Coevolutionary Augmented Lagrangian Methods for Constrained Optimization

Min-Jea Tahk and Byung-Chan Sun

***Abstract*—This paper introduces a coevolutionary method developed for solving constrained optimization problems. This algorithm is based on the evolution of two populations with opposite objectives to solve saddle-point problems. The augmented Lagrangian approach is taken to transform a constrained optimization problem to a zero-sum game with the saddle-point solution. The populations of the parameter vector and the multiplier vector approximate the zero-sum game by a static matrix game, in which the fitness of individuals is determined according to the security strategy of each population group. Selection, recombination, and mutation are done by using the evolutionary mechanism of conventional evolutionary algorithms such as evolution strategies, evolutionary programming, and genetic algorithms. Four benchmark problems are solved to demonstrate that the proposed coevolutionary method provides consistent solutions with better numerical accuracy than other evolutionary methods.**

***Index Terms*—Constrained optimization, evolutionary computation, Lagrangian methods.**

I. Introduction

EVOLUTIONARY algorithms have been applied to various optimization problems with some promising results. Evolutionary algorithms have many advantages over conventional nonlinear programming techniques: the gradients of the cost function and constraints functions are not required, and the chance of being trapped by a local minimum is lower. Nonetheless, the existing evolutionary algorithms have not been fully developed to handle constrained optimization problems. Specifically, their numerical accuracy is still far below that of deterministic nonlinear programming techniques.

Michalewicz and Schoenauer [1] have presented a comparison study of existing evolutionary algorithms developed for solving constrained optimization problems. According to this study, the existing algorithms are grouped as follows: 1) methods based on preserving feasibility of solutions, 2) methods based on penalty functions, 3) methods based on the superiority of feasible solutions over infeasible solutions, and 4) other hybrid methods. As in the deterministic nonlinear programming techniques, the key issue is how to handle various types of constraints. As shown in [1], the common drawback of these methods is that each method works well for some problems, but does not for other problems. Recently, Kim and Myung [2] have proposed a two-phase evolutionary programming using the augmented Lagrangian function in the second phase. In this method, the Lagrange multiplier is updated using the first-order update scheme applied frequently in the deterministic augmented Lagrangian methods. Although this method exhibits good convergence characteristics, it has been tested only for small-scale problems. Note that all of the existing evolutionary algorithms described above are based on evolution of a single group. One exception is the work of Paredis [3], which suggests the use of coevolution for constraint satisfaction. Although this work does not directly address constrained optimization problems, it is the first attempt known to the authors to use the concept of coevolution to handle constraints.

The coevolution method proposed in this paper is based on a game point of view, which is very different from any of the existing evolutionary methods. Originally, it was devised to solve minimax problems arising in robust control design [4]–[6]. If a minimax problem has a saddle-point solution, then it can be treated as a zero-sum game. By discretizing the strategy variables of the zero-sum game, we obtain a static matrix game from which the saddle point can be found approximately. At this point, we apply the concept of evolution to generate the next static matrix game that has a denser population around the saddle point than the previous matrix game. In the coevolution approach, the population of each group represents the set of strategies of each player involved in the game.

The same coevolution method can be used to solve constrained optimization problems. Consider the dual formulation of a constrained optimization problem for which x is the parameter vector to be optimized and λ is the Lagrange multiplier. It is well known that the saddle point (x^*, λ^*) of the Lagrangian gives the solution x^* of the constrained problem if the problem is convex. For nonconvex problems, we augment the Lagrangian to convexify the cost function in the small neighborhood of the solution. In fact, for any practical constrained optimization problems, it is possible to find a suitable augmented Lagrangian that possesses the saddle-point property around (x^*, λ^*). Hence, a constrained optimization problem can be reformulated as a zero-sum game problem for which the Lagrange multiplier vector λ is the maximizing player and the parameter vector x is the minimizing one.

The saddle-point of the augmented Lagrangian can then be found by the proposed coevolution method, which will be referred to as the coevolutionary augmented Lagrangian method (CEALM). Contrary to classical augmented Lagrangian

Manuscript received February 2, 1999; revised May 20, 1999. This work was supported by the Korea Science Foundation. This work was supported by Automatic Control Research Center (ACRC), Seoul National University, and Agency for Defense Development, Taejon, Korea.

