Lagrangian-Based Evolutionary Programming for Constrained Optimization

Hyun Myung and Jong-Hwan Kim

Dept. of EE, KAIST, 373-1 Kusong-dong,
Yusong-gu, Taejon, 305-701, Korea
{myung, johkim}@vivaldi.kaist.ac.kr

Abstract. In this paper, Lagrangian-based evolutionary programming, **Evolian** is proposed for the general constrained optimization problem, which incorporates the concept of (1) multi-phase optimization process and (2) constraint scaling techniques to resolve the ill-conditioning problem. In each phase of Evolian, the typical EP is performed using augmented Lagrangian objective function with a penalty parameter fixed. If there is no improvement in the best objective function in one phase, another phase is performed after scaling the constraints, and updating the Lagrange multipliers and penalty parameter. Simulation results indicate that Evolian gives outperforming or at least reasonable solutions for multivariable heavily constrained optimization problem compared to other several evolutionary computation based methods.

1 Introduction

Recently constraint handling techniques in evolutionary computation methods have been developed by some researchers [1, 2]. They are classified into several categories such as methods based on (1) penalty functions, (2) specialized operators, (3) the assumption of the superiority of feasible solutions over unfeasible solutions, (4) multi–objective optimization techniques, (5) co–evolutionary models, and (6) cultural algorithms. These methods, of course, have their own merits and drawbacks for nonlinear programming problems, however, they are unlikely to provide exact solutions for heavily constrained optimization problems.

A hybrid method which consists of evolutionary and deterministic optimization procedures was proposed [3]. Although the hybrid EP (Evolutionary Programming) applied to a series of nonlinear and quadratic optimization problems has proved useful when addressing heavily constrained optimization problems in terms of computational efficiency and solution accuracy, the hybrid EP offers an exact solution only when the mathematical form of the function to be minimized and its gradient information are known.

To remove such restrictions, Two–Phase EP (TPEP) method based on the hybrid method was proposed [4, 5]. TPEP consists of the standard EP as the first phase and the elitist EP with deterministic ranking strategy as the second phase. Using Lagrange multipliers and putting gradual emphasis on violated constraints in the objective function, the trial solutions are driven to the optimal point where all constraints are satisfied. In many test cases, TPEP was shown to effectively face with heavily constrained problems, except the case where the problem is ill-conditioned, e.g., one constraint function

is of different magnitude or changes more rapidly than the other constraint functions or the objective function and dominates the optimization process. Moreover, it would be necessary to introduce another subsequent phase if the dimension and the difficulty of the given problem are not manageable even by the second phase of TPEP.

In this paper, **Evolian** (**Evo**lutionary **O**ptimization based on **Lagrangian**), the extended version of TPEP, is proposed, which incorporates multi-phase optimization process and constraint scaling techniques. Scaling of the constraint functions eliminates the need to elaborately design the proper penalty parameter selection scheme or the cooling scheme. In each phase of Evolian, the typical EP is performed using augmented Lagrangian objective function with the penalty parameter fixed. If there is no improvement in the best objective function in one phase, another phase of Evolian is performed after scaling the constraints and then updating the Lagrange multipliers and penalty parameter. Through repeated execution of this procedure, we can obtain desired solutions.

2 Constrained Optimization Problem

The general constrained optimization problem (P) for continuous variables is defined as:

$$\text{Minimize } f(\mathbf{x}) \text{ subject to constraints}$$

$$g_1(\mathbf{x}) \leq 0, \cdots, g_r(\mathbf{x}) \leq 0, \quad h_1(\mathbf{x}) = 0, \cdots, h_m(\mathbf{x}) = 0 \tag{1}$$

where f and the g_k's are functions on R^n and the h_j's are functions on R^n for $m \leq n$, $\mathbf{x} = [x_1, \cdots, x_n]^T \in R^n$, and $\mathbf{x} \in \mathcal{F} \subseteq \mathcal{S}$. The set $\mathcal{S} \subseteq R^n$ defines the *search* space and the set $\mathcal{F} \subseteq \mathcal{S}$ defines a *feasible* part of the search space. Usually, the search space $\mathcal{S}$ is defined as an n-dimensional rectangle in R^n, e.g., $l_i \leq x_i \leq r_i$, $i = 1, \cdots, n$, whereas the feasibility set $\mathcal{F}$ is defined by the search space $\mathcal{S}$ and the constraints expressed by (1).

