorithm against four other recently proposed PSO-based algorithms in terms of best results. Of these four algorithms, Hu and Eberhart's PSO requires an initialization of particles inside the feasible region, which may have a very high computational cost in some cases [12], resulting in its prohibition in real-world applications. Zhang and Xie's DEPSO [18] is a hybrid PSO algorithm that make use of a reproduction operator similar to that used in differential evolution [26]. The maximum number of FFE of DEPSO in [18] is 1,400,000. PESO [25] proposed by Zavala et al. is also a hybrid PSO algorithm which incorporates two perturbation operators, expending 350,000 FFE for each problem in the experiment in [25]. CPSO proposed by Dong et al. [4] embeds a constraint fitness priority-based ranking metheod and a dynamic neighborhood operator into standard PSO algorithm, the relevant experiments with CPSO costed a maximum of 50,000 FFE.

As can be seen from Table 6, our algorithm outperforms or matches the above mentioned PSO, DEPSO, and CPSO on the benchmark functions in terms of best results. In comparison with PESO, our best results are better for problems `g02` and `g13`, slightly worse for problems `g07, g09` and `g10`, and comparable for other problems in the benchmark.

The experimental results above show that the combined effect of DOCHM and SAVPSO contributes to the performance of the integrated SAVPSO, i.e., DOCHM + SAVPSO. The developments in [19] have shown that DOCHM does contribute to the good performance of the underlying algorithm. In order to reveal the separate contribution of SAVPSO to the superior performance of the DOCHM + SAVPSO, we compare DOCHM + SAVPSO with DOCHM + SPSO and DOCHM + RVPSO [19]. Here in SPSO we set $c_1 = c_2 = 1.49445$ and $\omega = 0.5 + \frac{\text{rand}(\cdot)}{2}$, where $\text{rand}(\cdot) \in U[0, 1]$. The comparison results are described in Table 7. Clearly, it can be seen from Table 7 that DOCHM + SAVPSO performs much better than DOCHM + SPSO on all the benchmark functions, this implies that SAVPSO makes a

Table 7 Comparison of SAVPSO with SPSO and RVPSO [19]

fun	Opt	DOM+	Best	Median	Mean	Worst	Std
g01	−15	SAVPSO	−15	−15	−14.7151	−12.4531	7.4e−1
		RVPSO	−15	−15	−14.4187	−12.4531	8.5e−1
		SPSO	−15	−15	−14.6094	−11.8281	9.1e+1
g02	−0.803619	SAVPSO	−0.803443	−0.747502	−0.740577	−0.631598	4.2e−2
		RVPSO	−0.664028	−0.380820	−0.413257	−0.259980	1.2e−1
		SPSO	−0.803578	−0.710560	−0.700890	−0.483212	7.6e−2
g03	−1	SAVPSO	−1.0048	−1.0038	−1.0034	−0.9976	1.7e−2
		RVPSO	−1.0050	−1.0051	−1.0025	−0.9334	1.3e−2
		SPSO	−1.0042	−0.9979	−0.9753	−0.7771	4.7e−2
g04	−30665.539	SAVPSO	−30665.539	−30665.539	−30665.539	−30665.539	0
		RVPSO	−30665.539	−30665.539	−30665.539	−30665.539	0
		SPSO	−30665.539	−30665.539	−30665.539	−30665.539	3.3e−9
g05	5126.4981	SAVPSO	5126.4842	5126.4842	5202.3627	5520.1467	1.1e+2
		RVPSO	5126.4842	5127.0038	5241.0549	5708.2250	1.8e+2
		SPSO	5126.9691	5193.0519	5233.9116	5625.5668	1.2e+2
g06	−6961.81388	SAVPSO	−6961.81388	−6961.81388	−6961.81388	−6961.81388	0
		RVPSO	−6961.81388	−6961.81388	−6961.81388	−6961.81388	0
		SPSO	−6961.81384	−6961.81324	−6961.81269	−6961.80940	1.3e−3
g07	24.306	SAVPSO	24.319	24.887	24.989	26.194	5.5e−1
		RVPSO	24.306	24.307	24.317	24.385	2.4e−2
		SPSO	24.431	25.904	25.988	28.350	1.1e+0
g08	−0.095825	SAVPSO	−0.095825	−0.095825	−0.095825	−0.095825	0
		RVPSO	−0.095825	−0.095825	−0.095825	−0.095825	0
		SPSO	−0.095825	−0.095825	−0.095825	−0.095825	0
g09	680.630	SAVPSO	680.632	680.653	680.653	680.699	1.8e−2
		RVPSO	680.630	680.630	680.630	680.630	0
		SPSO	680.633	680.659	680.667	680.758	3.0e−2
g10	7049.3307	SAVPSO	7054.1256	7174.8991	7173.2661	7335.2477	8.4e+2
		RVPSO	7049.2480	7049.2483	7049.2701	7049.5969	7.9e−2
		SPSO	7070.2635	7296.8247	7356.0522	7874.0740	1.9e+2
g11	0.75	SAVPSO	0.749	0.749	0.749	0.749	3.9e−7
		RVPSO	0.749	0.749	0.749	0.749	0
		SPSO	0.749	0.749	0.749	0.749	1.5e−5
g12	−1	SAVPSO	−1	−1	−1	−1	0
		RVPSO	−1	−1	−1	−1	0
		SPSO	−1	−1	−1	−1	0
g13	0.0539498	SAVPSO	0.0538666	0.461471	0.552753	1.856102	4.2e−1
		RVPSO	0.0538666	0.645928	0.681123	2.042892	4.0e−1
		SPSO	0.2767082	0.873415	1.021262	3.893345	7.9e−1