M. J. Tahk is with the Department of Aerospace Engineering, Korea Advanced Institute of Science and Technology (KAIST), Taejon 305-701, Korea (e-mail: mjtahk@fdcl.kaist.ac.kr).

B.-C. Sun is with Korea Aerospace Research Institute (KARI), P.O. Box 113, Taejon 305-600, Korea.

Publisher Item Identifier S 1089-778X(00)02630-8.

methods that search for the minimum of the augmented Lagrangian with some update schemes for the multiplier, the CEALM relies on coevolution of the population of the parameter vector and that of the multiplier. The evolutionary mechanism of each group should include procedures for fitness evaluation, selection, recombination, and mutation. While fitness evaluation is based on a security strategy, which is unique to saddle-point problems, other procedures can be implemented by using the features of the existing evolutionary methods such as evolution strategies, evolutionary programming, and genetic algorithms. The CEALM uses an evolution strategy for the sake of convergence speed.

Although the initial study on the CEALM [7] has found its usefulness for solving various constrained optimization problems, the main drawbacks have been inconsistent solutions and poor numerical accuracy. While the self-adaptation feature of evolution strategies and evolutionary programming works satisfactorily with well-posed problems, it may reduce mutation variances too quickly, and stop the evolution process before the solution area is found. This premature freezing phenomenon is identified as the main culprit causing the inconsistency. Hence, an annealing scheme is employed as a remedy. Instead of strictly controlling the mutation variances (or temperature), the proposed annealing scheme only imposes the lower bound of the mutation variances to maintain the adaptation capability of the self-adaption procedure as much as possible. This scheme is found to be very effective in reducing the risk of being trapped by a local solution, as well as avoiding premature freezing. The CEALM improved by the annealing scheme is now capable of solving tightly constrained problems without difficulty. Also, the CEALM can be further improved by the addition of rotation angles as mutation parameters to produce more consistent solutions, as will be shown later.

This paper is organized as follows. The augmented Lagrangian formulation for constrained optimization is reviewed briefly, and the concept of coevolution is described next. The annealing scheme to improve convergence characteristics is then described. To demonstrate the capability of the proposed method, four benchmark problems of constrained optimization treated in [1] are solved, and the numerical results are presented. Concluding remarks are also provided.

II. Augmented Lagrangian Methods

This section summarizes the key ideas of the augmented Lagrangian methods for readers who are not familiar with this classical nonlinear programming method. Consider a general constrained optimization problem

$$\min_x f(x), \qquad x \in R^n \tag{1}$$

subject to

$$\begin{aligned} g_i(x) &\leq 0, & i &= 1, \cdots, m \qquad (2)\\ h_i(x) &= 0, & i &= 1, \cdots, l \qquad (3)\\ L_i \leq x_i &\leq U_i, & i &= 1, \cdots, n. \qquad (4) \end{aligned}$$

Let $S \subset R^n$ be the search space specified by (4). For this primal problem, we have the Lagrangian dual problem

$$\max_{\mu, \lambda} \theta(\mu, \lambda) \tag{5}$$

subject to

$$\mu_i \geq 0, \qquad i = 1, \cdots, m \tag{6}$$

where

$$\theta(\mu, \lambda) = \min_x \left\{ f(x) + \mu^T g(x) + \lambda^T h(x) \right\}, \qquad x \in S. \tag{7}$$

Here, μ is an $m \times 1$ multiplier, and λ is an $l \times 1$ multiplier for the inequality and equality constraints, respectively.

If the primal problem is convex over S (f and g are convex and h is affine over S), the strong duality theorem [8] states that

$$\begin{aligned} &\min\{f(x)\colon g(x) \leq 0,\, h(x) = 0,\, x \in S\} \\ &\quad = \max\{\theta(\mu, \lambda)\colon \mu \geq 0\}. \end{aligned} \tag{8}$$

In fact, the solution x^* of the primal problem along with (μ^*, λ^*) of the dual problem satisfies the Kuhn–Tucker condition, and corresponds to the saddle point of the Lagrangian function defined by

$$L(x, \mu, \lambda) = f(x) + \mu^T g(x) + \lambda^T h(x). \tag{9}$$

That is,

$$L(x^*, \mu, \lambda) \leq L(x^*, \mu^*, \lambda^*) \leq L(x, \mu^*, \lambda^*) \tag{10}$$

for any $x \in S$ and $\mu \geq 0$. Note that the right side of (10) implies that x^* is the unconstrained minimum of $L(x, \mu^*, \lambda^*)$. If (μ^*, λ^*) is known, then x^* can be searched in S without considering the constraints.