Recently several methods for handling infeasible solutions for continuous numerical optimization problems have emerged for the case of $\mathcal{F} \subset \mathcal{S}$. Some of them are based on the penalty function, however, they differ in methods how the penalty function is designed and applied to infeasible solutions. Yet they commonly use the cost function f and the constraint violation measure, i.e., the penalty function $\Phi_p(\mathbf{x})$ for the $r + m$ constraints usually defined as $\Phi_p(\mathbf{x}) = \sum_{k=1}^{r} g_k^+(\mathbf{x}) + \sum_{j=1}^{m} |h_j(\mathbf{x})|$, or $\Phi_p(\mathbf{x}) = \frac{1}{2}\left[\sum_{k=1}^{r}(g_k^+(\mathbf{x}))^2 + \sum_{j=1}^{m}(h_j(\mathbf{x}))^2\right]$, where $g_k^+ = \max(0, g_k)$. Then total evaluation of an individual $\mathbf{x}$ is obtained as $\Phi(\mathbf{x}) = f(\mathbf{x}) + s\Phi_p(\mathbf{x})$, where s is a *penalty parameter*. By associating a penalty for each constraint violation, a constrained problem is transformed to an unconstrained problem such that we can deal with candidates that violate the constraints to generate potential solutions without considering the constraints.

The major issue in using the penalty function approach is assigning proper penalty parameters to the constraints: these parameters play the role of scaling penalties if the potential solution does not satisfy them. There are several methods depending on the choice of the penalty parameter [6]. The penalty function theorem gives a guideline to these methods on how the penalty parameter should be selected [7, 8].

Penalty Function Theorem: Let $\{s_t\}_{t=1}^{\infty}$ be a nonnegative, strictly increasing sequence tending to infinity. Define the function

$$L(s, \mathbf{x}) = f(\mathbf{x}) + \frac{s}{2}\left[\sum_{k=1}^{r}(g_k^+(\mathbf{x}))^2 + \sum_{j=1}^{m}(h_j(\mathbf{x}))^2\right]. \tag{2}$$

Let the minimizer of $L(s_t, \mathbf{x})$ be $\mathbf{x}_t$. Then any limit point of the sequence $\{\mathbf{x}_t\}_1^{\infty}$ is an optimal solution to (*P*). Furthermore, if $\mathbf{x}_t \rightarrow \bar{x}$ and $\bar{x}$ is a regular point, then $s_t g_k^+(\mathbf{x}_t) \rightarrow \lambda_k$ where $g_k^+(\mathbf{x}_t) = \max\{0, g_k(\mathbf{x}_t)\}$ and $s_t h_j(\mathbf{x}_t) \rightarrow \mu_j$, which are the Lagrange multipliers associated with g_k and h_j, respectively.

3 Two–Phase EP

EP used as the first phase of TPEP is implemented as in [3, 4]. For brevity, the procedures are omitted here. The fitness score for each solution $\mathbf{x}^i$ is evaluated in the light of an objective function $\Phi_1(\mathbf{x}^i)$ defined as:

$$\Phi_1(\mathbf{x}^i) = f(\mathbf{x}^i) + \frac{s_t}{2}\left[\sum_{k=1}^{r}(g_k^+(\mathbf{x}^i))^2 + \sum_{j=1}^{m}(h_j(\mathbf{x}^i))^2\right] \tag{3}$$

where $\{s_t\}_1^{\infty}$ is a sequence of penalty parameter defined in the penalty function theorem. EP procedure stops if the following condition is satisfied: For the best solution at generation t, $\mathbf{x}^1[t]$, and generation $t-1$, $\mathbf{x}^1[t-1]$, if $|\mathbf{x}_j^1[t] - \mathbf{x}_j^1[t-1]|/|\mathbf{x}_j^1[t]| \leq \rho = \rho_1$ for a sufficiently small positive value ρ_1 and all j for successive $N_g = N_{g1}$ generations, then the procedure stops.

