Constrained Optimization by ε Constrained Differential Evolution with Dynamic ε-Level Control

Tetsuyuki Takahama[1] and Setsuko Sakai[2]

[1] Department of Intelligent Systems, Hiroshima City University, Asaminami-ku, Hiroshima 731-3194 Japan
takahama@its.hiroshima-cu.ac.jp
[2] Faculty of Commercial Sciences, Hiroshima Shudo University, Asaminami-ku, Hiroshima 731-3195 Japan
setuko@shudo-u.ac.jp

Summary. In this chapter, the improved ε constrained differential evolution (εDE) is proposed to solve constrained optimization problems with very small feasible region, such as problems with equality constraints, efficiently. The εDE is the combination of the ε constrained method and differential evolution. In general, it is very difficult to solve constrained problems with very small feasible region. To solve such problems, static control schema of allowable constraint violation is often used, where solutions are searched within enlarged region specified by the allowable violation and the region is reduced to the feasible region gradually. However, the proper control depends on the initial population and searching process. In this study, the dynamic control of allowable violation is proposed to solve problems with equality constraints efficiently. In the εDE, the amount of allowable violation can be specified by the ε-level. The effectiveness of the εDE with dynamic ε-level control is shown by comparing with the original εDE and well known optimization method on some nonlinear constrained problems with equality constraints.

1 Introduction

Constrained optimization problems, especially nonlinear optimization problems, where objective functions are minimized under given constraints, are very important and frequently appear in the real world. In this study, the following optimization problem (P) with inequality constraints, equality constraints, upper bound constraints and lower bound constraints will be discussed.

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

where $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ is an n dimensional vector, $f(\boldsymbol{x})$ is an objective function, $g_j(\boldsymbol{x}) \le 0$ and $h_j(\boldsymbol{x}) = 0$ are q inequality constraints and $m - q$

U.K. Chakraborty (Ed.): Advances in Differential Evolution, SCI 143, pp. 139–154, 2008.
springerlink.com

equality constraints, respectively. Functions f, g_j and h_j are linear or nonlinear real-valued functions. Values u_i and l_i are the upper bound and the lower bound of x_i, respectively. Also, let the feasible space in which every point satisfies all constraints be denoted by $\mathcal{F}$ and the search space in which every point satisfies the upper and lower bound constraints be denoted by $\mathcal{S}\,(\supset \mathcal{F})$.

There exist many studies on solving constrained optimization problems using evolutionary algorithms[1] and particle swarm optimization[2]. These studies can be classified into several categories according to the way the constraints are treated as follows:

(1) Constraints are only used to see whether a search point is feasible or not[3]. In this category, the searching 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. In this category, generating initial feasible points is difficult and computationally demanding when the feasible region is very small. It is almost impossible to find initial feasible points in problems with equality constraints.

(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[4]. 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 difficulty of selecting an appropriate value for the penalty coefficient that adjusts the strength of the penalty.

(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. Takahama and Sakai proposed the α constrained method[5, 6] and the ε constrained method[7], which adopt a lexicographic ordering with relaxation of the constraints. Deb[8] proposed a method in which the extended objective function that realizes the lexicographic ordering is used. Runarsson and Yao[9] proposed the stochastic ranking method in which the stochastic lexicographic order, which ignores the constraint violation with some probability, is used. 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[10, 11, 12]. But in many cases, solving multiobjective optimization problems is a more difficult and expensive task than solving single objective optimization problems.

In this study, the improved ε constrained differential evolution (εDE), is proposed to solve constrained optimization problems with very small feasible region, such as problems with equality constraints, efficiently. The εDE is the combination of the ε constrained method and differential evolution. The ε constrained methods can convert algorithms for unconstrained problems to algorithms for

constrained problems using the ε-level comparison, which compares the search points based on the constraint violation of them. The ε constrained method is in the promising category (3) and is proposed based on the α constrained method. The α constrained method was applied to Powell's direct search method in [5, 6], the nonlinear simplex method by Nelder and Mead in [13, 14, 15], a genetic algorithm (GA) using linear ranking selection in [16, 17] and particle swarm optimization (PSO) in [18]. The ε constrained method was applied to PSO in [7, 19, 20], GA in [21] and differential evolution (DE)[22, 23].

In general, it is very difficult to solve constrained problems with very small feasible region. To solve such problems, static control schema of allowable constraint violation is often used, where solutions are searched within enlarged region specified by the allowable violation and the region is reduced to the feasible region gradually. However, the proper control depends on the initial population and searching process. It is very difficult to decide the control beforehand. In this study, the dynamic control of allowable violation is proposed to solve problems with equality constraints efficiently. In the εDE, the amount of allowable violation can be specified by the ε-level. The effectiveness of the εDE with dynamic ε-level control is shown by comparing with the εDE with static control and well known optimization method on some nonlinear constrained problems with equality constraints.

The rest of this chapter is organized as follows: Section 2 describes the ε constrained method briefly. Section 3 describes the improved εDE by introducing dynamic control of the ε-level. Section 4 presents experimental results on various benchmark problems discussed in [9]. Comparisons with the results in [9] are included in this section. Finally, Section 5 concludes with a brief summary of this chapter and a few remarks.