Example 1: Consider a convex problem given by $f(x) = x_1^2 + x_2^2$, and $h(x) = x_1 + x_2 - 1$. The solution of this problem is found as $(x_1^*, x_2^*) = (1/2, 1/2)$, $\lambda^* = -1$, and $f(x^*) = 1/2$. Note that $L(x, \lambda^*)$ is given as

$$L(x, \lambda^*) = \left(x_1 - \tfrac{1}{2}\right)^2 + \left(x_2 - \tfrac{1}{2}\right)^2 + \tfrac{1}{2}$$

and (x_1^*, x_2^*) is indeed the unconstrained minimum of $L(x, \lambda^*)$.

Example 2: Consider a nonconvex problem given by $f(x) = 2x_1^2 - x_2^2$, $h(x) = x_1 + x_2 - 1$. Again, the solution is easily found as $(x_1^*, x_2^*) = (-1, 2)$, $\lambda^* = 4$, and $f(x^*) = -2$. Then, we obtain

$$L(x, \lambda^*) = 2(x_1 + 1)^2 - (x_2 - 2)^2 - 2$$

for which (x_1^*, x_2^*) is an extremum, but not an unconstrained minimum. For a given λ,

$$\theta(\lambda) = \min_x \left\{ 2x_1^2 - x_2^2 + \lambda(x_1 + x2 - 1) \right\} = -\infty.$$

Hence, there is no unique solution to the dual problem.

As shown in Example 2, we have a duality gap for nonconvex problems; the solution of the dual problem does not coincide with that of the primal problem. In this case, we have

$$\begin{aligned}\min\{f(x)\colon g(x)\leq 0,\, h(x)=0\}\\ >\max\{\theta(\mu,\,\lambda)\colon \mu\geq 0\},\qquad x\in S\end{aligned}\tag{11}$$

and there does not exist a saddle point satisfying (10). Moreover, an unconstrained search for x^* is not possible even if $(\mu^*,\,\lambda^*)$ is known. The augmented Lagrangian methods avoid this difficulty by convexifying f with quadratic penalty terms associated with the constraints; the augmented Lagrangian is usually defined as

$$\begin{aligned}&L_A(x,\,\mu,\,\lambda,\rho)\\ &=f(x)+\sum_{k=1}^{m}p_k(x,\,\mu_k,\,\rho)+\lambda^T h(x)+\rho\sum_{k=1}^{l}h_k^2(x)\end{aligned}\tag{12}$$

where the term p_k for the kth inequality constraint is given by

$$p_k(x,\,\mu_k,\,\rho)=\begin{cases}\mu_i g_i(x)+\rho g_i^2(x), & \text{if } g_i(x)\geq -\dfrac{\mu_i}{2\rho}\\ -\dfrac{\mu_i^2}{4\rho}, & \text{if } g_i(x)< -\dfrac{\mu_i}{2\rho}\end{cases}\tag{13}$$

(refer to [8] and [9] for the derivation). It can be easily shown that the Kuhn–Tucker solution $(x^*,\,\mu^*,\,\lambda^*)$ of the primal problem is identical to that of the augmented problem. It is also well known that, if the Kuhn–Tucker solution is a strong local minimum, then there exists $\overline{\rho}$ such that x^* is a strong local minimum of $L_A(x,\,\mu^*,\,\lambda^*,\,\rho)$ for all $\rho\geq\overline{\rho}$; the Hessian of L_A with respect to x near the solution $(x^*,\,\mu^*,\,\lambda^*)$ can be made positive definite. Therefore, x^* can be obtained by an unconstrained search if $(\mu^*,\,\lambda^*)$ is known and the search starts from a point close to x^*.

