Noname manuscript No.
(will be inserted by the editor)

Constrained Optimization by Improved Particle Swarm Optimization with the Equivalent Penalty Coefficient Method

Tetsuyuki Takahama · Setsuko Sakai ·

Received: date / Accepted: date

Abstract The penalty function method has been widely used to solve constrained optimization problems. In the method, an extended objective function, which is the sum of the objective value and the constraint violation weighted by the penalty coefficient, is optimized. However, it is difficult to control the coefficient properly because the proper control depends on each problem. Recently, the equivalent penalty coefficient (EPC) method, which is a new adaptive penalty method for population-based optimization algorithms (POAs), has been proposed. The EPC method can be applied to POAs where a new solution is compared with the old solution. The EPC value, which makes the two extended objective values of the solutions the same, is used to control the coefficient. In this study, we propose to apply the EPC method to particle swarm optimization (PSO) where a new solution is compared with the best solution found so far. In order to improve the performance of constrained optimization, a mutation operation is also proposed. The proposed method is examined using two topologies of PSO. The advantage of the proposed method is shown by solving well-known constrained optimization problems and comparing the results with those obtained by PSO with a standard constraint-handling technique.

Keywords Constrained optimization · Particle swarm optimization · Equivalent penalty coefficient method · Population-based optimization algorithm

Tetsuyuki Takahama
Hiroshima City University, Hiroshima, Japan
Tel.: +81-82-830-1698
E-mail: takahama@hiroshima-cu.ac.jp

Setsuko Sakai
Hiroshima Shudo University, Hiroshima, Japan
Tel.: +81-82-830-1186
E-mail: setuko@shudo-u.ac.jp

1 Introduction

Constrained optimization problems, especially nonlinear constrained optimization problems, where objective functions are minimized under given constraints, are very important and frequently appear in the real world. There exist many studies on solving constrained optimization problems using population-based optimization algorithms (POAs) such as evolutionary algorithms (EAs) [1–3] and particle swarm optimization (PSO) [4]. POAs basically lack a mechanism to incorporate the constraints of a given problem in the fitness value of individuals. Thus, many studies have been dedicated to handle the constraints in POAs.

The penalty function method has been widely used for solving constrained optimization problems. In the method, an extended objective function is optimized where the function is defined by the sum of the objective value and the constraint violation weighted by the penalty coefficient. Feasible solutions can be found by increasing the penalty coefficient into infinity theoretically. However, it is difficult to control the coefficient properly because proper control of the coefficient varies in each problem and the search process. Recently, the equivalent penalty coefficient (EPC) method was proposed for adaptive control of the penalty coefficient in POAs [5]. An EPC value is defined in POAs where a new solution is compared with the old solution. For example, a child individual is compared with the parent individual in differential evolution (DE) and a new position after moving is compared with the personal best position found so far in PSO. The EPC value is the penalty coefficient value that makes the two extended objective values of the old solution and the new solution the same when the objective value and the constraint violation are in a trade-off relationship. In POAs, there are plural EPCs in a population in general. Search that gives priority to the objective value and the constraint violation is realized by selecting a small EPC and a large EPC, respectively. The adaptive control of the penalty coefficient can be realized by selecting an appropriate EPC value.

In this study, we propose to apply the EPC method to PSO, whereas in [5] the EPC method was applied to DE. In order to improve the performance of constrained optimization, a mutation operation is also proposed. The proposed method is examined using two topologies of PSO such as the fully-connected topology and the ring topology. The advantage of the proposed method is shown by solving well-known constrained optimization problems and comparing the results with those obtained by PSO with a standard constraint-handling technique.

In Section 2, constrained optimization problems are defined and constrained optimization methods including the penalty function method are briefly reviewed. PSO is explained in Section 3. The proposed method is described in Section 4. In Section 5, experimental results on constrained problems are shown and the results of the proposed method are compared with those of a standard method. Finally, conclusions are described in Section 6.

2 Related Works

2.1 Constrained Optimization Problems

The general constrained optimization problem with inequality, equality, upper bound and lower bound constraints is defined as follows:

$$\begin{aligned} &\text{minimize} \;\; f(\boldsymbol{x}), \\ &\text{subject to} \; g_j(\boldsymbol{x}) \le 0, \; j = 1, \ldots, q, \\ &\qquad\qquad h_j(\boldsymbol{x}) = 0, \; j = q+1, \ldots, m, \\ &\qquad\qquad l_i \le x_i \le u_i, \; i = 1, \ldots, D, \end{aligned} \tag{1}$$

where $\boldsymbol{x} = (x_1, x_2, \cdots, x_D)$ is a D dimensional vector of decision variables, $f(\boldsymbol{x})$ is an objective function, $g_j(\boldsymbol{x}) \le 0$ are q inequality constraints and $h_j(\boldsymbol{x}) = 0$ are $m - q$ equality constraints. The functions f, g_j and h_j are linear or nonlinear real-valued functions. The values u_i and l_i are the upper and lower bounds of x_i, respectively. The upper and lower bounds define the *search space* $\mathcal{S}$. All constraints define the *feasible region* $\mathcal{F}$. Feasible solutions exist in $\mathcal{F} \subseteq \mathcal{S}$.

2.2 Constrained Optimization Methods

POAs for constrained optimization can be classified into several categories according to the way the constraints are treated as follows [3]:

1. Constraints are only used to see whether a search point is feasible or not. Approaches in this category are usually called death penalty methods. In this category, the search process begins with one or more feasible points and continues to search for new points within the feasible region. When a new search point is generated and the point is not feasible, the point is repaired or discarded. When the feasible region is very small, generating initial feasible points is difficult and computationally demanding.
2. The constraint violation, which is the sum of the violation of all constraint functions, is combined with the objective function. The penalty function method is in this category [6–9]. In the penalty function method, an extended objective function is defined by adding the constraint violation to the objective function as a penalty. The optimization of the objective function and the constraint violation is realized by the optimization of the extended objective function. The main difficulty of the penalty function method is the selection of an appropriate value for the penalty coefficient that adjusts the strength of the penalty. If the penalty coefficient is large, feasible solutions can be obtained, but the optimization of the objective function will be insufficient. On the contrary, if the penalty coefficient is small, high quality (but infeasible) solutions can be obtained because it is difficult to decrease the constraint violation. In order to solve the difficulty, some methods, where the penalty coefficient is adaptively controlled, are proposed [5, 10, 11].
3. The constraint violation and the objective function are used separately. In this category, both the constraint violation and the objective function are optimized by a lexicographic order in which the constraint violation precedes the objective function. The rule of comparison where the constraint violation precedes the objective function is called *feasibility rule* and is a standard constraint-handling technique. The comparison rule according to the feasibility rule is as follows: If two solutions are feasible, the one with the smaller objective value is better. If one solution is feasible and the other is infeasible, the feasible one is better. If both solutions are infeasible, the one with the smaller constraint violation is better. Also, the rule can be expressed as "If one solution has smaller constraint violation than the other, the solution is better. Otherwise the one with smaller

objective value is better". Note that the feasibility rule is equivalent to the penalty function method with the penalty coefficient $\rho = \infty$. Deb [12] proposed a method that adopts the extended objective function, which realizes the lexicographic ordering. Takahama and Sakai proposed the α constrained method [13] and the ε constrained method [14] that adopt a lexicographic ordering with relaxation of the constraints. Runarsson and Yao [15] proposed the stochastic ranking method that adopts the stochastic lexicographic order, which ignores the constraint violation with some probability. Mezura-Montes and Coello [16] proposed a comparison mechanism that is equivalent to the lexicographic ordering. Venkatraman and Yen [17] proposed a two-step optimization method, which first optimizes the constraint violation and then the objective function. These methods were successfully applied to various problems.