2 The ε Constrained Method

2.1 Constraint Violation and ε-Level Comparison

In the ε constrained method, constraint violation $\phi(\boldsymbol{x})$ is defined. The constraint violation can be given by the maximum of all constraints or the sum of all constraints.

$$\phi(\boldsymbol{x}) = \max\{\max_j\{0, g_j(\boldsymbol{x})\}, \max_j |h_j(\boldsymbol{x})|\} \tag{2}$$

$$\phi(\boldsymbol{x}) = \sum_j ||max\{0, g_j(\boldsymbol{x})\}||^p + \sum_j ||h_j(\boldsymbol{x})||^p \tag{3}$$

where p is a positive number.

The *ε-level comparison* is defined as an order relation on the set of $(f(\boldsymbol{x}), \phi(\boldsymbol{x}))$. If the constraint violation of a point is greater than 0, the point is not feasible and its worth is low. The ε-level comparisons are defined by a lexicographic order in which $\phi(\boldsymbol{x})$ proceeds $f(\boldsymbol{x})$, because the feasibility of $\boldsymbol{x}$ is more important than the minimization of $f(\boldsymbol{x})$.

Let $f_1\,(f_2)$ and $\phi_1\,(\phi_2)$ be the function values and the constraint violation at a point $\boldsymbol{x}_1\,(\boldsymbol{x}_2)$, respectively. Then, for any ε satisfying $\varepsilon \geq 0$, ε-level comparison $<_\varepsilon$ and $\leq_\varepsilon$ between (f_1, ϕ_1) and (f_2, ϕ_2) is defined as follows:

$$(f_1, \phi_1) <_\varepsilon (f_2, \phi_2) \Leftrightarrow \begin{cases} f_1 < f_2, \text{ if } \phi_1, \phi_2 \leq \varepsilon \\ f_1 < f_2, \text{ if } \phi_1 = \phi_2 \\ \phi_1 < \phi_2, \text{ otherwise} \end{cases} \tag{4}$$

$$(f_1, \phi_1) \leq_\varepsilon (f_2, \phi_2) \Leftrightarrow \begin{cases} f_1 \leq f_2, \text{ if } \phi_1, \phi_2 \leq \varepsilon \\ f_1 \leq f_2, \text{ if } \phi_1 = \phi_2 \\ \phi_1 < \phi_2, \text{ otherwise} \end{cases} \tag{5}$$

In case of $\varepsilon{=}\infty$, the ε-level comparison $<_\infty$ and $\leq_\infty$ are equivalent to the ordinal comparison $<$ and $\leq$ between function values. Also, in case of $\varepsilon = 0$, $<_0$ and $\leq_0$ are equivalent to the lexicographic order in which the constraint violation $\phi(\boldsymbol{x})$ precedes the function value $f(\boldsymbol{x})$.

2.2 The Properties of the ε Constrained Method

The ε constrained method converts a constrained optimization problem into an unconstrained one by replacing the order relation in direct search methods with the ε-level comparison. An optimization problem solved by the ε constrained method, that is, a problem in which the ordinary comparison is replaced with the ε-level comparison, $(\mathrm{P}_{\leq_\varepsilon})$, is defined as follows:

$$(\mathrm{P}_{\leq_\varepsilon}) \text{ minimize}_{\leq_\varepsilon}\ f(\boldsymbol{x}), \tag{6}$$

where $\text{minimize}_{\leq_\varepsilon}$ means the minimization based on the ε-level comparison $\leq_\varepsilon$. Also, a problem (P^ε) is defined that the constraints of (P), that is, $\phi(\boldsymbol{x}) = 0$, is relaxed and replaced with $\phi(\boldsymbol{x}) \leq \varepsilon$:

$$\begin{aligned} (\mathrm{P}^\varepsilon)\ &\text{minimize} \quad f(\boldsymbol{x}) \\ &\text{subject to } \phi(\boldsymbol{x}) \leq \varepsilon \end{aligned} \tag{7}$$

It is obvious that (P^0) is equivalent to (P).

For the three types of problems, (P^ε), $(\mathrm{P}_{\leq_\varepsilon})$ and (P), the following theorems are given based on the α constrained method[5, 6, 7].

Theorem 1. *If an optimal solution* (P^0) *exists, any optimal solution of* $(\mathrm{P}_{\leq_\varepsilon})$ *is an optimal solution of* (P^ε).

Theorem 2. *If an optimal solution of* (P) *exists, any optimal solution of* $(\mathrm{P}_{\leq_0})$ *is an optimal solution of* (P).

Theorem 3. *Let* $\{\varepsilon_n\}$ *be a strictly decreasing non-negative sequence and converge to 0. Let* $f(\boldsymbol{x})$ *and* $\phi(\boldsymbol{x})$ *be continuous functions of* $\boldsymbol{x}$. *Assume that an optimal solution* $\boldsymbol{x}^*$ *of* (P^0) *exists and an optimal solution* $\hat{\boldsymbol{x}}_n$ *of* $(\mathrm{P}_{\leq_{\varepsilon_n}})$ *exists for any* ε_n. *Then, any accumulation point to the sequence* $\{\hat{\boldsymbol{x}}_n\}$ *is an optimal solution of* (P^0).

Theorem 1 and 2 show that a constrained optimization problem can be transformed into an equivalent unconstrained optimization problem by using the ε-level comparison. So, if the ε-level comparison is incorporated into an existing unconstrained optimization method, constrained optimization problems can be solved. Theorem 3 shows that, in the ε constrained method, an optimal solution of (P^0) can be given by converging ε to 0 as well as by increasing the penalty coefficient to infinity in the penalty method.