Example 3: Consider the nonconvex problem of Example 2 again. Now, the Lagrangian function is augmented by the penalty function $\rho(x_1+x_2-1)^2$ as

$$\begin{aligned}L_A(x,\,\lambda,\,\rho)=&\,2x_1^2-x_2^2+\lambda(x_1+x_2-1)\\ &+\rho(x_1+x_2-1)^2.\end{aligned}$$

The solution of the augmented problem is identical to the original problem, but we see that

$$L_A(x,\,\lambda^*,\,\rho)=(2x_1+x_2)^2+(\rho-2)(x_1+x2-1)^2-2.$$

Hence, $(x_1^*,\,x_2^*)=(-1,\,2)$ is the unconstrained minimum of $L_A(x,\,\lambda^*)$ if $\rho>2$. Also, for any λ, we can show that

$$\begin{aligned}\theta_A(\lambda,\,\rho)&=\min_x L_A(x,\,\lambda,\,\rho)\\ &=(2x_1+x_2)^2+(\rho-2)(x_1+x2-1)^2\\ &\quad+(\lambda-4)(x_1+x2-1)-2\\ &=-\frac{(\lambda-4)^2}{4}-2\end{aligned}$$

if $\rho>2$. The solution to the dual problem is then given as $\lambda^*=4$ and $\theta_A(\lambda^*)=-2$. This solution is identical to that of the primal problem, and the equality of (8) holds.

Note that the convexity obtained by augmenting the Lagrangian is only a local property. The sufficient condition for $(x^*,\,\mu^*,\,\lambda^*)$ to be the saddle point of the augmented Lagrangian is that the augmented problem is convex over S. For many practical problems, however, the convexity over the entire search space is neither possible nor essential. Consider a local minimum of the primal problem, denoted as $\hat{x}$. Suppose that g and h are differentiable at $x=\hat{x}$. Let $N_\delta(\hat{x})$ be the neighborhood of $\hat{x}$ such that $\|x-\hat{x}\|<\delta$. Then, there always exists $\delta>0$ such that $\hat{x}$ is the global minimum of the primal problem with a search space restricted to $N_\delta(\hat{x})$. Hence, for any local minimum, we can construct an augmented Lagrangian for which $(\hat{x},\hat{\mu},\hat{\lambda})$ is the saddle point. (For convenience, this type of saddle point is referred to as a local saddle point, and the saddle point over the whole search space S as the global saddle point.)

The augmented Lagrangian methods have two distinct advantages over the other nonlinear programming techniques. First, the exact solution can be achieved by the augmented Lagrangian methods, while it is not true for the simple penalty function methods. Second, the search direction is not restricted by the constraints since x^* is the unconstrained minimum of $L_A(x,\,\mu^*,\,\lambda^*,\,\rho)$ in the neighborhood of x^*. The main issue of the deterministic augmented Lagrangian methods is how to update the Lagrange multiplier so that it converges to $(\mu^*,\,\lambda^*)$. The recent evolutionary method proposed in [2] also depends on efficient multiplier update schemes. Contrary to these approaches, the coevolution method proposed here takes a genuinely evolutionary approach, relying on the evolution of (μ,λ) as well as that of x to achieve the saddle point (x^*,μ^*,λ^*).

III. Matrix Games and Coevolution

The coevolution algorithm used in this paper is concerned with the evolution of two populations with opposite objectives to solve a static zero-sum game. Each population group has its own evolutionary process, and tries to generate the best individual that gives the best payoff for the group to which it belongs. Each individual represents a distinct strategy (or option) for its group, and the populations of the two groups define a matrix game of finite dimension, which is an approximation of the zero-sum game. The evolutionary process gradually improves the matrix game in the sense that each group has a denser population around the saddle-point solution. Fitness evaluation is based on the security strategy of each group, which is the basic strategy of matrix zero-sum games. This coevolution algorithm is directly applicable to solve a constrained optimization problem since the augmented Lagrangian formulation gives a saddle-point problem. The main ideas of matrix games and coevolution are summarized in the following.

Consider a static zero-sum game G for which a payoff function $F=F(u,\,v)$ is to be minimized by $u\in U$ and maximized by $v\in V$. We assume that there exists a pair $(u^*\in U,\,v^*\in V)$ such that

$$\min_{u\in U}F(u,\,v^*)=\max_{v\in V}F(u^*,\,v)=F(u^*,\,v^*)=F^*.\tag{14}$$

Then, the pair (u^*, v^*), which is called the saddle-point solution, satisfies the inequalities

$$F(u^*, v) \le F(u^*, v^*) \le F(u, v^*), \qquad \forall u \in U, \forall v \in V. \tag{15}$$

Now, suppose that u and v have a finite number of options given by

$$U_M = \{u_1, u_2, \cdots, u_{N_u}\} \tag{16}$$
$$V_M = \{v_1, v_2, \cdots, v_{N_v}\} \tag{17}$$

where N_u and N_v denote the numbers of options of u and v, respectively. Then, the triplet (F, U_M, V_M) defines a matrix game G_M.