4. The constraints and the objective function are optimized by multiobjective optimization methods. In this category, the constrained optimization problems are solved as the multiobjective optimization problems in which the objective function and the constraint functions are objectives to be optimized [18–23]. But in many cases, solving multiobjective optimization problems is a more difficult and expensive task than solving single objective optimization problems.
5. Hybridization methods. In this category, constrained problems are solved by combining some of the above mentioned methods. Mallipeddi and Suganthan [24] proposed a hybridization of the methods in the categories 2, 3 and 4.

In this study, the category 2 is paid attention to. The EPC method is utilized for constrained optimization by PSO.

2.3 Penalty Function Methods

In the constrained optimization, it is necessary to optimize the objective function and the constraint violation simultaneously. In the penalty function method, the constrained optimization problem is converted to the following unconstrained optimization problem by adding the constraint violation $\phi(\boldsymbol{x})$ weighted by the penalty coefficient to the objective function $f(\boldsymbol{x})$ as a penalty.

$$F(\boldsymbol{x}) = f(\boldsymbol{x}) + \rho\phi(\boldsymbol{x}) \tag{2}$$

where $F(\cdot)$ is the extended objective function and ρ is the penalty coefficient ($\rho > 0$). By increasing the penalty coefficient towards infinity, the constraint violation converges to 0, and a feasible solution can be obtained.

The constraint violation $\phi(\boldsymbol{x})$ satisfies the following:

$$\begin{cases} \phi(\boldsymbol{x}) = 0, \text{ if } \boldsymbol{x} \in \mathcal{F} \\ \phi(\boldsymbol{x}) > 0, \text{ if } \boldsymbol{x} \notin \mathcal{F} \end{cases} \tag{3}$$

Some types of constraint violations, which are adopted as a penalty in the penalty function method, can be defined as follows:

$$\phi(\boldsymbol{x}) = \max\{\max_j\{0, g_j(\boldsymbol{x})\}, \max_j |h_j(\boldsymbol{x})|\} \tag{4}$$

$$\phi(\boldsymbol{x}) = \sum_j (\max\{0, g_j(\boldsymbol{x})\})^p + \sum_j |h_j(\boldsymbol{x})|^p \tag{5}$$

where p is a positive number. In this study, Eq. (5) is used with $p = 1$.

There are three types of the penalty approaches: static penalty, dynamic penalty and adaptive penalty approaches. The value of the penalty coefficient is fixed in the static penalty approach, and it is changed dynamically according to the number of generations or iterations in the dynamic penalty approach. One needs to select a proper fixed value or proper changing schedule by trial and error, because the proper value and the proper schedule depend on the problem to be solved. In the adaptive penalty approach, the coefficient is changed based on information obtained from the population of solutions. The problem with the adaptive penalty approach is that some parameters for adaptive control of the penalty coefficient are introduced and proper parameter values still depend on the problems to be solved. Some approaches without the parameters are proposed. In [25], the normalization of the objective function and the constraint functions is proposed as follows:

$$\tilde{f}(\boldsymbol{x}) = \frac{f(\boldsymbol{x}) - f_{\min}}{f_{\max} - f_{\min}} \tag{6}$$

where $f_{\min}$ and $f_{\max}$ are the minimum value and the maximum value of f in the population, respectively.

$$\tilde{v}(\boldsymbol{x}) = \frac{1}{m}\sum_{j=1}^{m} \frac{v_j(\boldsymbol{x})}{v_{j,\max}} \tag{7}$$

$$v_j(\boldsymbol{x}) = \begin{cases} \max\{0, g_j(\boldsymbol{x})\}, & j = 1, \cdots, q \\ |h_j(\boldsymbol{x})|, & j = q+1, \cdots, m \end{cases} \tag{8}$$

where $v_{j,\max}$ is the maximum value of v_j in the population.

$$F(\boldsymbol{x}) = \begin{cases} \tilde{f}(\boldsymbol{x}), & \text{if } \boldsymbol{x} \in \mathcal{F} \\ \tilde{v}(\boldsymbol{x}), & \text{if } R_f = 0 \\ \sqrt{\tilde{f}(\boldsymbol{x})^2 + \tilde{v}(\boldsymbol{x})^2} + A(\boldsymbol{x}), & \text{otherwise} \end{cases} \tag{9}$$

where R_f is the rate of feasible solutions in the population and $A(\boldsymbol{x}) = (1 - R_f)\tilde{v}(\boldsymbol{x}) + R_f\tilde{f}(\boldsymbol{x})$.

In [26], balancing the objective value and the constraint violations is proposed as follows:

$$F(\boldsymbol{x}) = \begin{cases} f(\boldsymbol{x}), & \text{if } \boldsymbol{x} \in \mathcal{F} \\ \tilde{f}(\boldsymbol{x}) + \sum_{j=1}^{m} k_j v_j(\boldsymbol{x}), & \text{otherwise} \end{cases} \tag{10}$$

$$\tilde{f}(\boldsymbol{x}) = \begin{cases} f(\boldsymbol{x}), & \text{if } f(\boldsymbol{x}) > \bar{f} \\ \bar{f}, & \text{otherwise} \end{cases} \tag{11}$$

where $\bar{f}$ is the average of f in the population.

$$k_j = |\bar{f}| \frac{\bar{v}_j}{\sum_{l=1}^{m} \bar{v}_l^2} \tag{12}$$

where $\bar{v}_j$ is the average of v_j in the population.

The EPC method will be explained in Section 4.

3 Particle Swarm Optimization

An animal such as an ant, a fish, and a bird has limited memory and ability to perform simple actions. In contrast, a group of animals such as an ant swarm, a fish school, and a bird flock can take complex or intelligent actions such as avoiding predators and seeking foods efficiently. Swarm intelligence is defined as the collective actions of agents that act autonomously and communicate each other. PSO [27,28] is an optimization method based on swarm intelligence which is inspired by the movement of a bird flock. PSO imitates the movement to solve optimization problems and is considered as a population-based stochastic search method or POA.

Searching procedures by PSO can be described as follows: A group of agents, or a population minimizes the objective function f. At any time t, each agent i knows its current position $\boldsymbol{x}_i^t$ and velocity $\boldsymbol{v}_i^t$. It also remembers its personal best visited position found so far $\boldsymbol{x}_i^*$ and the objective value $pbest_i$.

$$\boldsymbol{x}_i^* = \arg \min_{\tau=0,1,\cdots,t} f(\boldsymbol{x}_i^\tau),\ pbest_i = f(\boldsymbol{x}_i^*) \tag{13}$$

Two models, gbest model and lbest model have been proposed [29,30]. In the gbest model, every agent knows the best visited position $\boldsymbol{x}_G^*$ in all agents and its objective value $gbest$.

$$\boldsymbol{x}_G^* = \arg \min_i f(\boldsymbol{x}_i^*),\ gbest = f(\boldsymbol{x}_G^*) \tag{14}$$

In the lbest model, each agent knows the best visited position $\boldsymbol{x}_l^*$ in the neighbors and its objective value $lbest_i$, where the neighbors are defined by a topology such as ring, mesh, star and tree topology.

$$\boldsymbol{x}_l^* = \arg \min_{k \in N_i} f(\boldsymbol{x}_k^*),\ lbest_i = f(\boldsymbol{x}_l^*) \tag{15}$$

where N_i is the set of neighbor agents to i. The velocity of the agent i at time $t+1$ is defined as follows:

$$\begin{aligned} v_{ij}^{t+1} = w v_{ij}^t &+ c_1\, rand_{1ij}\, (x_{ij}^* - x_{ij}^t) \\ &+ c_2\, rand_{2ij}\, (x_{lj}^* - x_{ij}^t) \end{aligned} \tag{16}$$

where $l = G$ in the gbest model, w is an inertia weight and $rand_{kij}$ is a uniform random number in $[0,1]$ which is generated in each dimension. c_1 is a cognitive parameter and c_2 is a social parameter which represent the weight of the movement to the personal best and the group/neighbors best, respectively. Usually, the maximum velocity $V_j^{\max}$ is specified to avoid too large velocity and $|v_{ij}| \le V_j^{\max}$ is satisfied.