After the first phase is halted, the EP formulation of the augmented Lagrangian method as a second phase is applied subsequently to the best evolved solution. In the light of the solution accuracy, the success rate and the computation time, the elitist EP with deterministic ranking strategy is considered for the EP formulation of the augmented Lagrangian method. In other words, the second phase initializes a population of N_{p2} trial solutions using the best solution found in the first phase, and employs the modified elitist EP, that is, the best solution always survives in the subsequent generation through the deterministic ranking strategy. Putting emphasis on violated constraints in the objective function whenever the best solution does not fulfill the constraints, the trial solutions are driven to the optimal point where all constraints are satisfied. The second phase of TPEP is implemented as follows:

1. A population of $N_{p2} \leq N_{p1}$ trial solutions is initialized. All vectors are initialized to be the same as the best evolved solution after the first phase. The values of $\boldsymbol{\sigma}^i$, $\forall i \in \{1, \cdots, N_{p2}\}$, are reset to $(\boldsymbol{\sigma}_2 + \boldsymbol{\sigma}_\epsilon)$, where $\boldsymbol{\sigma}_2$ is the strategy parameter of the best evolved solution after the first phase and $\boldsymbol{\sigma}_\epsilon$ is a positive constant vector, which can be considered as a strategy parameter resetting. Set $t \leftarrow 0$ and the Lagrange multipliers are initialized to zero.

2. The fitness score for each solution $\mathbf{x}^i$ is evaluated in the light of an objective function $\Phi_2(\mathbf{x}^i)$ defined as:

$$\Phi_2(\mathbf{x}^i) = \Phi_1(\mathbf{x}^i) + \sum_{k=1}^{r} \lambda_k g_k^+(\mathbf{x}^i) + \sum_{j=1}^{m} \mu_j h_j(\mathbf{x}^i) \tag{4}$$

where $g_k^+(\mathbf{x}) = \max(g_k(\mathbf{x}), -\frac{\lambda_k}{s_t})$.

3. From each of the N_{p2} parents $(\mathbf{x}^i, \boldsymbol{\sigma}^i)$, one offspring is generated by:

$$x_j^i[t] = x_j^i[t-1] + \sigma_j^i[t] \cdot N_j^i(0,1)$$
$$\sigma_j^i[t] = \sigma_j^i[t-1] \cdot \exp(\tau' \cdot N^i(0,1) + \tau \cdot N_j^i(0,1)).$$

4. The fitness score, $\Phi_2(\mathbf{x}^i)$, is determined.
5. Deterministic ranking over all the $2N_{p2}$ solutions is conducted.
6. The N_{p2} best solutions out of all the $2N_{p2}$ solutions based on the rank are selected to be the parents for the subsequent generation.
7. The Lagrange multipliers for the objective function Φ_2 are updated as follows:

$$\lambda_k[t] = \lambda_k[t-1] + \epsilon s_t g_k^+(\mathbf{x}^1[t-1]), \quad k = 1, \cdots, r$$
$$\mu_j[t] = \mu_j[t-1] + \epsilon s_t h_j(\mathbf{x}^1[t-1]), \quad j = 1, \cdots, m \tag{5}$$

where ϵ is a small positive constant. In this step, the penalty parameter s_t can be increased in an appropriate manner.

8. The algorithm proceeds $(t \leftarrow t+1)$ to step 3 unless the best solution does not change for a prespecified interval of generations.

By emphasizing the violated constraints using equation (5), the second phase eventually drives the solution to satisfy the constraints. The objective function of the form like equation (4) is called an augmented Lagrangian [7, 8]. In the definition of Φ_2, $g_k^+(\mathbf{x})$ is redefined as $\max(g_k(\mathbf{x}), -\frac{\lambda_k}{s_t})$, which is appropriate for the augmented Lagrangian [7, 8]. The stopping criteria for the second phase is the same as that of the first phase except for the values of $\rho = \rho_2 \leq \rho_1$ and $N_g = N_{g2} \geq N_{g1}$.