The matrix game can be interpreted in the context of coevolution as follows: Let X_i and Y_j denote individuals representing the options u_i and v_j, respectively. Population X is then defined as the collection of all X_i's, and population Y the collection of all Y_j's. The score of the match (X_i, Y_j) is defined as the value of $F(u_i, v_j)$. Then, solving G_M corresponds to finding the best individual of each population group. Since the option of the opposite player is unknown, the player u should consider the worst outcome for each option. That is, the fitness of X_i is determined from the largest payoff among the matches (X_i, Y_1), (X_i, Y_2), $\cdots$, (X_i, Y_{N_v}). Similarly, the smallest payoff of Y_j gives its fitness value. Hence, we define

$$\text{fitness of } X_i = \max_j F(X_i, Y_j) \tag{18}$$
$$\text{fitness of } Y_j = -\min_i F(X_i, Y_j). \tag{19}$$

In fact, the way to determine the fitness of each individual is equivalent to the security strategy of matrix games [10]. A security strategy of u, denoted as u^s, is the strategy (or option) that gives the minimum worst cast payoff for u, $\overline{F}(G_M)$, which is also called the security level for player u's losses. The security strategy satisfies the inequality

$$\begin{aligned}\overline{F}(G_M) &\equiv \max_j F(u^s, v_j)\\ &\le \max_j F(u_i, v_j), \qquad i = 1, \cdots, N_u.\end{aligned} \tag{20}$$

Similarly, a security strategy of v, denoted as v^s, and the security level for player v's gains $\underline{F}(G_M)$ are determined from

$$\begin{aligned}\underline{F}(G_M) &\equiv \min_i F(u_i, v^s)\\ &\ge \min_i F(u_i, v_j), \qquad i = j, \cdots, N_v.\end{aligned} \tag{21}$$

Note that the security levels always satisfy

$$\underline{F}(G_M) \le \overline{F}(G_M). \tag{22}$$

If the equality of (22) holds, (u^s, v^s) corresponds to the saddle-point solution of G_M. Since G_M is only an approximation of G, G_M may not have a solution at all, even if G does. Or the solution of G_M can be quite different from (u^*, v^*), the saddle-point solution of G. However, as X and Y are made denser around (u^*, v^*), the security strategy (u^s, v^s) gives a better approximation of (u^*, v^*). The convergence of (u^s, v^s) to (u^*, v^*) depends on the procedures of parent selection and offspring generation. If the offspring population is always produced denser around the saddle point than the previous generation, a coevolution process with a security strategy will surely converge to the saddle point.

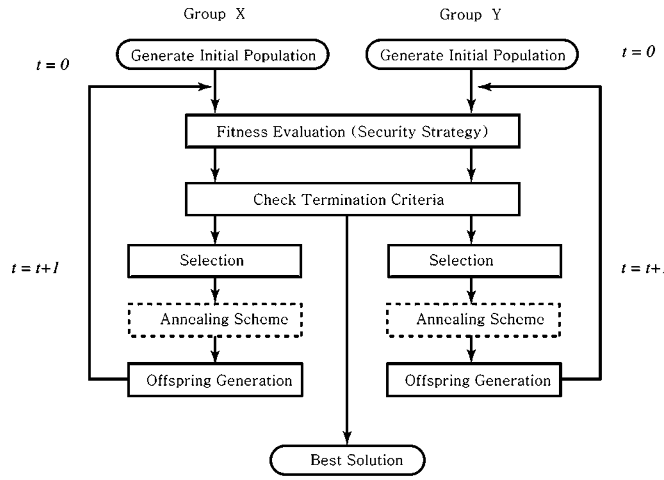


Fig. 1. Flowchart of coevolution for saddle-point problems.