3 The εDE

In this section, we first describe differential evolution. Then, we describe the εDE, which is the integration of the ε constrained method and DE. Static and dynamic control functions of relaxing equality constraints are also defined.

3.1 Differential Evolution

Differential evolution is an evolutionary algorithm proposed by Storn and Price [24, 25]. DE is a stochastic direct search method using population or multiple search points. DE has been successfully applied to the optimization problems including non-linear, non-differentiable, non-convex and multi-modal functions. It has been shown that DE is fast and robust to these functions.

The main feature of DE is that DE uses simple arithmetic operations to avoid the control of Gaussian mutation adopted in evolution strategy. In general, the mutation process must be adaptive to the step size of the Gaussian mutation, because the ideal step size depends on the gene or element that is mutated and the state of the evolution process. DE adopts the sum of a base vector and the scaled difference vectors as the mutation operation instead of Gaussian mutation. The base vector is an individual selected from the population. The difference vectors are formed by the differences between a pair of individuals randomly selected from the population. As the search space by the population contracts and expands over generations, the step size in each dimension, which is given by the difference vectors, adapts automatically.

There are some variants of DE that have been proposed, such as DE/best/1/bin and DE/rand/1/exp. The variants are classified using the notation DE/*base*/*num*/*cross*. "*base*" indicates the method of selecting a parent that will form the base vector. For example, DE/rand/*num*/*cross* selects the parent for the base vector at random from the population. DE/best/*num*/*cross* selects the best individual in the population. "*num*" indicates the number of difference vectors used to perturb the base vector. "*cross*" indicates the crossover mechanism used to create a child. For example, DE/*base*/*num*/bin shows that crossover is controlled by binomial crossover using constant crossover rate. DE/*base*/*num*/exp shows that crossover is controlled by a binomial crossover using exponentially decreasing the crossover rate.

In DE, initial individuals are randomly generated within the search space and form an initial population. Each individual contains n genes as decision variables or a decision vector. At each generation or iteration, all individuals are selected as parents. Each parent is processed as follows: The mutation process begins by choosing $1 + 2\,num$ individuals from the parents except for the parent in the processing. The first individual is a base vector. All subsequent individuals are paired to create num difference vectors. The difference vectors are scaled by the scaling factor F and added to the base vector. The resulting vector is then recombined or crossovered with the parent. The probability of recombination at an element is controlled by the crossover factor CR. This crossover process produces a trial vector. Finally, for survivor selection, the trial vector is accepted for the next generation if the trial vector is better than the parent.

3.2 The Algorithm of the εDE

The algorithm of the εDE based on DE/rand/1/exp variant, which is used in this study, is as follows:

Step0 Initialization. Initial N individuals $\boldsymbol{x}^i$ are generated as the initial search points, where there is an initial population $P(0) = \{\boldsymbol{x}^i, i = 1, 2, \cdots, N\}$. An initial ε-level is given by the ε-level control function $\varepsilon(0)$.

Step1 Termination condition. If the number of generations (iterations) exceeds the maximum generation $T_{\max}$, the algorithm is terminated.

Step2 Mutation. For each individual $\boldsymbol{x}^i$, three different individuals $\boldsymbol{x}^{p1}$, $\boldsymbol{x}^{p2}$ and $\boldsymbol{x}^{p3}$, each of which is also different from $\boldsymbol{x}^i$, are chosen from the population. A new vector $\boldsymbol{x}'$ is generated by the base vector $\boldsymbol{x}^{p1}$ and the difference vector $\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3}$ as follows:

$$\boldsymbol{x}' = \boldsymbol{x}^{p1} + F(\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3}) \tag{8}$$

where F is a scaling factor.

Step3 Crossover. The vector $\boldsymbol{x}'$ is crossovered with the parent $\boldsymbol{x}^i$. A crossover point j is chosen randomly from all dimensions $[1, n]$. The element at the j-th dimension of the trial vector $\boldsymbol{x}^{\text{new}}$ is inherited from the j-th element of the vector $\boldsymbol{x}'$. The elements of subsequent dimensions are inherited from $\boldsymbol{x}'$ with exponentially decreasing probability defined by a crossover factor CR. Otherwise, the elements are inherited from the parent $\boldsymbol{x}^i$. In real processing, Step2 and Step3 are integrated as one operation.

Step4 Survivor selection. The trial vector $\boldsymbol{x}^{\text{new}}$ is accepted for the next generation if the trial vector is better than the parent $\boldsymbol{x}^i$.

Step5 Controlling the ε-level. The ε-level is updated by the ε-level control function $\varepsilon(t)$.

Step6 Go back to Step1.