4 Evolian

Hybrid EP could only face with specific types of inequality constraints such as linear inequalities [3], while TPEP could consistently offer an exact feasible solution for other types of constraints [4, 5]. Generally speaking, TPEP was shown to be more applicable to the problems with various types of constraints. On the other hand, for problems having a moderate type of inequality constraints, the hybrid EP could give an accurate solution with less computation time and more convergence stability. Thus the proper application of either TPEP or hybrid EP, depending on the type of constraints, would offer better results with respect to the solution accuracy, convergence stability and the computation time.

While applying TPEP to multivariable heavily constrained optimization problems, it was noticed that when the penalty function is used in the objective function, any

optimization method would encounter some numerical difficulties if the problem is ill-conditioned, e.g., one constraint function is of different magnitude or changes more rapidly than the other constraint functions or the objective function and dominates the optimization process. If the constraints are normalized so that the magnitude of the constraints is unity, the conditioning of the problem is improved considerably. The normalization is recommended when formulating the problem or during the optimization process. It is reported, however, that the rate of change of the constraint is the more critical issue than the normalization, and this additional scaling may be done during the optimization process itself [11]. The most well-known concept is to scale each constraint so that the gradient of the constraint is of the same order of magnitude as the gradient of the original objective function. This ensures that the penalized objective function is not dominated by a single constraint and the result is less sensitive to the initial choice of penalty parameter. The scale factor c_i, $i = 1, \cdots, r+m$, of each constraint is usually chosen so that

$$\|\nabla f(\mathbf{x})\| = \begin{cases} c_k \|\nabla g_k(\mathbf{x})\|, & k = 1, \cdots, r \\ c_{r+j} \|\nabla h_j(\mathbf{x})\|, & j = 1, \cdots, m. \end{cases} \tag{6}$$

Another improvement is possible if another subsequent phase is employed after the second phase of TPEP depending on the dimension and the difficulty of the given problem. In some cases, due to the complexity of the problem, TPEP can not give meaningful solution in a reasonable time. Therefore, the application of the multi-phase concept is necessary to cope with various types of constrained problems, which can be regarded as a continuous extension of TPEP.

In this section, **Evolian** (**Evo**lutionary **O**ptimization based on **Lagrangian**) is proposed to integrate the concept of constraint scaling and multi-phase optimization. In each phase of Evolian, the typical EP is performed using augmented Lagrangian objective function with the penalty parameter fixed. If there is no improvement in the best objective function in one phase, another phase of Evolian is performed after scaling the constraints and then updating the Lagrange multipliers and penalty parameter. The procedure for Evolian is as follows:

1. **Initialization**
   (a) $t \longleftarrow 0$ (t indicates the phase number)
   (b) set Lagrange multipliers:
   $\lambda_k[t] \leftarrow 0, \quad k = 1, \cdots, r$
   $\mu_j[t] \leftarrow 0, \quad j = 1, \cdots, m$
   (c) set a penalty parameter: $s_t \leftarrow s_0$
   (d) set scaling factors: $c_i[t] \leftarrow 1, \quad i = 1, \cdots, r+m$
   (e) scale the objective function $f(\mathbf{x})$ if necessary
   (f) initialize the population for EP
2. **While** (**not** termination condition) **do**
   (a) execute EP for the augmented Lagrangian described by (4):

$$\Phi_2(\mathbf{x}^i) = \Phi_1(\mathbf{x}^i) + \sum_{k=1}^{r} \lambda_k g_k^+(\mathbf{x}^i) + \sum_{j=1}^{m} \mu_j h_j(\mathbf{x}^i), \quad i = 1, \cdots, N_p$$

where $g_k^+(\mathbf{x}) = \max(g_k(\mathbf{x}), -\frac{\lambda_k}{s_t})$.