substantial contribution to the superior performance of DOCHM + SAVPSO since the two algorithms use the same constraint-handling method DOCHM. Besides, from Table 7 we can see that both DOCHM + SAVPSO and DOCHM + RVPSO perform sufficiently good on all the 13 benchmarks. Both of them achieved optimal solutions on problems `g04, g06, g08, g11` and `g12` in terms of best, median, mean and worst results. DOCHM + SAVPSO performs better than DOCHM + RVPSO on almost all other problems except `g07, g09` and `g10`. This indicates that SAVPSO acquires additional advantages over RVPSO by getting along the line of thinking proposed in this paper.

6 Conclusions and future work

This paper has analyzed the potential impact of constraints (or the feasible region) on the search pattern of the particles in PSO, and then based on these investigations, we has primarily investigated the methods for directly incorporating this impact into the inherent search mechanism PSO, thereby presenting an improved PSO algorithm, i.e., SAVPSO. To deal with constraints effectively and efficiently, we has integrated into SAVPSO our recently proposed constraint-handling technique DOCHM. The validity of our idea and strategy are justified via a series of experiments on a well-known benchmark suite, and the comparison results of SAVPSO with other recently proposed PSO algorithms have shown that our integrated SAVPSO performs better than or rather competitive with these algorithms on the benchmark functions. Our developments in in this work indicate that both SAVPSO and DOCHM contribute to the superior performance of the integrated SAVPSO.

The results obtained in this paper provide us some insight into understanding and improving search mechanism of PSO from the constrained optimization perspective. Several interesting directions needs to be explored in the future work. For example, we will study in our future work the approach to more effectively incorporate the impact of constraints into the inherent search mechanism of PSO which, at the same time, remains the advantageous features of PSO such as simplicity, fast convergence and easy implementation. Evaluating the performance of the improved algorithm on a wider variety of benchmark functions is also an interesting work. In addition, a lot of work needs to be done to further improve the global optimization ability of our algorithm.

Acknowledgements The authors are grateful to Dr. J. Kennedy for answering their questions about the standard Particle Swarm Optimization, and to Profs. Enyu Yao, Kai Yan and Shitong Wang for their encouragement and helpful discussions. This research is supported by Scientific Research Fund of Jiangnan University (No. 0003182) and National Natural Science Foundation of China (No. 10371028).

Appendix

All benchmark functions used for our experiments are summarized here for completeness. This is the well known Michalewicz's benchmark [10] extended by Runarsson and Yao [16].

`1. g01`

$$\text{Minimize } f(x) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

subject to

$$
\begin{aligned}
g_1(x) &= 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0\\
g_2(x) &= 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0\\
g_3(x) &= 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0\\
g_4(x) &= -8x_1 + x_{10} \leq 0\\
g_5(x) &= -8x_2 + x_{11} \leq 0\\
g_6(x) &= -8x_3 + x_{12} \leq 0\\
g_7(x) &= -2x_4 - x_5 + x_{10} \leq 0\\
g_8(x) &= -2x_6 - x_7 + x_{11} \leq 0\\
g_9(x) &= -2x_8 - x_9 + x_{12} \leq 0
\end{aligned}
$$

where the bounds are $0 \leq x_i \leq 1$ $(i = 1, \ldots, 9)$, $0 \leq x_i \leq 100$ $(i = 10, 11, 12)$ and $0 \leq x_{13} \leq 1$. The global optimum is at $x^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$ where $f(x^*) = -15$. Constraints g_1, g_2, g_3, g_4, g_5 and g_6 are active.