Fig. 1 shows the algorithm of the εDE.

```
εDE/rand/1/exp()
{
  P(0)=Generate N individuals {x^i} randomly;
  ε=ε(0);
  for(t=1; t ≤ T_max; t++) {
    for(i=1; i ≤ N; i++) {
      (p1, p2, p3)=select randomly from [1, N]
                   s.t. p1 ≠ p2 ≠ p3 ≠ i;
      x^new=x^i ∈ P(t − 1);
      j=select randomly from [1, n];
      k=1;
      do {
        x_j^new=x_j^p1+F(x_j^p2 − x_j^p3);
        j=(j + 1)%n;
        k++;
      } while(k ≤ n && u(0, 1) < CR);
      if((f(x^new), φ(x^new)) <_ε (f(x^i), φ(x^i)))
        z^i=x^new;
      else
        z^i=x^i;
    }
    P(t)={z^i, i = 1, 2, ···, N}
    ε=ε(t);
  }
}
```

Fig. 1. The algorithm of the ε constrained differential evolution with control of the ε-level, where $\varepsilon(t)$ is the ε-level control function, F is a scaling factor, CR is a crossover factor, and $u(0,1)$ is a uniform random number generator in $[0,1]$

3.3 Controlling the ε-Level

Usually, the ε-level does not need to be controlled. Many constrained problems can be solved based on the lexicographic order where the ε-level is constantly 0. However for problems with equality constraints, the ε-level should be controlled properly to obtain high quality solutions.

Static control

A simple static control of the ε-level proposed in [7] can be defined according to the equation (9). The initial ε-level $\varepsilon(0)$ is the constraint violation of the top θ-th individual in the initial search points. The ε-level is updated until the number of iterations t becomes the control generation T_c. After the number of iterations exceeds T_c, the ε-level is set to 0 to obtain solutions with minimum constraint violation.

$$\begin{aligned} \varepsilon_s(0) &= \phi(\boldsymbol{x}_\theta) \\ \varepsilon_s(t) &= \begin{cases} \varepsilon_s(0)(1-\frac{t}{T_c})^{cp}, & 0 < t < T_c, \\ 0, & t \geq T_c \end{cases} \end{aligned} \tag{9}$$

where $\boldsymbol{x}_\theta$ is the top θ-th individual. When θ is too small, although a feasible solution can be found rapidly, the stability to find an optimal solution becomes low. When θ is too large, although the optimal solution can be found, the efficiency to find feasible region and the optimal solution becomes low. By using $\theta = 0.2N$, the stability and the efficiency can be balanced in many problems.

Dynamic control

To improve the efficiency of the εDEwith static control, we propose dynamic control of the ε-level according to the equation (10). Modified generation t' ($t' \geq t$) is introduced to speed up the convergence of enlarged region into the feasible region. If the violation is reduced enough in search process, the generation t' is increased faster than usual generation t and the ε-level is decreased faster. If not, the generation t' is increased by 1 like usual generation t.

$$\begin{aligned} \varepsilon_d(0) &= \phi(\boldsymbol{x}_\theta) \\ \varepsilon_d(t) &= \begin{cases} \varepsilon_d(0)(1 - \frac{t'}{T_c})^{cp}, & 0 < t' < T_c, \\ 0, & t' \geq T_c \end{cases} \end{aligned} \tag{10}$$

The modified generation t' is updated in each generation as follows.

$$t' = \begin{cases} 0, & t = 0, \\ t' + 1, & \phi(\boldsymbol{x}_\eta) \geq \varepsilon_d(t) \\ t' + 2, & \phi(\boldsymbol{x}_\eta) < \varepsilon_d(t) \text{ and } t' + 2 \geq \varepsilon_s^{-1}(\phi(\boldsymbol{x}_\eta)) \\ \frac{1}{2}(t' + 2) + \frac{1}{2}\varepsilon_s^{-1}(\phi(\boldsymbol{x}_\eta)), & \text{otherwise} \end{cases} \tag{11}$$

where $\boldsymbol{x}_\eta$ is the worst η-th individual. $\varepsilon_s^{-1}(\varepsilon)$ is the inverse function of $\varepsilon_s(t)$ that returns the generation of the ε-level being ε in $\varepsilon_s(t)$ and is defined as follows.

$$\varepsilon_s^{-1}(\varepsilon) = \left(1 - \sqrt[cp]{\frac{\varepsilon}{\varepsilon_d(0)}}\right) T_c \tag{12}$$

4 Solving Constrained Nonlinear Programming Problems with Equality Constraints

In this section, four benchmark problems that are mentioned in some studies [9, 26, 15] are optimized.

4.1 Test Problems and Experimental Conditions

Four test problems, which are nonlinear optimization problems with equality constraints, are shown as follows.

g03 [27]:

$$\begin{aligned}
\text{maximize} \quad & f(\boldsymbol{x}) = (\sqrt{n})^n \prod_{i=1}^{n} x_i, \\
\text{subject to} \quad & h_1(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2 - 1 = 0, \\
& 0 \le x_i \le 1 \, (i = 1, \cdots, n), \quad n = 10
\end{aligned}$$

The optimal solution $\boldsymbol{x}_i^* = \dfrac{1}{\sqrt{n}}$ $(i = 1, \cdots, n)$ and the optimal value $f(\boldsymbol{x}^*) = 1$.