(b) update constraint scaling factors by:

$$c_k = \sqrt{\frac{\sum_{i=1}^{n}(f(\mathbf{x}^1 + \delta\mathbf{e}_i) - f(\mathbf{x}^1))^2}{\sum_{i=1}^{n}(g_k^+(\mathbf{x}^1 + \delta\mathbf{e}_i) - g_k^+(\mathbf{x}^1))^2}}, \quad k = 1, \cdots, r \quad \text{and}$$

$$c_{r+j} = \sqrt{\frac{\sum_{i=1}^{n}(f(\mathbf{x}^1 + \delta\mathbf{e}_i) - f(\mathbf{x}^1))^2}{\sum_{i=1}^{n}(h_j(\mathbf{x}^1 + \delta\mathbf{e}_i) - h_j(\mathbf{x}^1))^2}}, \quad j = 1, \cdots, m \tag{7}$$

where δ is a small positive number and $\mathbf{e}_i$ denotes the ith standard basis vector, i.e., $\mathbf{e}_i = [0, \cdots, 1, \cdots, 0]$ with 1 in the ith slot.

(c) update Lagrange multipliers by:

$$\lambda_k[t+1] \leftarrow \lambda_k[t] + s_t g_k^+(\mathbf{x}^1[t]), \quad k = 1, \cdots, r$$
$$\mu_j[t+1] \leftarrow \mu_j[t] + s_t h_j(\mathbf{x}^1[t]), \quad j = 1, \cdots, m$$

(d) update the penalty parameter: $s_{t+1} \leftarrow \min(\gamma s_t, s_{max})$

(e) $t \longleftarrow t+1$

In the initialization step, all the Lagrange multipliers, the penalty parameter, and the scaling factors are set to 0, s_0, and 1, respectively. In addition, the original objective function $f(\mathbf{x})$ can be scaled by a constant factor if it is needed to avoid the ill-condition. The proper scaling of $f(\mathbf{x})$ can reduce the computation time greatly. In the initialization of EP, each individual is randomly generated according to the uniform random variable ranging over the search space $\mathcal{S}$. The jth parameter of $\mathbf{x}^i$, x_j^i is chosen in $[l_j, r_j]$.

In Step 2, the termination condition can be chosen among some known criteria such as (1) maximum generation number or (2) the termination condition used in the second phase of TPEP.

In Step 2(a), the typical EP is not reinitialized, and terminated if there is no improvement in the best solution. Step 2(a) stops if the following condition is satisfied: For the best solution at generation p, $\mathbf{x}^1[p]$, and generation $p-1$, $\mathbf{x}^1[p-1]$, if $\sqrt{\frac{1}{n}\sum_j(\mathbf{x}_j^1[p] - \mathbf{x}_j^1[p-1])^2} \leq \rho$ for a sufficiently small positive real number ρ for successive N_g generations, then the procedure stops. All variables x_j^i are forced to be in the search space during the evolution process. If the variable x_j^i is greater than r_j, x_j^i is set to r_j, and if x_j^i is less than l_j, x_j^i is set to l_j.

In Step 2(b), it should be noted that the inequality constraints are scaled based on the penalized constraints $g_k^+(\mathbf{x})$ not on the original constraints $g_k(\mathbf{x})$. The equation (7) is derived from equation (6) following the definition of partial derivative using the Euclidean norm for more precise approximation.

Lagrange multipliers are updated using the constraint functions evaluated at the best solution in Step 2(c).

In Step 2(d), the penalty parameter is increased by a constant rate $\gamma > 0$ according to the penalty function theorem until it reaches the prespecified maximum value s_{max}.

After increasing the time counter in Step 2(e), Step 2 is repeated until the termination condition is met.

It should be noted that the scale factors c_i's are multiplied to the constraint functions whenever the constraints are evaluated in all steps, and that the scaling of constraints and update of Lagrange multipliers are done around the best solution $\mathbf{x}^1$. It should also be noted that any kind of evolutionary optimization algorithm such as Genetic Algorithms, Evolution Strategies, Genocop series, etc. can be used instead of the typical EP.

5 Simulation Results

Five test cases were chosen from Michalewicz (1996). Some experimental results are also provided using six different constrained optimization techniques.