The position of the agent i at time $t+1$ is given as follows:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \boldsymbol{v}_i^{t+1} \tag{17}$$

In linearly decreasing inertia weight (LDIW) method [31], the inertia weight w is linearly decreasing with the number of iterations as follows:

$$w = w_{\max} - (w_{\max} - w_{\min}) \frac{t}{T_{\max}} \tag{18}$$

where $w_{\max}$ and $w_{\min}$ are the maximum weight and the minimum weight for w, respectively. $T_{\max}$ is the maximum number of iterations. Recommended values are $w_{\max}$=0.9, $w_{\min}$=0.4, c_1=c_2=2 and $V_j^{\max}$=u_j.

In constriction model [32], Eq.(16) is modified to guarantee the convergence of agents to a local optimum as follows:

$$\begin{aligned} v_{ij}^{t+1} = \chi[v_{ij}^t &+ c_1\, rand_{1ij}\, (x_{ij}^* - x_{ij}^t) \\ &+ c_2\, rand_{2ij}\, (x_{Gj}^* - x_{ij}^t)] \end{aligned} \tag{19}$$

$$\chi = \frac{2}{\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}} \tag{20}$$

$$\varphi = c_1 + c_2,\ \varphi > 4 \tag{21}$$

In [33], it is shown that the constriction model with χ=0.729 and c_1=c_2=2.05 is equivalent to Eq.(16) with w=0.729 and c_1=c_2=0.729×2.05 =1.49445 and was performed well with using $V_j^{\max}$=u_j.

In this study, the constriction model is used for solving optimization problems.

4 Proposed method

4.1 Equivalent Penalty Coefficient (EPC) Method

Let assume the POAs where a new solution $\boldsymbol{x}_i'$ is compared with the old solution $\boldsymbol{x}_i$ and the old solution is replaced with the new solution only if the new solution is better than the old solution. When both of the objective value and the constraint violation of $\boldsymbol{x}_i'$ are better than those of $\boldsymbol{x}_i$, the value of the extended objective function of $\boldsymbol{x}_i'$ is always better than that of $\boldsymbol{x}_i$, or $F(\boldsymbol{x}') < F(\boldsymbol{x})$ for any ρ, and vice versa. Also, if the objective values and the constraint violations are the same, $F(\boldsymbol{x}) = F(\boldsymbol{x}')$ holds for any ρ. Therefore, if the following conditions are satisfied, there is no need to determine the penalty coefficient.

$$f(\boldsymbol{x}') \le f(\boldsymbol{x}) \text{ and } \phi(\boldsymbol{x}') \le \phi(\boldsymbol{x}) \quad (22)$$
$$\text{or } f(\boldsymbol{x}) \le f(\boldsymbol{x}') \text{ and } \phi(\boldsymbol{x}) \le \phi(\boldsymbol{x}')$$

Otherwise, there exist a solution with a better objective value and a worse violation value and a solution with a worse objective value and a better violation value. The equivalent penalty coefficient value (EPC) ρ_i is defined as the value that makes the extended objective values of $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$ the same:

$$F(\boldsymbol{x}_i) = F(\boldsymbol{x}'_i) \quad (23)$$
$$f(\boldsymbol{x}_i) + \rho_i\phi(\boldsymbol{x}_i) = f(\boldsymbol{x}'_i) + \rho_i\phi(\boldsymbol{x}'_i) \quad (24)$$
$$\rho_i = -\frac{f(\boldsymbol{x}_i) - f(\boldsymbol{x}'_i)}{\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}'_i)} \quad (25)$$

Let assume that $f(\boldsymbol{x}_i) < f(\boldsymbol{x}'_i)$ and $\phi(\boldsymbol{x}'_i) < \phi(\boldsymbol{x}_i)$. When the penalty coefficient is $\rho_i + \triangle$, the following equation is satisfied:

$$F(\boldsymbol{x}_i) - F(\boldsymbol{x}'_i) \quad (26)$$
$$= (f(\boldsymbol{x}_i) + (\rho_i + \triangle)\phi(\boldsymbol{x}_i)) - (f(\boldsymbol{x}'_i) + (\rho_i + \triangle)\phi(\boldsymbol{x}'_i))$$
$$= \triangle(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}'_i))$$

If the penalty coefficient is larger than ρ_i, or $\triangle$ is positive, $F(\boldsymbol{x}'_i) < F(\boldsymbol{x}_i)$ is satisfied and $\boldsymbol{x}'_i$ is the better solution. On the contrary, if the penalty coefficient is smaller than ρ_i, or $\triangle$ is negative, $F(\boldsymbol{x}_i) < F(\boldsymbol{x}'_i)$ is satisfied and $\boldsymbol{x}_i$ is the better solution.

Let consider the list of ρ_i sorted in ascending order, $H = \{\rho_k \,|\, \rho_k < \rho_{k+1}, k = 1, 2, \cdots\}$. Because the solutions that satisfy Eq.(22) are excluded, $|H| \le N$, where $|H|$ is the number of elements in H and N is the number of solutions, or agents in PSO. In order to control the penalty coefficient simply and adaptively, an algorithm parameter R_{cp} ($R_{cp} \ge 0$), which is a constraint priority rate and specifies the pressure to move the population into the feasible region, is introduced. The coefficient ρ is decided as the $R_{cp}|H|$-th element in H using linear interpolation. Note that ρ is commonly used to calculate the extended objective function values $F(\cdot)$ for all solutions according to Eq.(2) and the values are used as the objective values $f(\cdot)$ in Eqs.(13)–(15). For example, in case of $R_{cp} = 0.9$ and $|H| = 10$, $R_{cp}|H| = 9$ and $\rho = \rho_9$. In case of $R_{cp} = 0.15$ and $|H| = 10$, $R_{cp}|H| = 1.5$, but $\rho_{1.5}$ is not defined and can be obtained from ρ_1 and ρ_2 using the linear interpolation as follows:

$$\rho = \begin{cases} R_{cp}|H|\rho_1, & \text{if } R_{cp}|H| < 1 \\ \rho_{|H|}R_{cp}, & \text{if } R_{cp} > 1 \\ \rho_{\lfloor R_{cp}|H| \rfloor} + \triangle R_{cp}\triangle\rho, & \text{otherwise} \end{cases} \quad (27)$$

$$\triangle R_{cp} = R_{cp}|H| - \lfloor R_{cp}|H| \rfloor \quad (28)$$
$$\triangle\rho = \rho_{\lceil R_{cp}|H| \rceil} - \rho_{\lfloor R_{cp}|H| \rfloor} \quad (29)$$

where $\lfloor\cdot\rfloor$ is rounding down to the nearest integer and $\lceil\cdot\rceil$ is rounding up to the nearest integer. The first equation in Eq.(27) is the linear interpolation between 0 and ρ_1. For example, in case of $R_{cp}|H| = 0.5$, $\rho = 0.5\rho_1$. When $R_{cp} = 0$, only the objective value will be optimized because $\rho = 0$. In the second equation when $R_{cp} > 1$, the constraint violation has always higher priority than the objective value because $\rho > \rho_{|H|}$. Setting $R_{cp} > 1$ has similar effect of $\rho = \infty$ in the ordinary penalty function method. It is thought that a feasible solution can be found by changing R_{cp} to over 1 theoretically as $\rho \to \infty$. In the third equation, $\triangle R_{cp}$ is the decimal part of $R_{cp}|H|$. $\triangle\rho$ is the difference between ρ_{k+1} and ρ_k where k is the integer part of $R_{cp}|H|$ if the decimal part is not zero. If the decimal part is zero, $\triangle\rho$=0. The value of ρ is obtained by the linear interpolation of ρ_k and ρ_{k+1}. For example, in the case of $R_{cp}|H| = 1.5$, $\triangle R_{cp}$=0.5, $\triangle\rho$=$\rho_2 - \rho_1$ and $\rho = \rho_1 + 0.5(\rho_2 - \rho_1)$.