g05 [28]:

$$\begin{aligned}
\text{minimize} \quad & f(\boldsymbol{x}) = 3x_1 + 0.000001x_1^3 + 2x_2 + \frac{0.000002}{3}x_2^3, \\
\text{subject to} \quad & g_1(\boldsymbol{x}) = x_3 - x_4 - 0.55 \le 0, \\
& g_2(\boldsymbol{x}) = -x_3 + x_4 - 0.55 \le 0, \\
& h_3(\boldsymbol{x}) = 1000 \sin(-x_3 - 0.25) + 1000 \sin(-x_4 - 0.25) \\
& \qquad +894.8 - x_1 = 0, \\
& h_4(\boldsymbol{x}) = 1000 \sin(x_3 - 0.25) + 1000 \sin(x_3 - x_4 - 0.25) \\
& \qquad +894.8 - x_2 = 0, \\
& h_5(\boldsymbol{x}) = 1000 \sin(x_4 - 0.25) + 1000 \sin(x_4 - x_3 - 0.25) \\
& \qquad +1294.8 = 0, \\
& 0 \le x_i \le 1200 \, (i = 1, 2), \quad -0.55 \le x_i \le 0.55 \, (i = 3, 4)
\end{aligned}$$

The minimum value is unknown. The known best value is $f(\boldsymbol{x}) = 5126.4981$ [29].

g11 [29]:

$$\begin{aligned}
\text{minimize} \quad & f(\boldsymbol{x}) = x_1^2 + (x_2 - 1)^2, \\
\text{subject to} \quad & h(\boldsymbol{x}) = x_2 - x_1^2 = 0, \\
& -1 \le x_i \le 1 \, (i = 1, 2)
\end{aligned}$$

The optimal solution is $\boldsymbol{x}^* = \left(\pm \dfrac{1}{\sqrt{2}}, \dfrac{1}{2} \right)$ and the optimal value $f(\boldsymbol{x}^*) = 0.75$.

g13 [28]:

$$\begin{aligned}
\text{minimize} \quad & f(\boldsymbol{x}) = e^{x_1 x_2 x_3 x_4 x_5}, \\
\text{subject to} \quad & h_1(\boldsymbol{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0, \\
& h_2(\boldsymbol{x}) = x_2 x_3 - 5 x_4 x_5 = 0, \\
& h_3(\boldsymbol{x}) = x_1^3 + x_2^3 + 1 = 0, \\
& -2.3 \le x_i \le 2.3 \, (i = 1, 2), \ -3.2 \le x_i \le 3.2 \, (i = 3, 4, 5)
\end{aligned}$$

The optimal solution is $\boldsymbol{x}^* = (-1.717143, 1.595709, 1.827247, -0.7636413, -0.763645)$ and the optimal value $f(\boldsymbol{x}^*) = 0.0539498$.

In many other methods, problems with equality constraints cannot be solved directly. Thus, the equality constraints are relaxed, that is, all equality constraints $h_j(\boldsymbol{x}) = 0, \ j = q + 1, \cdots, m$ are replaced by inequalities:

Table 1. Results using static control and dynamic control; 25 independent runs

Problem	Control	Best	Mean	Median	Worst
↑g03	static	1.00050010	1.00050010	1.00050010	1.00050010
(1.0)	dynamic	1.00050010	1.00050010	1.00050010	1.00050010
g05	static	5126.49671	5126.49671	5126.49671	5126.49671
(5126.498)	dynamic	5126.49671	5126.49671	5126.49671	5126.49671
g11	static	0.74990000	0.74990000	0.74990000	0.74990000
(0.750)	dynamic	0.74990000	0.74990000	0.74990000	0.74990000
g13	static	0.05394151	0.05394151	0.05394151	0.05394151
(0.053950)	dynamic	0.05394151	0.05394151	0.05394151	0.05394151

$$|h_j(\boldsymbol{x})| \leq \delta,\ \delta > 0 \tag{13}$$

In the experiments, $\delta = 0.001$.

In εDE, the same settings are used for all problems. The parameters for the ε constrained method are defined as follows: The constraint violation ϕ is given by the sum of all constraints (p=1) in equation (3). The ε-level is controlled by equation (9) in static control and (10) in dynamic control where $T_c = 0.5T$. The εDE can solve problems with equality constraints directly. However, to compare with other method, $\delta = 0.001$ is tested. The parameters for DE are as follows: The number of agents $N = 40$, $F = 0.7$, $CR = 0.9$. The maximum number of generation $T = 5000$, and independent 25 runs are performed in each problem.

4.2 Experimental Results

Experimental results on the test problems are shown in Table 1, in which each value is the average of 25 runs. The column labeled "problem" shows the problem number. The optimal value in each problem is shown in parentheses under the problem number. The column labeled "control" shows the type of control for the ε-level, where the parameter for dynamic control is $\eta = 5$. Also, "best", "mean", "median", and "worst" are the best value, the average value, the median value, the worst value, respectively. Problem g03 is the maximization problem and is shown with an up arrow.

In all problems, both of εDE with static control and εDE with dynamic control found the same solutions in all runs. In all cases, the εDEs found smaller values than the optimal values. The solutions obtained by the εDEs were away about 0.001 from the feasible region because the constraints are relaxed with equation (13). Thus, it is thought that the εDEs' ability to search for feasible solutions is very high for the problems with the equality constraints. This result shows that both of static and dynamic control have equivalent ability to find optimal solutions.

To compare the efficiency of the static control and the dynamic control, another experiment is performed with changing the parameter $\eta = 0, 1, 2, 3, 4, 5, 6$.