Problem #1:

Minimize $G_1(\mathbf{x}) = 5x_1 + 5x_2 + 5x_3 + 5x_4 - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$, subject to:

$$2x_1 + 2x_2 + x_{10} + x_{11} \leq 10, \quad 2x_1 + 2x_3 + x_{10} + x_{12} \leq 10,$$
$$2x_2 + 2x_3 + x_{11} + x_{12} \leq 10, \quad -8x_1 + x_{10} \leq 0,$$
$$-8x_2 + x_{11} \leq 0, \quad -8x_3 + x_{12} \leq 0,$$
$$-2x_4 - x_5 + x_{10} \leq 0, \quad -2x_6 - x_7 + x_{11} \leq 0,$$
$$-2x_8 - x_9 + x_{12} \leq 0, \quad 0 \leq x_i \leq 1, i = 1, \ldots, 9,$$
$$0 \leq x_i \leq 1, i = 10, 11, 12, \quad 0 \leq x_{13} \leq 1.$$

The global solution is $\mathbf{x}^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$, and $G_1(\mathbf{x}^*) = -15$.

Problem #2:

Minimize $G_2(\mathbf{x}) = x_1 + x_2 + x_3$, subject to:

$$1 - 0.0025(x_4 + x_6) \geq 0, \quad 1 - 0.0025(x_5 + x_7 - x_4) \geq 0,$$
$$1 - 0.01(x_8 - x_5) \geq 0, \quad x_1x_6 - 833.33252x_4 - 100x_1 + 83333.333 \geq 0,$$
$$x_2x_7 - 1250x_5 - x_2x_4 + 1250x_4 \geq 0, \quad x_3x_8 - 1250000 - x_3x_5 + 2500x_5 \geq 0,$$
$$100 \leq x_1 \leq 10000, \quad 1000 \leq x_i \leq 10000, i = 2, 3, \quad 10 \leq x_i \leq 1000, i = 4, \ldots, 8.$$

The global solution is

$$\mathbf{x}^* = (579.3167, 1359.943, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979),$$

and $G_2(\mathbf{x}^*) = 7049.330923$.

Problem #3:

Minimize $G_3(\mathbf{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$, subject to:

$$127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0, \quad 282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0,$$
$$196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0, \quad -4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0,$$
$$-10.0 \leq x_i \leq 10.0, i = 1, \cdots, 7.$$

The global solution is

$$\mathbf{x}^* = (2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227),$$

and $G_3(\mathbf{x}^*) = 680.6300573$.

Problem #4:

Minimize $G_4(\mathbf{x}) = e^{x_1x_2x_3x_4x_5}$, subject to:

$$x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 = 10, \quad x_2 x_3 - 5x_4 x_5 = 0, \quad x_1^3 + x_2^3 = -1,$$
$$-2.3 \leq x_i \leq 2.3, i = 1, 2, \quad -3.2 \leq x_i \leq 3.2, i = 3, 4, 5.$$

The global solution is

$$\mathbf{x}^* = (-1.717143, 1.595709, 1.827247, -0.7636413, -0.7636450),$$

and $G_4(\mathbf{x}^*) = 0.0539498478$.

Problem #5:

Minimize

$$G_5(\mathbf{x}) = x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45,$$

subject to:

$$105 - 4x_1 - 5x_2 + 3x_7 - 9x_8 \geq 0, \quad -10x_1 + 8x_2 + 17x_7 - 2x_8 \geq 0,$$
$$8x_1 - 2x_2 - 5x_9 + 2x_{10} + 12 \geq 0,$$
$$-3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4 + 120 \geq 0,$$
$$-5x_1^2 - 8x_2 - (x_3 - 6)^2 + 2x_4 + 40 \geq 0,$$
$$-x_1^2 - 2(x_2 - 2)^2 + 2x_1 x_2 - 14x_5 + 6x_6 \geq 0,$$
$$-0.5(x_1 - 8)^2 - 2(x_2 - 4)^2 - 3x_5^2 + x_6 + 30 \geq 0,$$
$$3x_1 - 6x_2 - 12(x_9 - 8)^2 + 7x_{10} + 12 \geq 0, \quad -10 \leq x_i \leq 10, i = 1, \ldots, 10.$$

The global solution is

$$\mathbf{x}^* = (2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927),$$

and $G_5(\mathbf{x}^*) = 24.3062091$.