Fig. 1 shows the flowchart of the proposed coevolutionary algorithm. Except that a fitness evaluation procedure based on security strategy is used, a coevolutionary process is basically a parallel evolution of two population groups. Hence, a coevolution mechanism can be easily implemented by using the techniques of various single-group evolutionary algorithms: for example, evolution strategy, evolutionary programming, or genetic algorithms. For the numerical examples of this paper, the algorithm of evolution strategy [11] is employed for selection, recombination, and mutation of each population group. Specifically, the (μ, λ) method is used for selection, discrete recombination for parameter values, and intermediate recombination for mutation variances. The self-adaptation feature of evolution strategy is also adopted. The use of rotation angles provides a sophisticated mutation process for better convergence characteristics, but can be omitted for reduction in computation time. In addition to these features of evolution strategy, an annealing scheme is included to prevent the self-adaption scheme from cooling down the temperature (or mutation variance) too quickly before the population arrives at the solution area.

The CEALM proposed in this paper is a straightforward application of coevolution to constrained optimization. Using the augmented Lagrangian formulation, a constrained optimization problem can be transformed to a zero-sum game between the parameter vector x and the multiplier vector (μ, λ). If the primal problem has a solution, then it is always possible to formulate an augmented Lagrangian with at least one local saddle point. If the global saddle point is the only local saddle point, then it can be achieved by a properly designed coevolution algorithm in most cases. In the case of multiple local saddle points, the coevolution process may converge to one of them, giving only a local minimum of the primal problem.

Given $X = \{X_i\}$ and $Y = \{Y_j\}$, we need to evaluate $L_A(X_i, Y_j)$ for all i and j. Thus, the number of evaluations of L_A is $N_u \times N_v$. However, we need to evaluate $f(x)$, $g(x)$, and $h(x)$ only for N_u times, once for each X_i. Once $f(x)$, $g(x)$,

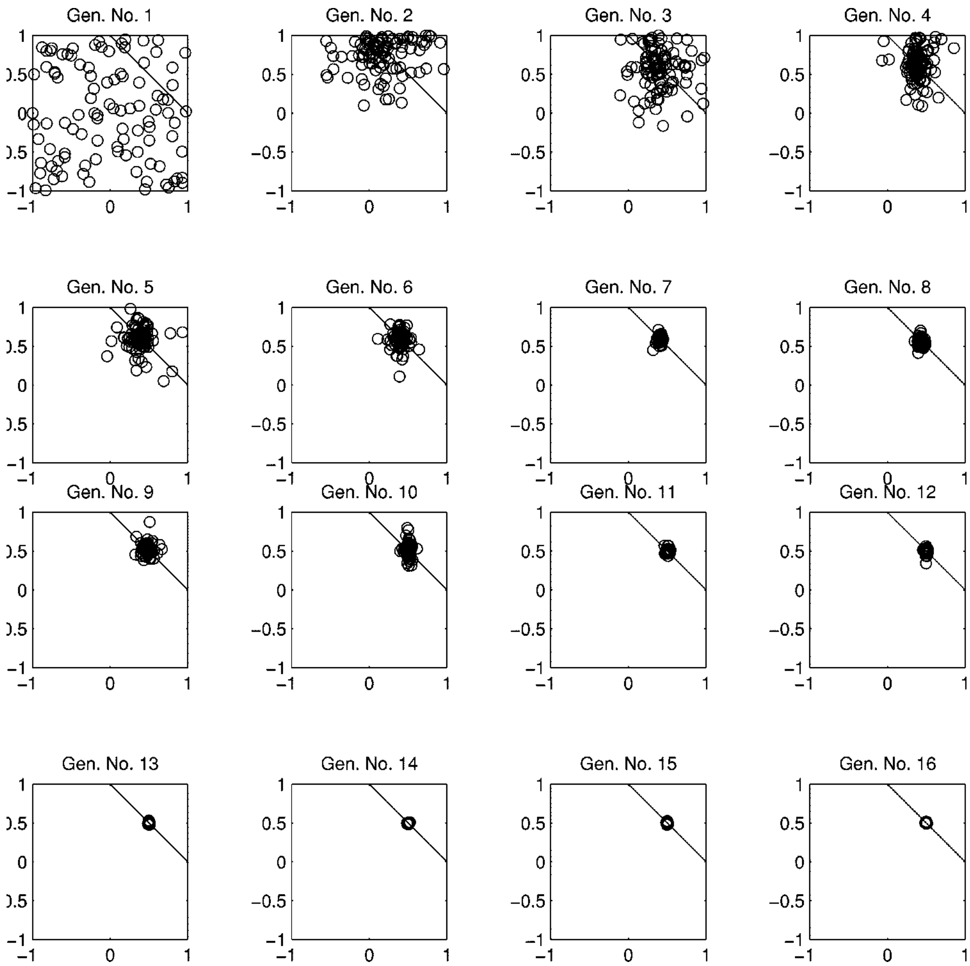


Fig. 2. Evolution of parameters for Example 1 (horizontal axis $= x_1$, vertical axis $= x_2$).