2. g02

$$
\text{Maximize } f(x) = \left| \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right|
$$

subject to

$$
\begin{aligned}
g_1(x) &= 0.75 - \prod_{i=1}^{n} x_i \leq 0\\
g_2(x) &= \sum_{i=1}^{n} x_i - 7.5n \leq 0
\end{aligned}
$$

where $n = 20$ and $0 \leq x_i \leq 10 (i = 1, \ldots, n)$. The global maximum is unknown; the best reported solution is $f(x) = 0.803619$. Constraint g_1 is close to being active ($g_1 = -10^{-8}$).

3. g03

$$
\text{Maximize } f(x) = (\sqrt{n})^n \prod_{i=1}^{n} x_i
$$

subject to

$$
h(x) = \sum_{i=1}^{n} x_i^2 - 1
$$

where $n = 10$ and $0 \leq x_i \leq 1$ $(i = 1, \ldots, n)$. The global maximum is at $x^* = 1/\sqrt{n}$ $(i = 1, \ldots, n)$ where $f(x^*) = 1$.

4. g04

$$
\text{Minmize } f(x) = 5.3578547x_2^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141
$$

subject to

$$
\begin{aligned}
g_1(x) &= 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92 \le 0\\
g_2(x) &= -85.334407 - 0.0056858x_2x_5 - 0.0006262x_1x_4 + 0.0022053x_3x_5 \le 0\\
g_3(x) &= 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3x^2 - 110 \le 0\\
g_4(x) &= -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \le 0\\
g_5(x) &= 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0010985x_3x_4 - 25 \le 0\\
g_6(x) &= -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \le 0
\end{aligned}
$$

where $78 \le x_1 \le 102$, $33 \le x_2 \le 45$ and $27 \le x_i \le 45 (i = 3, 4, 5)$. The optimum solution is $x^* = (78, 33, 29.995256025682, 45, 36.775812905788)$ where $f(x^*) = -30665.539$. Constraints g_1 and g_6 are active.

5. g05

$$\text{Minimize } f(x) = 3x_1 + 0.000001x_1^3 + 2x_2 + (0.000002/3)x_2^3$$

subject to

$$
\begin{aligned}
g_1(x) &= -x_4 + x_3 - 0.55 \le 0\\
g_2(x) &= -x_3 + x_4 - 0.55 \le 0\\
h_3(x) &= 1000\sin(-x_3 - 0.25) + 1000\sin(-x_4 - 0.25) + 894.8 - x_1 = 0\\
h_4(x) &= 1000\sin(x_3 - 0.25) + 1000\sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0\\
h_5(x) &= 1000\sin(x_4 - 0.25) + 1000\sin(x_4 - x_3 - 0.25) + 1294.8 = 0
\end{aligned}
$$

where $0 \le x_1 \le 1200$, $0 \le x_2 \le 1200$, $-0.55 \le x_3 \le 0.55$ and $-0.55 \le x_4 \le 0.55$. The best known solution is $x^* = (679.9453, 1026.067, 0.1188764, -0.3962336)$ where $f(x^*) - 5126.4981$.

6. g06

$$\text{Minimize } f(x) = (x_1 - 10)^3 + (x_2 - 20)^3$$

subject to

$$
\begin{aligned}
g_1(x) &= -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0\\
g_2(x) &= (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0
\end{aligned}
$$

where $13 \le x_1 \le 100$ and $0 \le x_2 \le 100$. The optimum solution is $x^* = (14.095, 0.84296)$ where $f(x^*) = -6961.81388$. Both constraints are active.

7. g07

$$
\begin{aligned}
\text{Minimize } f(x) = {} & x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2\\
& + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45
\end{aligned}
$$

subject to

$$
\begin{aligned}
g_1(x) &= -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0\\
g_2(x) &= 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0\\
g_3(x) &= -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0\\
g_4(x) &= 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0\\
g_5(x) &= 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0\\
g_6(x) &= x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leq 0\\
g_7(x) &= 0.5(x_1 - 8)^2 + x(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leq 0\\
g_8(x) &= -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0
\end{aligned}
$$

where $-10 \leq x_i \leq 10$ $(i = 1, \ldots, 10)$. The global optimum is $x^* = (2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927)$ where $f(x^*) = 24.3062091$. Constraints g_1, g_2, g_3, g_4, g_5 and g_6 are active.