In each problem, the constraints were transformed to the form of equation (1) before solving the problem, and 20 independent experiments were carried out. The specific parameter values for Evolian are shown in Table 1. Methods #1 - #6(f) and their results are described in Michalewicz (1996). Specifically Method #4 is Genocop II and Method #6(f) is a death penalty method used in ES (Evolution Strategies) and EP initialized by feasible solutions [12]. Method #7 is the Evolian with no constraint scaling operation. The results are summarized again in Table 2, where the best (b), median (m), and the worst (w) results are reported. c indicates the numbers of violated constraints at the median solution : the sequence of three numbers indicate the number of violations with violation amount between 0.1 and 10.0, between 0.1 and 1.0, and between 0.001 and 0.1, respectively. If at least one constraint was violated by more than 10.0, the solution was considered as `not meaningful'. Each method stops if it has reached its maximum generation number (5,000).

It is not an easy job to provide a complete analysis of each algorithm based on the five test problems. Considering Methods #1 to #7, Method #7 shows the best results for Problems #3, 4 and 5, and the reasonable results for Problems #1 and 2. For Problem #4, it was necessary to scale $f(\mathbf{x})$ by a factor of 100. To investigate the differences of the norm among objective and constraint functions, the norm of $f(\mathbf{x})$ and constraints $g_k^+(\mathbf{x})$ were compared for Problem #2 using Method #7. The active range of $f(\mathbf{x})$ is around

Table 1. The specific parameter values for Evolian.

Parameter	Value	Meaning
N_p	70	Population size
N_c	10	Num. of competing opponents in the ranking
σ_1	5.0	Initial perturbation
δ	0.01	Perturbation for partial derivative
ρ	10^{-3}	Error tolerance for EP
N_g	10	Generation tolerance for EP
s_0	1.0	Initial penalty parameter
s_{max}	100.0	Maximun penalty parameter
γ	3.0	Increasing rate of penalty parameter

Table 2. Experimental results for five problems. Methods #1 - #6(f) are described in Michalewicz (1996), of which results are referred here. Method #7 is the Evolian with no constraint scaling operation. The symbols `*' and `—' stand for `the method was not applied to this test case' and `the results were not meaningful', respectively.

Prob. #	Exact opt.		Method #1	Method #2	Method #3	Method #4	Method #5	Method #6	Method #6(f)	Method #7	Evolian
#1	-15.000	b	-15.002	-15.000	-15.000	-15.000	-15.000		-15.000	-15.000	-15.000
		m	-15.002	-15.000	-15.000	-15.000	-15.000	—	-14.999	-14.999	-14.999
		w	-15.001	-14.999	-14.998	-15.000	-14.999		-13.616	-13.000	-13.000
		c	0, 0, 4	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0		0, 0, 0	0, 0, 0	0, 0, 0
#2	7049.331	b	2282.723	3117.242	7485.667	7377.976	2101.367		7872.948	7667.240	7305.400
		m	2449.798	4213.497	8217.292	8206.151	2101.411		8559.423	11116.400	7476.080
		w	2756.679	6056.211	8752.412	9652.901	2101.551	—	8668.648	14665.100	13500.400
		c	0, 3, 0	0, 3, 0	0, 0, 0	0, 0, 0	1, 2, 0		0, 0, 0	0, 3, 1	0, 0, 0
#3	680.630	b	680.771	680.787	680.836	680.642	680.805	680.934	680.847	680.630	680.630
		m	681.262	681.111	681.175	680.718	682.682	681.771	681.826	680.633	680.636
		w	689.660	682.798	685.640	680.955	685.738	689.442	689.417	681.400	680.766
		c	0, 0, 1	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
#4	0.054	b	0.084	0.059		0.054	0.067			0.054	0.054
		m	0.955	0.812	*	0.064	0.091	*	*	0.054	0.054
		w	1.000	2.542		0.557	0.512			0.057	0.770
		c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0			0, 0, 0	0, 0, 0
#5	24.306	b	24.690	25.486		18.917	17.388		25.653	24.306	24.355
		m	29.258	26.905	—	24.418	22.932	—	27.116	25.125	24.923
		w	36.060	42.358		44.302	48.866		32.477	26.737	26.459
		c	0, 1, 1	0, 0, 0		1, 0, 0	0, 0, 0		0, 0, 0	0, 0, 0	0, 0, 0