In order to avoid a sudden change in penalty coefficient value, the penalty coefficient value of generation t, $\rho(t)$, is determined using the following exponential moving average:

$$\rho(t) = \begin{cases} (1-\lambda)\rho(t-1) + \lambda\rho, & \text{if } t > 1 \\ \rho, & \text{if } t = 1 \end{cases} \quad (30)$$

At time t, ρ is calculated by Eq.(27) and then $\rho(t)$ is obtained and used as the penalty coefficient. The recommended value of λ is [0.5,0.8] based on some preliminary experiments.

4.2 Adaptive Control of the Constraint Priority Rate

The EPC method can adaptively control the penalty coefficient using the fixed value of $R_{cp} = 0.9$ in problems without equality constraints. However, it is difficult to solve some problems with equality constraints. Therefore, an adaptive control of R_{cp} is introduced as follows:

$$R_{cp} = \begin{cases} R_{cp}^{\min}, & \text{if } R_f = 0 \\ R_{cp}^{\min} + (1 - R_{cp}^{\min})(1 - R_f), & \text{otherwise} \end{cases} \quad (31)$$

$$R_f = \frac{|\{\boldsymbol{x}_i^* \mid \phi(\boldsymbol{x}_i^*) = 0\}|}{N} \quad (32)$$

where $R_{cp}^{\min}(0 \le R_{cp}^{\min} < 1)$ is a parameter which specifies the minimum value of R_{cp} and R_f is the feasible rate of $\boldsymbol{x}_i^*$ (best visited position found so far) in all best visited positions in PSO. If $R_f = 0$, that is, the population is far from the feasible region, $R_{cp} = R_{cp}^{\min}$ so that the population gradually approaches the feasible region with balancing the optimization of the objective function and the constraint. Otherwise, the value of R_{cp} is in $[R_{cp}^{\min}, 1]$ according to R_f. If $R_f > 0$ and R_f is very small, that is, a few feasible solutions are found, R_{cp} is set to nearly 1 in order to push the population towards the feasible region and R_f will be increased. If R_f is 1, that is, the population is inside of the feasible region, R_{cp} is set to $R_{cp}^{\min}$ in order to search solutions in wide area including the boundary of the feasible region and R_f will be decreased. The recommended value of $R_{cp}^{\min}$ is 0.9 which is determined based on some preliminary experiments.

4.3 Mutation and Repair

In this study, a mutation operation similar to rand/1 mutation of DE is utilized as follows:

$$\boldsymbol{x}_i' = \boldsymbol{x}_{r1}^* + F(\boldsymbol{x}_{r2}^* - \boldsymbol{x}_{r3}^*) \tag{33}$$

where r_1, r_2 and r_3 are random numbers in $\{1, 2, \cdots, N\}$ excluding i and are different from each other. N is the number of agents and F is a scaling factor. A new position is created using personal best positions. It is expected that the mutation will help to move in a narrow feasible region. The mutation is applied with probability $P_m = 0.25$ based on some preliminary experiments.

When a new position is out of the search space, the position is repaired to be inside of the search space. The repaired position is the middle of the violated bounds and the corresponding variables of the current position as follows [34]:

$$x_{ij}^{\text{repaired}} = \begin{cases} \frac{1}{2}(l_j + x_{ij}^t), & x_{ij}^{t+1} < l_j \\ \frac{1}{2}(u_j + x_{ij}^t), & x_{ij}^{t+1} > u_j \\ x_{ij}^{t+1}, & \textit{otherwise} \end{cases} \tag{34}$$

Also, the velocity is changed so that the bounds are not violated again as follows:

$$v_{ij}^{\text{repaired}} = \begin{cases} -\frac{1}{2}v_{ij}^{t+1}, & x_{ij}^{t+1} < l_j \text{ or } x_{ij}^{t+1} > u_j \\ v_{ij}^{t+1}, & \textit{otherwise} \end{cases} \tag{35}$$

4.4 The algorithm of the proposed method

The algorithm of the proposed method PSOEPC (PSO with EPC) is as follows:

0. The number of agents N, PSO parameters w, c_1, and c_2, PSO topology, the mutation parameters P_m and F, and EPC parameters p, λ, and $R_{cp}^{\min}$ are specified.
1. Initializing agents: Initial agent i with a position $\boldsymbol{x}_i$ and a velocity $\boldsymbol{v}_i$ is created for all $i \in \{1, 2, \cdots, N\}$. $\boldsymbol{x}_i$ is randomly generated in the search space $\mathcal{S}$ where each element x_{ij} is a uniform random number in $[l_j, u_j]$. $\boldsymbol{v}_i$ is initialized randomly where v_{ij} is a random number in $[-V_{\max_j}, V_{\max_j}]$ and $V_{\max_j} = \frac{1}{2}(u_j - l_j)$, which is half the range of j-th variable, in this study.
2. Evaluating agents: All agents i are evaluated and $f(\boldsymbol{x}_i)$ and $\phi(\boldsymbol{x}_i)$ are obtained. The personal best position is set to the initial position, namely $\boldsymbol{x}_i^*{=}\boldsymbol{x}_i$.
3. Obtaining the feasible rate: The feasible rate of personal best positions is obtained according to Eq.(32).
4. Termination condition: If the number of function evaluations exceeds the maximum number of function evaluations $FE_{\max}$, the algorithm is terminated.
5. Updating agents: Mainly, the agents are updated by the movement. The new velocity of each agent i are obtained according to Eq.(16). The each element of the new velocity is truncated in $[-V_{max_j}, V_{max_j}]$. The new position is obtained according to Eq.(17). Otherwise, with probability P_m, the agents are updated by the mutation according to Eq. (33). If the position is out of the search space, the position and the velocity is repaired according to Eqs. (34) and (35).
6. EPC method: ρ_i, R_{cp}, ρ and $\rho(t)$ are determined according to Eqs. (25), (31), (27) using the feasible rate and Eq.(30).
7. Updating the personal best positions: The values of the extended objective function for the new positions $\boldsymbol{x}_i'$ and the best visited positions $\boldsymbol{x}_i^*$ are obtained according to Eq. (2) using $\rho(t)$ instead of ρ. If the extended objective value of the new position $F(\boldsymbol{x}_i')$ is better than that of the personal best position $F(\boldsymbol{x}_i^*)$, the personal best position is replaced with the new position. If the extended objective value of the new position is better than that of the best position in all agents, the best position is set to the new position.
8. Go back to Step2.

5 Solving Nonlinear Optimization Problems

In this paper, thirteen benchmark problems that are mentioned in some studies [3, 15, 16] are optimized by the proposed method PSOEPC.

5.1 Test problems and the experimental conditions

In the thirteen benchmark problems, problems `g03`, `g05`, `g11` and `g13` contain equality constraints. In problems with equality constraints, the equality constraints are relaxed and converted to inequality constraints according to Eq. (36), which is adopted in many methods:

$$|h_j(\boldsymbol{x})| \le 10^{-4} \tag{36}$$

Problem `g12` has disjointed feasible regions. Table 1 shows the outline of the thirteen problems [16, 35]. The table contains the number of variables D, the form of the objective function, the number of linear inequality constraints (LI), nonlinear inequality constraints (NI), linear equality constraints (LE), nonlinear equality constraints (NE) and the number of constraints active at the optimal solution.