8. g08

$$
\text{Maximize } f(x) = \frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1 + x_2)}
$$

subject to

$$
\begin{aligned}
g_1(x) &= x_1^2 - x_2 + 1 \leq 0\\
g_2(x) &= 1 - x_1 + (x_2 - 4)^2 \leq 0
\end{aligned}
$$

where $0 \leq x_1 \leq 10$ and $0 \leq x_2 \leq 10$. The optimum solution is located at $x^* = (1.2279713, 4.2453733)$ where $f(x^*) = 0.095825$. The solution lies within the feasible solution.

9. g09

$$
\begin{aligned}
\text{Minimize } f(x) = {} & (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2\\
& + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7
\end{aligned}
$$

subject to

$$
\begin{aligned}
g_1(x) &= -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \leq 0\\
g_2(x) &= -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0\\
g_3(x) &= -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0\\
g_4(x) &= 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0
\end{aligned}
$$

where $-10 \leq x_i \leq 10$ for $i = 1, \ldots, 7$. The global optimum is $x^* = (2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227)$ where $(x^*) = 680.6300573$. Constraints g_1 and g_4 are active.

10. g10

$$
\text{Minimize } f(x) = x_1 + x_2 + x_3
$$

subject to

$$g_1(x) = -1 + 0.0025(x_4 + x_6) \leq 0$$
$$g_2(x) = -1 + 0.0025(x_5 + x_7 - x_4) \leq 0$$
$$g_3(x) = -1 + 0.01(x_8 - x_5) \leq 0$$
$$g_4(x) = -x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333 \leq 0$$
$$g_5(x) = -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \leq 0$$
$$g_6(x) = -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leq 0$$

where $100 \leq x_1 \leq 10000$, $1000 \leq x_i \leq 10000$ $(i = 2, 3)$, and $10 \leq x_i \leq 1000$ $(i = 4, \ldots, 8)$. The global optimum is $x^* = (579.3167, 1359.943, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979)$ where $f(x^*) = 7049.3307$. There constraints are active ($g_1, g_2,$ and g_3).

11. g11

$$\text{Minimize } f(x) = x_1^2 + (x_2 - 1)^2$$

subject to

$$h(x) = x_2 - x_1^2 = 0$$

where $-1 \leq x_1 \leq 1$ and $-1 \leq x_2 \leq 1$. The optimum solution is $x^* = (\pm 1/\sqrt{2}, 1/2)$ where $f(x^*) = 0.75$.

12. g12

$$\text{Maximize } f(x) = (100 - (x_1 - 5)^2 - (x_2 - 5)^2 - (x_3 - 5)^2)/100$$

subject to

$$g(x) = (x_1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \leq 0$$

where $1 \leq x_i \leq 10$ $(i = 1, 2, 3)$ and $p, q, r = 1, 2, \ldots, 9$. The feasible region of the search space consists of 9^3 disjointed spheres. A point (x_1, x_2, x_3) is feasible if and only if there exist p, q, r such that the above inequality holds. The optimum is located at $x^* = (5, 5, 5)$ where $f(x^*) = 1$. The solution lies within the feasible region.

13. g13

$$\text{Minimize } f(x) = e^{x_1x_2x_3x_4x_5}$$

$$h_1(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$$
$$h_2(x) = x_2x_3 - 5x_4x_5 = 0$$
$$h_3(x) = x_1^3 + x_2^3 + 1 = 0$$

where $-2.3 \leq x_i \leq 2.3$ $(i = 1, 2)$ and $-3.2 \leq x_i \leq 3.2 (i = 3, 4, 5)$. The optimum solution is $x^* = (-1.717143, 1.595709, 1.827247, -0.7636413, -0.763645)$ where $f(x^*) = 0.0539498$.