Table 2. Results with changing the parameter η; 25 independent runs

Problem	Param	Success	Best	Worst	Mean	Sigma	Ratio
g03	static	25	87,031	92,438	90,034.2	1,347.6	1
	0	25	79,062	84,124	81,821.4	1,309.3	0.91
	1	25	63,647	72,966	69,184.8	2,281.1	0.77
	2	25	55,334	62,539	59,032.2	1,849.0	0.66
	3	25	47,261	54,213	51,170.0	1,861.2	0.57
	4	25	41,342	**48,950**	**44,467.4**	2,033.8	**0.49**
	5	25	**36,584**	69,402	45,551.7	6,118.2	0.51
	6	19	39,870	105,202	73,796.9	17,928.0	0.82
g05	static	25	96,688	98,290	97,572.0	362.0	1
	0	25	94,531	97,058	95,850.9	572.2	0.98
	1	25	91,776	94,321	93,099.6	732.8	0.95
	2	25	87,248	90,730	88,980.2	817.7	0.91
	3	25	81,998	85,982	84,225.6	1058.3	0.86
	4	25	76,026	81,286	78,620.8	1,299.8	0.81
	5	25	71,180	76,797	73,722.4	1,296.6	0.76
	6	25	**65,644**	**71,312**	**67,783.2**	1,315.4	**0.69**
g11	static	25	22,558	62,184	45,046.8	9,330.4	1
	0	25	7,044	47,736	34,894.0	8,811.0	0.77
	1	25	21,714	49,530	34,130.1	7,496.9	0.76
	2	25	12,545	41,469	30,411.5	6,805.9	0.68
	3	25	7,810	37,409	23,740.1	6,818.6	0.53
	4	25	17,403	33,025	26,380.2	4,100.5	0.59
	5	25	**4,676**	29,511	19,533.7	5,873.8	0.43
	6	25	5,918	**28,345**	**18,572.0**	6,849.0	**0.41**
g13	static	25	76,582	88,947	85,037.3	2,694.5	1
	0	25	75,854	84,101	80,048.8	2,178.8	0.94
	1	25	67,573	78,873	73,090.9	2,396.2	0.86
	2	25	57,850	70,317	63,716.6	2,902.4	0.75
	3	25	51,056	59,640	55,575.9	1,902.8	0.65
	4	25	42,432	51,819	47,945.0	2,401.3	0.56
	5	25	37,404	46,565	42,308.2	2,562.8	0.50
	6	25	**31,639**	**40,491**	**36,964.5**	2,216.6	**0.43**

Table 2 shows the number of function evaluations needed for satisfying the success condition of $f^{\text{best}} - f^* \leq 0.0001$ and $\boldsymbol{x}^{\text{best}}$ being feasible, where $\boldsymbol{x}^{\text{best}}$ and f^{best} are the best solution found in each run and its value, respectively. The column labeled "param" shows the parameter value of η for dynamic control, and the result of static control is also shown. The column labeled "success" shows the number of runs in which solutions satisfying success condition are found. Also, "best", "worst", "mean" and "sigma" are the best value, the worst value, the average value and the standard deviation of the number of function evaluations for satisfying success condition. The ratio of the number of function evaluations

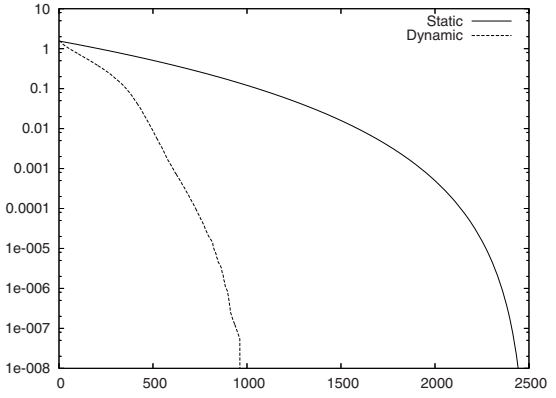

Fig. 2. The control of the ε-level in problem `g03`

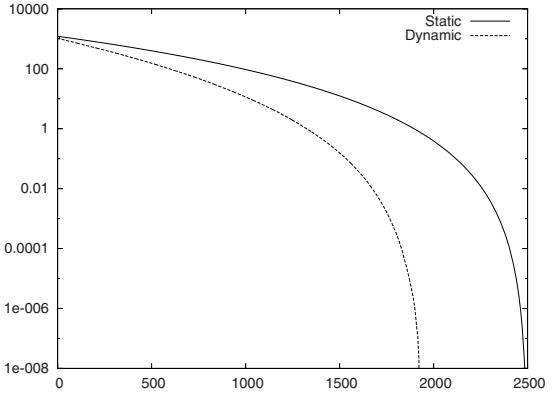

Fig. 3. The control of the ε-level in problem `g05`

between the static control and dynamic control is shown in the column labeled "ratio". The better cases are highlighted using boldface.

When the parameter η is in $[0, 5]$, solutions satisfying success condition are found in all runs. If η is large, the convergence speed of enlarged region into feasible region becomes high and feasible solutions and the optimal solution can be found faster. When the parameter η is 5, the number of function evaluations is reduced about 50% in `g03`, `g11` and `g13` and it is reduced about 24% in `g05`. So, the efficiency of the εDE with dynamic control is very higher than that of the εDE with static control. However, when η is 6, success condition cannot be attained in 6 runs out of 25 runs for `g03`. It is thought that the decrease of the ε-level is too fast, enlarged region is reduced too fast, and search process cannot find the optimal solution. Thus, the value of η should be selected properly to find optimal solutions, and the proper value of parameter η is 5.