Table 1 Summary of test problems

f	D	Form of f	LI	NI	LE	NE	active
g01	13	quadratic	9	0	0	0	6
g02	20	nonlinear	1	1	0	0	1
g03	10	polynomial	0	0	0	1	1
g04	5	quadratic	0	6	0	0	2
g05	4	cubic	2	0	0	3	3
g06	2	cubic	0	2	0	0	2
g07	10	quadratic	3	5	0	0	6
g08	2	nonlinear	0	2	0	0	0
g09	7	polynomial	0	4	0	0	2
g10	8	linear	3	3	0	0	6
g11	2	quadratic	0	0	0	1	1
g12	3	quadratic	0	9^3	0	0	0
g13	5	nonlinear	0	0	1	2	3

Settings for PSO are as follows: The constriction model with w=0.729 and c_1=c_2=1.49445 is adopted. The number of agents $N = 50$ and the maximum number of function evaluations $FE_{\max} = 200,000$. Settings for the mutation are as follows: The mutation rate P_m=0.25 and scaling factor F is randomly generated in [0.4,0.9]. Settings for EPC method are as follows: Every constraint violation is defined as a simple sum of constraints, or $p = 1$ in Eq. (5). The value of λ in Eq.(30) is 0.8. The R_{cp} is controlled using Eq. (31) with $R_{cp}^{\min} = 0.9$.

In this paper, 30 independent runs are performed.

5.2 Experimental results

In the experiments, several methods are examined: PSO with the feasibility rule (PSO), PSOEPC without the mutation (PSOEPC), PSO with the mutation (PSOm) and PSOEPC with the mutation (PSOEPCm). Two representative topologies, or fully-connected topology (gbest) and the ring topology (ring), are also examined. In the ring topology, the neighborhood size is 3, that is, i-th agent is connected to $(i-1)$-th and $(i+1)$-th agent.

Table 2 shows results of the best, median, mean, worst values and the standard deviation for the above methods. The success rate, which is the ratio of finding a feasible solution in 30 runs, is shown in parentheses.

All methods found optimal solutions in all 30 runs for g04 and g12. As for g01, PSO, PSOm, PSOEPC and PSOEPCm with the ring topology found the optimal solutions in all 30 runs. As for g02, PSOEPCm (ring) attained the best results. As for g03, g07, g10 and g13, PSOEPCm (ring) attained the best results. As for g05, PSOEPCm (gbest) and PSOEPCm (ring) attained the best results and found the optimal solutions in all runs. As for g06 and g11, PSOEPC (ring), PSOm with both topologies and PSOEPCm with both topologies found the optimal solutions in all runs. As for g08, all methods except for PSOEPC (gbest) found the optimal solutions in all runs. As for g09, PSOm with both topologies and PSOEPCm with both topologies found the optimal solutions in all runs. Therefore, it is thought that PSOEPCm (ring) is the best method which find near optimal solutions in all runs for all problems except for g02.

Wilcoxon signed rank test is performed and the result for each function is shown in Table 3. Symbols '+', '−' and '=' are shown when a method is significantly better than PSOEPCm (ring), is significantly worse than PSOEPCm (ring), and is not significantly different from PSOEPCm (ring), respectively. Symbols '++' and '−−' are shown when the significance level is 1% and '+' and '−' are shown when the significance level is 5%.

It is thought that EPC method is effective because PSOEPCm (ring) is significantly better than PSOm (ring) in 5 functions. It is thought that the mutation is effective because PSOEPCm (ring) is significantly better than PSOEPC (ring) in 9 functions. It is thought that the ring topology is effective because PSOEPCm (ring) is significantly better than PSOEPCm (gbest) in 5 functions. Thus, it is thought that the EPC method, the mutation and the ring topology is effective to constrained optimization.

5.3 Comparison with Other Methods

In order to show the performance of PSOEPC, PSOEPCm (ring) is compared with other methods including PSOm (ring) with static penalty of ρ=100 and 10,000, PSOm (ring) with the adaptive penalty method in [25] and DEEPC [5]. Also, PSOEPCm (ring) with Pm=0.25 is compared with PSOEPCm (ring) with Pm=0.5 and 0.75 to investigate the effect of the mutation.

Table 4 shows the results of the mean value and the standard deviation for the above methods. The success rate is shown in parentheses. As for the static penalty of ρ=100, it was difficult to find feasible solutions in g03 and g10. Also, near optimal solutions cannot be found in g04, g06 and g13. As for the static penalty of ρ=10,000, it was difficult to find feasible solutions in g10. Also, near optimal solutions cannot be found

Table 2 Comparison of statistical results among PSO with the gbest and ring topologies using the feasibility rule, PSOEPC (PSOEPC), PSO with the mutation (PSOm), and PSOEPC with the mutation.