and $h(x)$ are evaluated for an X_i, the value of $L_A(X_i, Y_j)$ for any Y_j requires a simple calculation of $\mu^T g(x) + \lambda^T h(x)$ and the penalty terms. Note that the evaluation of the cost and constraints takes most of computation time for practical optimization problems. Therefore, the computational burden of the coevolutionary method is comparable to that of the other methods based on evolution of a single group.

The example problems treated in the previous section are solved by using the CEALM. For convenience, the search space is restricted to $-1 \leq x_1 \leq 1$ and $-1 \leq x_2 \leq 1$ for Example 1, and $-5 \leq x_1 \leq 5$ and $-5 \leq x_2 \leq 5$ for Examples 2 and 3. The population sizes are chosen as $\lambda = 100$ and $\mu = 20$. Fig. 2 shows the evolution of the offspring population in the (x_1, x_2) plane for Example 1. A fast evolution is well observed in this figure. The offspring evolution for Example 2 is shown in Fig. 3. Since the problem is not convex, the offspring population keeps wandering around the search space without any sign of convergence. Augmentation of the Lagrangian function transforms the problem to a convex one, as discussed in Example 3, resulting in a satisfactory convergence, as shown in Fig. 4.

IV. Self-Adaptation and Annealing

The algorithm of the CEALM is based on an evolution strategy for selection, recombination, and mutation of the population groups X and Y. Although evolutionary programming and genetic algorithms are also applicable to coevolution, an evolution strategy is found most effective for the CEALM studied here. A feature of evolution strategy is self-adaptation, which automatically adjusts the mutation size of each individual. According to [11], the mutation process of evolution strategy is described as

$$\sigma_i' = \sigma_i \cdot \exp(\tau_o \cdot N(0, 1) + \tau \cdot N_i(0, 1)), \quad i = 1, \cdots, m \tag{23}$$

$$\alpha_{ij}' = \alpha_{ij} + \beta \cdot N_{ij}(0, 1), \qquad i, j = 1, \cdots, m, \quad i > j \tag{24}$$

$$\vec{x}' = \vec{x} + \vec{N}(\vec{0}, C(\vec{\sigma}', \vec{\alpha}')) \tag{25}$$

where $\vec{x}$ is the parameter vector to be optimized, m is the size of $\vec{x}$, and $N(0, 1)$ is the random number with a Gaussian distribution of zero mean and unity variance. The covariance matrix C given by the diagonal elements σ_i and rotation angles α_j determines the statistics of the mutation size of $\vec{x}$. The rotation angles are used to generate correlated mutations among the parameters x_i.

Although the self-adaptation process automatically adjusts the covariance matrix, it is often observed that the mutation variances converge to zero before a local solution is achieved. If this happens to the whole population, then the evolution process is