$[2,000, 7,000]$, whereas $g_k^+(\mathbf{x}), k = 1, \cdots, 6$ are around $[0, 8]$. The constraints are of different magnitude from the objective function and some constraints, e.g., g_4^+, g_5^+ and g_6^+ change more rapidly than the objective function or the other constraints. This ill-condition makes Method #7 difficult to solve the problem. The application of Evolian to this problem results in the effective scaling of all constraints. Rapidly changing constraints g_4^+, g_5^+ and g_6^+ in Method #7 have become almost zero. The results of Evolian is shown in Table 2. Evolian seems to give best results for all problems among 9 different methods. For Problem #4, however, the selection of the sequence of penalty parameters (s_0, s_{max} and γ) was shown to have significant effect on the results, as can be seen from the fact that Evolian could not give better results than Method #7 for Problem #4.

6 Conclusions

In this paper, **Evolian** was proposed incorporating the concept of multi-phase optimization process and constraint scaling techniques to resolve the ill-condition. In each phase of Evolian, the typical EP was performed using augmented Lagrangian objective function with the penalty parameter fixed. If there was no improvement in the best objective function in one phase, another phase of Evolian was performed after scaling the constraints and then updating the Lagrange multipliers and penalty parameter.

Experimental results indicate that Evolian gives reasonable results for multivariable heavily constrained function optimization with smaller number of design parameters than the other methods.

References

1. Michalewicz, Z. (1995). A survey of constraint handling techniques in evolutionary computation methods. In McDonnell, J. R., Reynolds, R. G., and Fogel, D. B., editors, *Proc. of the Fourth Annual Conference on Evolutionary Programming*, pages 135–155, Cambridge, MA. MIT Press.
2. Michalewicz, Z. (1995). Genetic algorithms, numerical optimization, and constraints. In Eshelman, L. J., editor, *Proc. of the Sixth International Conference on Genetic Algorithms*, pages 151–158, Los Altos, CA. Morgan Kaufmann.
3. Myung, H. and Kim, J.-H. (1996). Hybrid evolutionary programming for heavily constrained problems. *BioSystems*, 38:29–43.
4. Kim, J.-H. and Myung, H. (1996). A two–phase evolutionary programming for general constrained optimization problem. In *Proc. of the Sixth Annual Conference on Evolutionary Programming*, Cambridge, MA. MIT Press. in press.
5. Myung, H. and Kim, J.-H. (1996). Constrained optimization using two–phase evolutionary programming. In *Proc. of the 1996 IEEE International Conference on Evolutionary Computation*, pages 262–267. IEEE Press.
6. Michalewicz, Z. and Attia, N. (1994). Evolutionary optimization of constrained problems. In Sebald, A. V. and Fogel, L. J., editors, *Proc. of Third Annual Conference on Evolutionary Programming*, pages 98–108, River Edge, NJ. World Scientific.
7. Luenberger, D. G. (1973). *Introduction to Linear and Nonliner Programming*. Addison-Wesley, Reading, MA.
8. Bertsekas, D. P. (1982). *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, New York.
9. Saravanan, N. and Fogel, D. B. (1994). Evolving neurocontrollers using evolutionary programming. In Michalewicz, Z., Kitano, H., Schaffer, D., Schwefel, H.-P., and Fogel, D. B., editors, *Proc. of the First IEEE Conf. on Evolutionary Computation*, pages 217–222, Orlando, Florida. IEEE press.
10. Bäck, T., Rudolph, G., and Schwefel, H.-P. (1993). Evolutionary programming and evolution strategies: Similarities and differences. In Fogel, D. B. and Atmar, W., editors, *Proc. of the Second Annual Conference on Evolutionary Programming*, pages 11–22, San Diego, CA. Evolutionary Programming Society.
11. Vanderplaats, G. N. (1984). *Numerical Optimization Techniques for Engineering Design*. McGraw-Hill Inc.
12. Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Berlin. Third, Revised and Extended Edition.