	Statistics	PSO (gbest)	PSO (ring)	PSOEPC (gbest)	PSOEPC (ring)	PSOm (gbest)	PSOm (ring)	PSOEPCm (gbest)	PSOEPCm (ring)
g01 -15.000	best	-15.0000000	-15.0000000	-15.0000000	-15.0000000	-15.0000000	-15.0000000	-15.0000000	-15.0000000
	median	-15.0000000	-15.0000000	-15.0000000	-15.0000000	-15.0000000	-15.0000000	-15.0000000	-15.0000000
	mean	-14.1640625	**-15.0000000**	-14.3656250	**-15.0000000**	-14.4604167	**-15.0000000**	-14.2421875	**-15.0000000**
	worst	-12.4531250	-15.0000000	-12.6562500	-15.0000000	-12.4531250	-15.0000000	-12.4531250	-15.0000000
	σ	1.07e+00 (100)	**0.00e+00 (100)**	8.74e-01 (100)	**0.00e+00 (100)**	9.87e-01 (100)	**0.00e+00 (100)**	1.08e+00 (100)	**0.00e+00 (100)**
g02 -0.803619	best	-0.8036162	-0.8034665	-0.7631314	-0.8034396	-0.8036190	-0.8036189	-0.8036190	-0.8036189
	median	-0.6573037	-0.7951584	-0.6599638	-0.7933039	-0.7700933	-0.8036172	-0.7744902	-0.8036178
	mean	-0.6411993	-0.7889189	-0.6516125	-0.7833147	-0.7552821	-0.7985702	-0.7560894	**-0.8017130**
	worst	-0.3954511	-0.7369670	-0.3954511	-0.7202204	-0.6153742	-0.7808419	-0.5576084	-0.7926078
	σ	9.33e-02 (100)	1.81e-02 (100)	8.80e-02 (100)	2.44e-02 (100)	5.11e-02 (100)	6.43e-03 (100)	5.54e-02 (100)	**3.84e-03 (100)**
g03 -1.0005	best	-1.0004948	-0.9997957	-1.0004935	-1.0004480	-1.0005001	-1.0005001	-1.0005000	-1.0005001
	median	-1.0004870	-0.9988941	-1.0004643	-0.9945826	-1.0004998	-1.0004979	-1.0004937	-1.0004999
	mean	-1.0004847	-0.9986018	-1.0004430	-0.9398613	-1.0004958	-1.0003296	-0.9803877	**-1.0004987**
	worst	-1.0004598	-0.9967287	-1.0002049	-0.3078240	-1.0004600	-0.9978407	-0.4066266	-1.0004834
	σ	9.08e-06 (100)	8.59e-04 (100)	6.26e-05 (100)	1.36e-01 (100)	1.00e-05 (100)	5.85e-04 (100)	1.07e-01 (100)	**3.26e-06 (100)**
g04 -30665.5387	best	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718
	median	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718
	mean	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718
	worst	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718	-30665.5386718
	σ	**0.00e+00 (100)**	8.17e-10 (100)	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**
g05 5126.4967	best	5126.4996291	5126.5073298	5126.5240655	5126.5056529	5126.4967140	5126.4967140	5126.4967140	5126.4967140
	median	5151.8051173	5151.7224741	5133.1918329	5128.0081289	5126.4967140	5126.4967140	5126.4967140	5126.4967140
	mean	5179.6315473	5194.1348257	5136.9595236	5131.3478866	5126.8461961	5126.6143143	**5126.4967140**	**5126.4967140**
	worst	5359.5313214	5470.5300974	5186.2574538	5162.1289704	5132.9810511	5129.6092865	5126.4967140	5126.4967140
	σ	6.44e+01 (93)	8.52e+01 (87)	1.30e+01 (100)	8.00e+00 (63)	1.24e+00 (100)	5.61e-01 (100)	6.01e-09 (100)	**2.83e-11 (100)**
g06 -6961.8139	best	-6961.8138756	-6961.8138756	-6961.4512932	-6961.8138756	-6961.8138756	-6961.8138756	-6961.8138756	-6961.8138756
	median	-6961.8138756	-6961.8138756	-6959.9112774	-6961.8138756	-6961.8138756	-6961.8138756	-6961.8138756	-6961.8138756
	mean	-6961.8138756	-6961.8138756	-6959.4269765	**-6961.8138756**	**-6961.8138756**	**-6961.8138756**	**-6961.8138756**	**-6961.8138756**
	worst	-6961.8138755	-6961.8138753	-6954.5538565	-6961.8138756	-6961.8138756	-6961.8138756	-6961.8138756	-6961.8138756
	σ	1.83e-08 (100)	5.52e-08 (100)	1.56e+00 (100)	4.24e-10 (100)	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**
g07 24.3062	best	24.3532828	24.4324760	24.3597940	24.3739501	24.3062093	24.3062092	24.3062091	24.3062091
	median	24.9934956	24.6358790	24.9871796	24.6135761	24.3063520	24.3062606	24.3062261	24.3062149
	mean	25.0677962	24.6823089	25.2239374	24.6300412	24.3069002	24.3063747	24.3062972	**24.3062433**
	worst	26.5638488	25.2118705	27.5565406	24.9675733	24.3117616	24.3070163	24.3067668	24.3065045
	σ	5.17e-01 (100)	2.04e-01 (100)	7.62e-01 (100)	1.57e-01 (100)	1.35e-03 (100)	2.10e-04 (100)	1.51e-04 (100)	**6.57e-05 (100)**
g08 -0.095825	best	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250
	median	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250
	mean	**-0.0958250**	**-0.0958250**	-0.0958249	**-0.0958250**	**-0.0958250**	**-0.0958250**	**-0.0958250**	**-0.0958250**
	worst	-0.0958250	-0.0958250	-0.0958243	-0.0958250	-0.0958250	-0.0958250	-0.0958250	-0.0958250
	σ	**0.00e+00 (100)**	**0.00e+00 (100)**	2.18e-07 (100)	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**
g09 680.630057	best	680.6355188	680.6341766	680.6327793	680.6423843	680.6300574	680.6300574	680.6300574	680.6300574
	median	680.6557118	680.6504461	680.6475737	680.6575376	680.6300574	680.6300574	680.6300574	680.6300574
	mean	680.6599146	680.6533174	680.6511996	680.6661846	**680.6300574**	**680.6300574**	**680.6300574**	**680.6300574**
	worst	680.7163843	680.7048074	680.7117605	680.7170877	680.6300574	680.6300574	680.6300574	680.6300574
	σ	1.73e-02 (100)	1.48e-02 (100)	1.82e-02 (100)	2.05e-02 (100)	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**
g10 7049.248	best	7083.8886187	7073.9011643	7051.9120139	7079.6718113	7049.2480207	7049.2480216	7049.2480209	7049.2480207
	median	7269.2943399	7160.5622036	7479.6219922	7183.2302366	7049.2480432	7049.2480373	7049.2480230	7049.2480224
	mean	7310.4078886	7186.8170074	7814.5071983	7218.4466184	7049.2481105	7049.2480771	7049.2480330	**7049.2480232**
	worst	7625.6551359	7593.2805862	13539.9905406	7473.9271745	7049.2484709	7049.2483591	7049.2481216	7049.2480327
	σ	1.30e+02 (100)	1.12e+02 (100)	1.14e+03 (100)	1.09e+02 (100)	1.35e-04 (100)	8.82e-05 (100)	2.48e-05 (100)	**2.90e-06 (100)**
g11 0.749900	best	0.7499000	0.7499000	0.7499000	0.7499000	0.7499000	0.7499000	0.7499000	0.7499000
	median	0.7499000	0.7499001	0.7499006	0.7499000	0.7499000	0.7499000	0.7499000	0.7499000
	mean	0.7499001	0.7499003	0.7499019	**0.7499000**	**0.7499000**	**0.7499000**	**0.7499000**	**0.7499000**
	worst	0.7499010	0.7499032	0.7499152	0.7499000	0.7499000	0.7499000	0.7499000	0.7499000
	σ	1.81e-07 (100)	7.16e-07 (100)	3.63e-06 (100)	4.12e-09 (100)	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**	**0.00e+00 (100)**
g12 -1.000	best	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000
	median	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000
	mean	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000
	worst	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000
	σ	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g13 0.0539415	best	0.0544964	0.0557530	0.0539452	0.0543969	0.0539415	0.0539415	0.0539415	0.0539415
	median	0.4405095	0.4604644	0.0548208	0.1026147	0.4388026	0.4388026	0.0539415	0.0539415
	mean	0.3511268	0.4187795	0.2212339	0.1665166	0.3365596	0.3538300	0.1052563	**0.0539415**
	worst	1.0000000	0.9956896	0.4408550	0.5352735	0.7359228	0.5784183	0.4388026	0.0539415
	σ	2.22e-01 (100)	1.96e-01 (100)	1.91e-01 (93)	1.31e-01 (90)	1.80e-01 (100)	1.57e-01 (100)	1.31e-01 (100)	**2.79e-15 (100)**

in g03 and g13. As for the adaptive penalty method, feasible solutions can be found in all problems in all runs. But near optimal solutions cannot be found in g01, g03–g07, g10, and g13. Therefore, it is thought that PSOEPCm (ring) is better than the static penalty methods and the adaptive penalty method.

As for PSOEPCm (ring) with Pm=0.25, 0.5 and 0.75, same near optimal solutions were found in 8 problems of g01, g04–g06, g08, g09, g11 and g12. PSOEPCm (ring) with Pm=0.25 attained the best results in g02 and g13. PSOEPCm (ring) with Pm=0.5 attained the best result in g10. PSOEPCm (ring) with Pm=0.75 attained the best results in g03, g07 and g10. The difference of mean values are very small except for g02 and g13. Therefore, from the viewpoint of mean values in Table 4 it is thought that PSOEPCm (ring) with Pm=0.25 is better than PSOEPCm (ring) with Pm=0.5 and 0.75, and the difference is not large.

As for DEEPC, DEEPC found better mean values than PSOEPCm (ring) with Pm=0.25 in g02, g03, g07 and g10, but the difference is very small. it is thought that from the viewpoint of mean values in Table 4 DEEPC is a little better than PSOEPCm (ring) with Pm=0.25. PSOEPCm (ring) with Pm=0.25 is a very good PSO-based method.

Table 3 Results of Wilcoxon signed rank test against PSOEPCm with the ring topology.

	PSO gbest	PSO ring	PSOEPC gbest	PSOEPC ring	PSOm gbest	PSOm ring	PSOEPCm gbest
g01	$--$	$=$	$--$	$=$	$-$	$=$	$--$
g02	$--$	$--$	$--$	$--$	$--$	$--$	$--$
g03	$--$	$--$	$--$	$--$	$=$	$--$	$--$
g04	$=$	$--$	$=$	$=$	$=$	$=$	$=$
g05	$--$	$--$	$--$	$--$	$=$	$=$	$--$
g06	$--$	$--$	$--$	$--$	$=$	$=$	$=$
g07	$--$	$--$	$--$	$--$	$--$	$--$	$=$
g08	$=$	$=$	$--$	$=$	$=$	$=$	$=$
g09	$--$	$--$	$--$	$--$	$=$	$=$	$=$
g10	$--$	$--$	$--$	$--$	$--$	$--$	$=$
g11	$--$	$--$	$--$	$--$	$=$	$=$	$=$
g12	$=$	$=$	$=$	$=$	$=$	$=$	$=$
g13	$--$	$--$	$--$	$--$	$--$	$--$	$--$
$+$	0	0	0	0	0	0	0
$=$	3	3	2	4	8	8	8
$-$	10	10	11	9	5	5	5

Wilcoxon signed rank test for PSOEPCm against DEEPC is performed and the result for each function is shown in Table 5. Symbols '$+$', '$-$' and '$=$' are shown when a method is significantly better than DEEPC, is significantly worse than DEEPC, and is not significantly different from DEEPC, respectively. Symbols '$++$' and '$--$' are same in Table 3.