References

1. Floudas, C.A., Pardalos, P.M.: A collection of test problems for constrained global optimization algorithms. Lect. Notes Comput. Sci. **455**, Springer-Verlag (1987)
2. Himmelblau, D.M.: Applied Nonlinear Programming. McGraw-Hill (1972)

3. Coello Coello, C.A.: Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. Comput. Meth. Appl. Mech. Eng. **191**, 1245–1287 (2002)
4. Dong, Y., Tang, J.-F., Xu, B.-D., Wang, D.-W.: An application of swarm optimization to nonlinear programming. Comput. Math. Appl. **49**, 1655–1668 (2005)
5. Hu, X., Eberhart, R.C., Shi, Y.: Engineering optimization with particle swarm. In: Proceedings of 2003 IEEE Swarm Intelligence Symposium, pp. 53–57 (2003)
6. Parsopoulos, K.E., Vrahatis, M.N.: Unified particle swarm optimization for solving constrained engineering optimization problems. Lect. Notes Comput. Sci. **3612**, 582–591 (2005)
7. Yeniay, Ö.: Penalty function methods for constrained optimization with genetic algorithms. Math. Comput. Appl. **10**, 45–56 (2005)
8. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948 (1995)
9. Eberhart, R.C., Kennedy, J.: A new Optimizer using particle swarm theory. In: Proceedings of 6th International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
10. Michalewicz, Z., Schoenauer, M.: Evolutionary algorithms for constrained parameter optimization problems. Evol. Comput. **4**, 1–32 (1996)
11. Coello Coello, C.A.: Treating constraints as objectives for single objective evolutionary computations. Eng. Optim. **32**, 275–308 (2000)
12. Hu, X., Eberhart, R.C.: Solving constrained nonlinear optimization problems with particle swarm optimization. In: Proceedings of 6th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2002), Orlando, USA (2002)
13. Toscano, G., Coello Coello, C.A.: A constraint-handling mechanism for particle swarm optimization. In: Proceedings of the 2004 Congress on Evolutionary Computation, June, IEEE, pp. 1396–1403 (2004)
14. Sedlaczek, K., Eberhart, P.: Constrained particle swarm optimization of mechanical systems. In: Proceedings of Sixth World Congresses of Structural and Multidisciplinary Optimization. Rio de Janeiro, Brizil (2005)
15. Parsopoulos, K.E., Vrahatis, M.N.: Particle swarm optimization method for constrained optimization problems. In: Proceedings of the Euro-International Symposium on Computational Intelligence (E-ISCI 2002) (2002)
16. Runarsson, T.P., Yao, X.: Stochastic ranking for constrained evolutionary optimization. IEEE Trans. Evol. Comput. **4**, 284–294 (2000)
17. Koziel, S., Michalewicz, Z.: Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evol. Comput. **7**, 19–44 (1999)
18. Zhang, W.-J., Xie, X.-F.: DEPSO: hybrid particle swarm with differential evolution operator. In: Proceedings of IEEE International Conference on Systems, Man and Cybernetics, October, IEEE, pp. 3816–3821 (2003)
19. Lu, H.Y., Chen, W.Q.: Dynamic-objective particle swarm optimization for constrained optimization problems. J. Comb. Optim. **12**, 409–419 (2006)
20. Kennedy, J.: Dynamic-probabilistic particle swarms. GECCO'05, June 2005, Washington, DC, USA, pp. 201–207 (2005)
21. Eberhart, R.C., Shi, Y.: Comparison between genetic algorithms and particle swarm optimization. In: Porto, V.W., Saravanan, N., Waagen, D., Eiben, A.E. (eds.) Evolutionary Programming, Vol. 7, pp. 611–616. Springer-Verlag, Berlin (1998)
22. Shi, Y., Eberhart, R.C. : Parameter selection in particle swarm optimization. In: Porto, V.W., Saravanan, N., Waagen, D., Eiben, A.E. (eds.) Evolutionary Programming, vol. 7, pp. 591–600. Springer-Verlag, Berlin (1998)
23. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proceeding of IEEE Conference on Evolutionary Computation, Anchorage, AK, pp. 69–73 (1998)
24. Shi, Y.: Particle swarm optimization. IEEE Neural Networks Society, February, pp. 8–13 (2004)
25. Muñoz Zavala, A.E., Hernández Aguirre, A., Villa Diharce, E.R.: Constrained optimization via particle evolutionary swarm optimization algorithm (PESO), GECCO'05, Washington, DC, USA, 25–27 June, pp. 209–216 (2005)
26. Storn, R., Price, K.: Differential evolution—a simple andd efficient heuristic for global optimization over continuous spaces. J. Global Optim. **11**, 341–359 (1997)