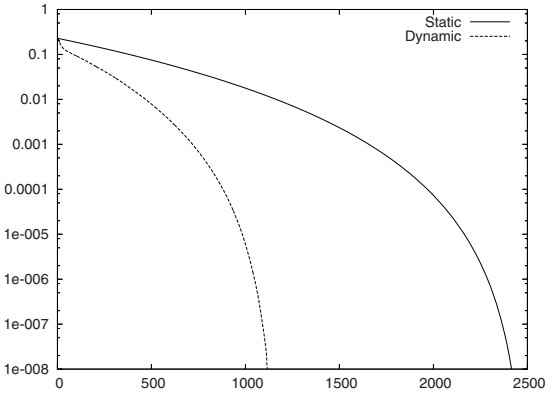

Fig. 4. The control of the ε-level in problem `g11`

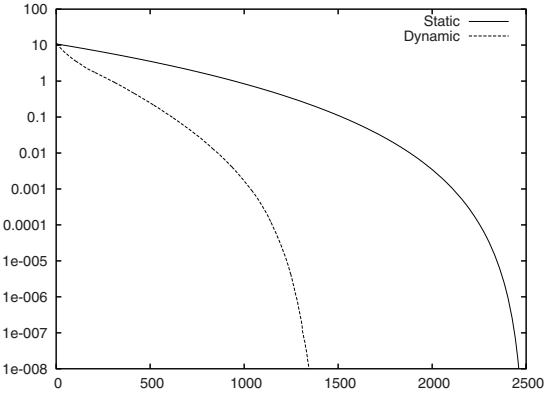

Fig. 5. The control of the ε-level in problem `g13`

Figures 2, 3, 4 and 5 show the change of the ε-level over generations for static control and dynamic control with $\eta = 5$ in problems `g03`, `g05`, `g11` and `g13`, respectively. Apparently, the convergence speed of the ε-level in dynamic control with $\eta = 5$ is higher than that in static control. These graphs show the effectiveness of the dynamic control.

4.3 Comparison with the Stochastic Ranking Method

To show the effectiveness of the εDE with dynamic control, the solutions found by this method are compared to those found by Runarsson and Yao's stochastic ranking method[9]. In [9], the maximum number of evaluations in each run was $1750 \times 200 = 350,000$ and all equality constraints were relaxed using $\delta = 10^{-4}$. Table 3 shows the comparison of the two methods. The better cases are highlighted using boldface. The results of the εDE were taken from Table 1 where

Table 3. Comparison between our (indicated by εDE) and Runarsson and Yao's (indicated by RY[9]) algorithms

f	Optimal	Best Result		Median Result		Mean Result		Worst Result	
		εDE	RY	εDE	RY	εDE	RY	εDE	RY
†g03	1.000	**1.001**	1.000	**1.001**	1.000	**1.001**	1.000	**1.001**	1.000
g05	5126.498	5126.497	5126.497	**5126.497**	5127.372	**5126.497**	5128.881	**5126.497**	5142.472
g11	0.750	0.750	0.750	0.750	0.750	0.750	0.750	0.750	0.750
g13	0.053950	**0.053942**	0.053957	**0.053942**	0.057006	**0.053942**	0.067543	**0.53942**	0.438803

all equality constraints were relaxed using $\delta = 10^{-4}$ and the maximum number of evaluations was $5000 \times 40 = 200,000$ which was much less than that of the stochastic ranking method. The solutions found by the stochastic ranking method were very high quality solutions that were equivalent to the known optimal solutions. Nevertheless, the εDE found better solutions for problems g03 and g13. Also, the stability of the εDE was better than that of the stochastic ranking method for problems g05 and g13. Therefore, the performance of the εDE is better than the stochastic ranking method.

5 Conclusions

This chapter presented the improved εDE with dynamic ε-level control to solve problems with very small feasible region, such as problems with equality constraints. By applying the εDE to the four constrained optimization problems with equality constraints, it was shown that the εDE obtained the optimal solution for every problem by the numerical experiments and the εDE was a high precision and stable optimization algorithm. By comparing the εDE with static control, we showed that the improved εDE was a very efficient algorithm. Also, by comparing the εDE with stochastic ranking method that is known as an efficient algorithm for the constrained optimization problems, it was shown that the εDE was a very stable and good algorithm.

In the future, we will explore ways to prevent the ε-level from converging too fast. Also, we will apply the εDE to various application fields.

Acknowledgements

This research is supported in part by Grant-in-Aid for Scientific Research (C) (No. 16500083, 17510139) of Japan society for the promotion of science and Hiroshima City University Grant for Special Academic Research (General Studies) 7111.