The results are slightly different from Table 4 because very small differences of 10^{-10} or less are taken into account. According to the number of '+' and '-' signs, PSOEPCm with Pm=0.75 is the best method followed by DEEPC, PSOEPCm with Pm=0.5 and PSOEPCm with Pm=0.25. Therefore, it is thought that PSOEPCm is equivalent to or better than DEEPC by selecting proper Pm.

PSOEPCm with Pm=0.75 attained significantly better results than DEEPC in 2 functions g03 which has a unimodal objective function and g10 which has a linear objective function, and attained significantly worse result in g02 which has a highly multimodal objective function. It is thought that PSOEPCm is more suitable to solve unimodal objective functions as PSO than highly multimodal objective functions.

5.4 Discussion

In order to investigate the scalability of PSOEPCm (ring) for the number of dimensions D, the following constrained optimization problem with an equality constraint is solved with changing D.

$$\text{minimize} \qquad f(\boldsymbol{x}) = \sum_{j=1}^{D} x_j^2 \tag{37}$$

$$\text{subject to} \qquad h_1(\boldsymbol{x}) = \sum_{j=1}^{D} x_j^2 - 1 = 0 \tag{38}$$

$$-100 \le x_j \le 100, j = 1, 2, \cdots, D \tag{39}$$

where the equality constraint is relaxed according to Eq.(36). The optimal value is 1. The problem is solved by PSOEPCm (ring) with the same settings as the above experiments such as N=50, Pm=0.25 and 30 runs except for $FE_{\max} = 10000D$. Table 6 shows the results of the mean value and the standard deviation in case of D=50, 100, 200, 300, 400 and 500.

PSOEPCm (ring) found the near optimal solutions stably in all runs and all D, where the best values are less then 1 due to the relaxation. It is thought that PSOEPCm (ring) is robust to D in this problem. Since the scalability for D depends on problems to be solved and also $FE_{\max}$, it is difficult to show the scalability in general. It is thought that enhanced PSOs for large scale problems [36] are suitable to solve problems with the large number of dimensions.

6 Conclusions

In the penalty function method, feasible solutions can be found by increasing the penalty coefficient towards infinity theoretically, although it is difficult to do so computationally. In the EPC method, the penalty coefficient is controlled adaptively using the equivalent penalty coefficient (EPC) values. The constraint priority rate is defined to select a proper EPC value from EPC values and the adaptive control of the constraint priority rate using the feasible rate of the population is also defined.

In this study, the EPC method is introduced to PSO, the mutation and repair operations for PSO are defined, and PSOEPC is proposed. The experiments for solving well known constrained problems were performed and it was shown that PSOEPC with the ring topology and the mutation could search for high quality solutions in all problems compare with standard methods. Also, it was shown that the EPC method attained better results than other constraint-handling methods including the static penalty and the adaptive penalty methods.

In the future, we will investigate the scalability of PSOEPC and apply PSOEPC to various real world problems that have large numbers of decision variables and constraints. Also, we will introduce the idea of EPC method into other POAs.

Acknowledgements This study is supported by JSPS KAKENHI Grant Numbers 17K00311 and 19K04916.

References

1. Michalewicz Z. (1995) A survey of constraint handling techniques in evolutionary computation methods. In:

Table 4 Comparison of results among PSOm (ring) with the static penalty methods, PSOm (ring) with the adaptive penalty method, PSOEPCm (ring) with Pm=0.25, 0.5 and 0.75, and DEEPC.

	Statistics	static penalty ρ=100	static penalty ρ=10,000	adaptive penalty	PSOEPCm Pm=0.25	PSOEPCm Pm=0.5	PSOEPCm Pm=0.75	DEEPC
g01	mean	**-15.0000000**	**-15.0000000**	-14.8218545	**-15.0000000**	**-15.0000000**	**-15.0000000**	**-15.0000000**
-15.000	σ	0.00e+00 (100)	0.00e+00 (100)	1.86e-02 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g02	mean	-0.7961152	-0.7964215	-0.7976481	-0.8017130	-0.7861999	-0.7543802	**-0.8022674**
-0.803619	σ	7.69e-03 (100)	7.80e-03 (100)	8.61e-03 (100)	3.84e-03 (100)	1.75e-02 (100)	3.79e-02 (100)	4.18e-03 (100)
g03	mean	-0.0216546	-0.3154008	-0.2839239	-1.0004987	-1.0004999	**-1.0005001**	**-1.0005001**
-1.0005	σ	3.73e-02 (3)	1.53e-01 (100)	1.23e-01 (100)	3.26e-06 (100)	4.62e-07 (100)	7.86e-11 (100)	2.29e-11 (100)
g04	mean	-29660.4568327	**-30665.5386718**	-30611.8723547	**-30665.5386718**	**-30665.5386718**	**-30665.5386718**	**-30665.5386718**
-30665.5387	σ	3.34e+02 (100)	0.00e+00 (100)	1.36e+01 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g05	mean	**5126.4967140**	**5126.4967140**	5187.1995051	**5126.4967140**	**5126.4967140**	**5126.4967140**	**5126.4967140**
5126.4967	σ	0.00e+00 (100)	1.75e-09 (100)	7.74e+01 (100)	2.83e-11 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g06	mean	-6044.8226251	**-6961.8138756**	-6959.1019138	**-6961.8138756**	**-6961.8138756**	**-6961.8138756**	**-6961.8138756**
-6961.8139	σ	6.43e+02 (100)	0.00e+00 (100)	1.54e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g07	mean	24.3063207	24.3065779	24.4674655	24.3062433	24.3062174	24.3062092	**24.3062091**
24.3062	σ	2.51e-04 (100)	4.56e-04 (100)	2.79e-02 (100)	6.57e-05 (100)	4.43e-05 (100)	3.51e-07 (100)	1.78e-09 (100)
g08	mean	-0.0958174	**-0.0958250**	**-0.0958250**	**-0.0958250**	**-0.0958250**	**-0.0958250**	**-0.0958250**
-0.095825	σ	1.63e-05 (100)	0.00e+00 (100)	7.24e-08 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g09	mean	**680.6300574**	**680.6300574**	680.6300937	**680.6300574**	**680.6300574**	**680.6300574**	**680.6300574**
680.630057	σ	0.00e+00 (100)	0.00e+00 (100)	3.10e-05 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g10	mean	8705.0397468	9592.1262915	8125.9859481	7049.2480232	**7049.2480205**	**7049.2480205**	**7049.2480205**
7049.248	σ	6.56e+03 (0)	5.69e+03 (0)	1.62e+02 (100)	2.90e-06 (100)	3.88e-12 (100)	0.00e+00 (100)	1.77e-09 (100)
g11	mean	**0.7499000**	**0.7499000**	0.7499084	**0.7499000**	**0.7499000**	**0.7499000**	**0.7499000**
0.749900	σ	0.00e+00 (100)	0.00e+00 (100)	6.98e-06 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g12	mean	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000	-1.0000000
-1.000	σ	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)	0.00e+00 (100)
g13	mean	0.6428133	0.7429714	0.9209897	**0.0539415**	0.0667702	0.0795989	**0.0539415**
0.0539415	σ	2.32e-01 (100)	1.68e-01 (100)	1.32e-01 (100)	2.79e-15 (100)	6.91e-02 (100)	9.60e-02 (100)	0.00e+00 (100)

Table 5 Results of Wilcoxon signed rank test for PSOEPCm against DEEPC.

	PSOEPCm Pm=0.25	PSOEPCm Pm=0.5	PSOEPCm Pm=0.75
g01	=	=	=
g02	=	$--$	$--$
g03	$--$	$-$	++
g04	=	=	=
g05	$--$	=	=
g06	=	=	=
g07	$--$	$--$	=
g08	=	=	=
g09	=	=	=
g10	$--$	++	++
g11	=	=	=
g12	=	=	=
g13	$--$	=	=
+	0	1	2
=	8	9	10
$-$	5	3	1

Table 6 Scalability for the number of dimensions D

D	mean	σ
50	0.9999134	1.30e-05
100	0.9999056	4.19e-06
200	0.9999023	1.96e-06
300	0.9999010	1.23e-06
400	0.9999012	2.09e-06
500	0.9999003	5.53e-07

Proc. of the 4th Annual Conference on Evolutionary Programming, The MIT Press, Cambridge, Massachusetts, pp. 135–155.
2. Coello C.A.C. (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. Computer Methods in Applied Mechanics and Engineering, 191(11-12):1245–1287.
3. Takahama T., Sakai S. (2005) Constrained optimization by applying the α constrained method to the nonlinear simplex method with mutations. IEEE Trans. on Evolutionary Computation, 9(5):437–451.
4. Coath G., Halgamuge S.K. (2003) A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems. In: Proc. of the 2003 IEEE Congress on Evolutionary Computation, Canberra, Australia, pp. 2419–2425.
5. Takahama T., Sakai S. (2019) An equivalent penalty coefficient method: An adaptive penalty approach for population-based constrained optimization. In: Proc. of the 2019 IEEE Congress on Evolutionary Computation, pp. 1621–1628.
6. Homaifar A., Lai S.H.Y., Qi X. (1994) Constrained optimization via genetic algorithms. Simulation, 62(4):242–254.
7. Joines J., Houck C. (1994) On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs. In: D. Fogel (Ed.), Proc. of the first IEEE Conference on Evolutionary Computation, IEEE Press, Orlando, Florida, pp. 579–584.
8. Michalewicz Z., Attia N. (1994) Evolutionary optimization of constrained problems. In: A. Sebald, L. Fogel (Eds.), Proc. of the 3rd Annual Conference on Evolutionary Programming, World Scientific Publishing, River Edge, NJ, pp. 98–108.
9. Coello C.A.C. (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Computers in Industry, 41(2):113–127.
10. Tessema B., Yen G. (2006) A self adaptive penalty function based algorithm for constrained optimization. In: G.G. Yen, S.M. Lucas, G. Fogel, G. Kendall, R. Salomon, B.T. Zhang, C.A.C. Coello, T.P. Runarsson (Eds.), Proceedings of the 2006 IEEE Congress on Evolutionary Computation, IEEE Press, Vancouver, BC, Canada, pp. 246–253.
11. Wang Y., Cai Z., Xhau Y., Zeng W. (2008) An adaptive tradeoff model for constrained evolutionary computation. IEEE Trans. on Evolutionary Computation, 12(1):80–92.
12. Deb K. (2000) An efficient constraint handling method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering, 186(2/4):311–338.
13. Takahama T., Sakai S. (2000) Tuning fuzzy control rules by the α constrained method which solves constrained

nonlinear optimization problems. Electronics and Communications in Japan, Part 3: Fundamental Electronic Science, 83(9):1–12.
14. Takahama T., Sakai S. (2005) Constrained optimization by ε constrained particle swarm optimizer with ε-level control. In: Proc. of the 4th IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology (WSTST'05), pp. 1019–1029.
15. Runarsson T.P., Yao X. (2000) Stochastic ranking for constrained evolutionary optimization. IEEE Trans. on Evolutionary Computation, 4(3):284–294.
16. Mezura-Montes E., Coello C.A.C. (2005) A simple multimembered evolution strategy to solve constrained optimization problems. IEEE Trans. on Evolutionary Computation, 9(1):1–17.
17. Venkatraman S., Yen G.G. (2005) A generic framework for constrained optimization using genetic algorithms. IEEE Trans. on Evolutionary Computation, 9(4):424–435.
18. Surry P.D., Radcliffe N.J. (1997) The COMOGA method: Constrained optimisation by multiobjective genetic algorithms. Control and Cybernetics, 26(3):391–412.
19. Coello C.A.C. (2000) Constraint-handling using an evolutionary multiobjective optimization technique. Civil Engineering and Environmental Systems, 17:319–346.
20. Ray T., Liew K.M., Saini P. (2002) An intelligent information sharing strategy within a swarm for unconstrained and constrained optimization problems. Soft Computing – A Fusion of Foundations, Methodologies and Applications, 6(1):38–44.
21. Runarsson T.P., Yao X. (2003) Evolutionary search and constraint violations. In: Proc. of the 2003 Congress on Evolutionary Computation, vol. 2, IEEE Service Center, Piscataway, New Jersey, pp. 1414–1419.
22. Aguirre A.H., Rionda S.B., Coello C.A.C., Lizárraga G.L., Montes E.M. (2004) Handling constraints using multiobjective optimization concepts. International Journal for Numerical Methods in Engineering, 59(15):1989–2017.
23. Wang Y., Cai Z., Cuo G., Zhou Z. (2007) Multiobjective optimization and hybrid evolutionary algorthm to solve constrained optimization problems. IEEE Trans. on Systems, Man and Cybernetics, Part B, 37(3):560–575.
24. Mallipeddi R., Suganthan P.N. (2010) Ensemble of constraint handling techniques. IEEE Transactions on Evolutionary Computation:561–579.
25. Tessema B., Yen G.G. (2009) An adaptive penalty formulation for constrained evolutionary optimization. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 39(3):565–578, doi: 10.1109/TSMCA.2009.2013333.
26. Barbosa H.J.C., Lemonge A.C.C. (2002) An adaptive penalty scheme in genetic algorithms for constrained optimization problems. In: Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 287–294.
27. Kennedy J., Eberhart R.C. (1995) Particle swarm optimization. In: Proc. of IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948.
28. Kennedy J., Eberhart R.C. (2001) Swarm Intelligence. Morgan Kaufmann, San Francisco.
29. Eberhart R., Shi Y. (2001) Particle swarm optimization: developments, applications and resources. In: Proc. of the 2001 Congress on Evolutionary Computation, pp. 81–86.
30. Engelbrecht A. (2013) Particle swarm optimization: Global best or local best? In: 2013 BRICS Congress on Computational Intelligence & 11th Brazilian Congress on Computational Intelligence, IEEE, pp. 124–135.
31. Shi Y., Eberhart R. (1999) Empirical study of particle swarm optimization. In: Proc. of the 1999 Congress on Evolutionary Computation, pp. 1945–1950.
32. Clerc M., Kennedy J. (2002) The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 6(1):58–73.
33. Eberhart R., Shi Y. (2000) Comparing inertia weights and constriction factors in particle swarm optimization. In: Proc. of the 2000 Congress on Evolutionary Computation, pp. 84–88.
34. Zhang J., Sanderson A.C. (2009) JADE: Adaptive differential evolution with optional external archive. IEEE Transactions on Evolutionary Computation, 13(5):945–958.
35. Farmani R., Wright J.A. (2003) Self-adaptive fitness formulation for constrained optimization. IEEE Trans. on Evolutionary Computation, 7(5):445–455.
36. Yan D., Lu Y. (2018) Recent advances in particle swarm optimization for large scale problems. Journal of Autonomous Intelligence, 1(1).