References

1. Michalewicz, Z.: A survey of constraint handling techniques in evolutionary computation methods. In: Proceedings of the 4th Annual Conference on Evolutionary Programming, pp. 135–155. MIT Press, Cambridge (1995)

2. Coath, G., Halgamuge, S.K.: A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems. In: Proc. of IEEE Congress on Evolutionary Computation, Canberra, Australia, pp. 2419–2425 (2003)
3. Hu, X., Eberhart, R.C.: Solving constrained nonlinear optimization problems with particle swarm optimization. In: Proc. of the Sixth World Multiconference on Systemics, Cybernetics and Informatics, Orlando, Florida (2002)
4. Parsopoulos, K.E., Vrahatis, M.N.: Particle swarm optimization method for constrained optimization problems. In: Sincak, P., Vascak, J., et al. (eds.) Intelligent Technologies — Theory and Application: New Trends in Intelligent Technologies. Frontiers in Artificial Intelligence and Applications, vol. 76, pp. 214–220. IOS Press, Amsterdam (2002)
5. Takahama, T., Sakai, S.: Tuning fuzzy control rules by α constrained method which solves constrained nonlinear optimization problems. The Transactions of the Institute of Electronics, Information and Communication Engineers J82-A(5), 658–668 (1999) (in Japanese)
6. Takahama, T., Sakai, S.: Tuning fuzzy control rules by the α constrained method which solves constrained nonlinear optimization problems. Electronics and Communications in Japan, Part3: Fundamental Electronic Science 83(9), 1–12 (2000)
7. Takahama, T., Sakai, S.: 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 2005), May 2005, pp. 1019–1029 (2005)
8. Deb, K.: An efficient constraint handling method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering 186(2/4), 311–338 (2000)
9. Runarsson, T.P., Yao, X.: Stochastic ranking for constrained evolutionary optimization. IEEE Transactions on Evolutionary Computation 4(3), 284–294 (2000)
10. Camponogara, E., Talukdar, S.N.: A genetic algorithm for constrained and multiobjective optimization. In: Alander, J.T. (ed.) 3rd Nordic Workshop on Genetic Algorithms and Their Applications (3NWGA), August 1997, pp. 49–62. University of Vaasa, Vaasa, Finland (1997)
11. Surry, P.D., Radcliffe, N.J.: The COMOGA method: Constrained optimisation by multiobjective genetic algorithms. Control and Cybernetics 26(3), 391–412 (1997)
12. Ray, T., Liew, K.M., Saini, P.: 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 (2002)
13. Takahama, T., Sakai, S.: Learning fuzzy control rules by α-constrained simplex method. The Transactions of the Institute of Electronics, Information and Communication Engineers J83-D-I(7), 770–779 (2000) (in Japanese)
14. Takahama, T., Sakai, S.: Learning fuzzy control rules by α-constrained simplex method. Systems and Computers in Japan 34(6), 80–90 (2003)
15. Takahama, T., Sakai, S.: Constrained optimization by applying the α constrained method to the nonlinear simplex method with mutations. IEEE Transactions on Evolutionary Computation 9(5), 437–451 (2005)
16. Takahama, T., Sakai, S.: Constrained optimization by α constrained genetic algorithm (αGA). The Transactions of the Institute of Electronics, Information and Communication Engineers J86-D-I(4), 198–207 (2003) (in Japanese)
17. Takahama, T., Sakai, S.: Constrained optimization by α constrained genetic algorithm (αGA). Systems and Computers in Japan 35(5), 11–22 (2004)

18. Takahama, T., Sakai, S.: Constrained optimization by combining the α constrained method with particle swarm optimization. In: Proc. of Joint 2nd International Conference on Soft Computing and Intelligent Systems and 5th International Symposium on Advanced Intelligent Systems (2004)
19. Takahama, T., Sakai, S., Iwane, N.: Constrained optimization by the ε constrained hybrid algorithm of particle swarm optimization and genetic algorithm. In: Zhang, S., Jarvis, R. (eds.) AI 2005. LNCS (LNAI), vol. 3809, pp. 389–400. Springer, Heidelberg (2005)
20. Takahama, T., Sakai, S.: Solving constrained optimization problems by the ε constrained particle swarm optimizer with adaptive velocity limit control. In: Proc. of the 2nd IEEE International Conference on Cybernetics & Intelligent Systems, June 2006, pp. 683–689 (2006)
21. Takahama, T., Sakai, S.: Constrained optimization by the ε constrained genetic algorithm. IPSJ Journal 47(6), 1861–1871 (2006)
22. Takahama, T., Sakai, S., Iwane, N.: Solving nonlinear constrained optimization problems by the ε constrained differential evolution. In: Proc. of the 2006 IEEE Conference on Systems, Man, and Cybernetics, October 2006, pp. 2322–2327 (2006)
23. Takahama, T., Sakai, S.: Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites. In: Proc. of 2006 IEEE Congress on Evolutionary Computation, July 2006, pp. 308–315 (2006)
24. Storn, R., Price, K.: Minimizing the real functions of the ICEC 1996 contest by differential evolution. In: Proc. of the International Conference on Evolutionary Computation, pp. 842–844 (1996)
25. Storn, R., Price, K.: Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11, 341–359 (1997)
26. Mezura-Montes, E., Coello, C.A.C.: A simple multimembered evolution strategy to solve constrained optimization problems. IEEE Trans. on Evolutionary Computation 9(1), 1–17 (2005)
27. Mchalewicz, Z., Nazhiyath, G., Michalewicz, M.: A note on usefullness of geometricalcrossover of numerical optimization problems. In: Fogel, L.J., Angeline, P.J., Bäck, T. (eds.) Proc. 5th Annual Conference on Evolutionary Programming, pp. 305–312. MIT Press, Cambridge (1996)
28. Hock, W., Schittkowski, K.: Test examples for nonlinear programming codes. Lecture Notes in Economics and Mathematical Systems. Springer, Heidelberg (1981)
29. Koziel, S., Michalewicz, Z.: Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evolutionary Computation 7(1), 19–44 (1999)