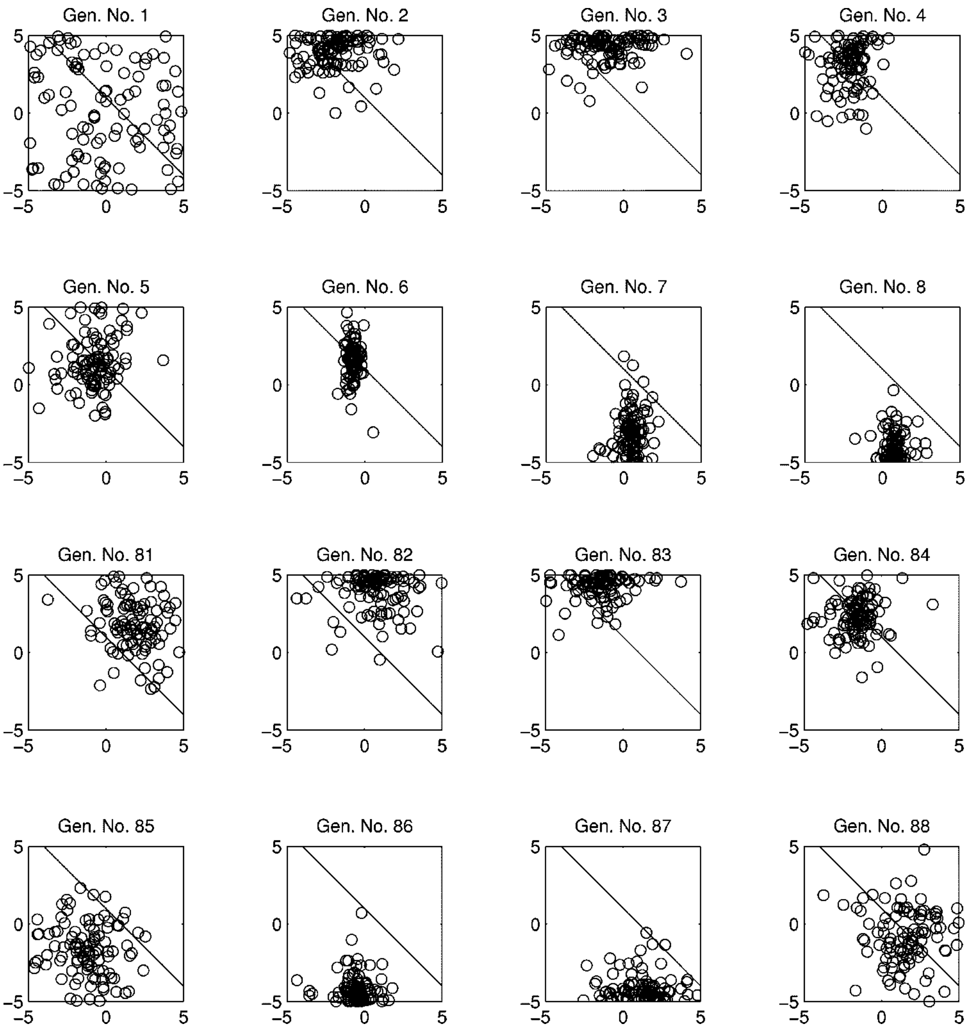


Fig. 3. Evolution of parameters for Example 2 (horizontal axis $= x_1$, vertical axis $= x_2$).

completely frozen. This phenomenon, referred to as *premature freezing* here, takes place whenever the fitness of the population is not easily improved. Ref. [11] recommends that σ_i be kept larger than a certain small number to keep the evolutionary process from freezing. However, this remedy usually does not work if premature freezing occurs at a point far from the solution. For example, consider a cost function with a geometry of a deep valley. If the population happens to wander inside the valley, the individuals located in the middle of the valley are more likely to survive, reducing their mutation variances very quickly. By using the rotation angles α_{ij}, the possibility of premature freezing is reduced if the population evolves to find the best direction to reach the solution. Numerical experiments show that the use of rotation angles produces more consistent results, but does not guarantee that premature freezing can always be avoided. Moreover, implementing this feature requires $n(n+1)/2$ coordinate transformations, each of which involves the computation of $\cos\alpha_{ij}$ and $\sin\alpha_{ij}$ to generate correlated mutations. For large-dimensional problems, this significantly increases the computational burden. For the benchmark problems of the next section, the computation time is found to increase by 25%–30% when the mutation process includes rotation angles.

It is well known that a simulated annealing procedure reaches the global solution if the temperature (or mutation variance) is reduced slowly. In terms of simulated annealing, the self-adaptation scheme of evolution strategy is a type of adaptive temperature control. Now, we suggest a simple annealing feature to alleviate the problem of premature freezing. In addition to self-adaptation, the lower temperature bound is programmed as a function of the time (or generation) to avoid premature freezing. Let t denote the current generation number, and let t_d denote the time interval for the lower bound of σ_i, denoted as $\underline{\sigma}_i$, to be reduced by a factor of 10, i.e.,

$$\underline{\sigma}_i(t+t_d) = 0.1\underline{\sigma}_i(t). \tag{26}$$

Given t_d, $\underline{\sigma}_i$ is reduced at each generation by

$$\underline{\sigma}_i(t+1) = \beta\underline{\sigma}_i(t) \tag{27}$$

where $\beta = 10^{-(1/t_d)}$. In our scheme, the proposed annealing feature is active only when self-adaptation is done too fast. Since only the lower temperature bound is imposed, the self-adaptation scheme works unless the temperature drops below the programmed bound.

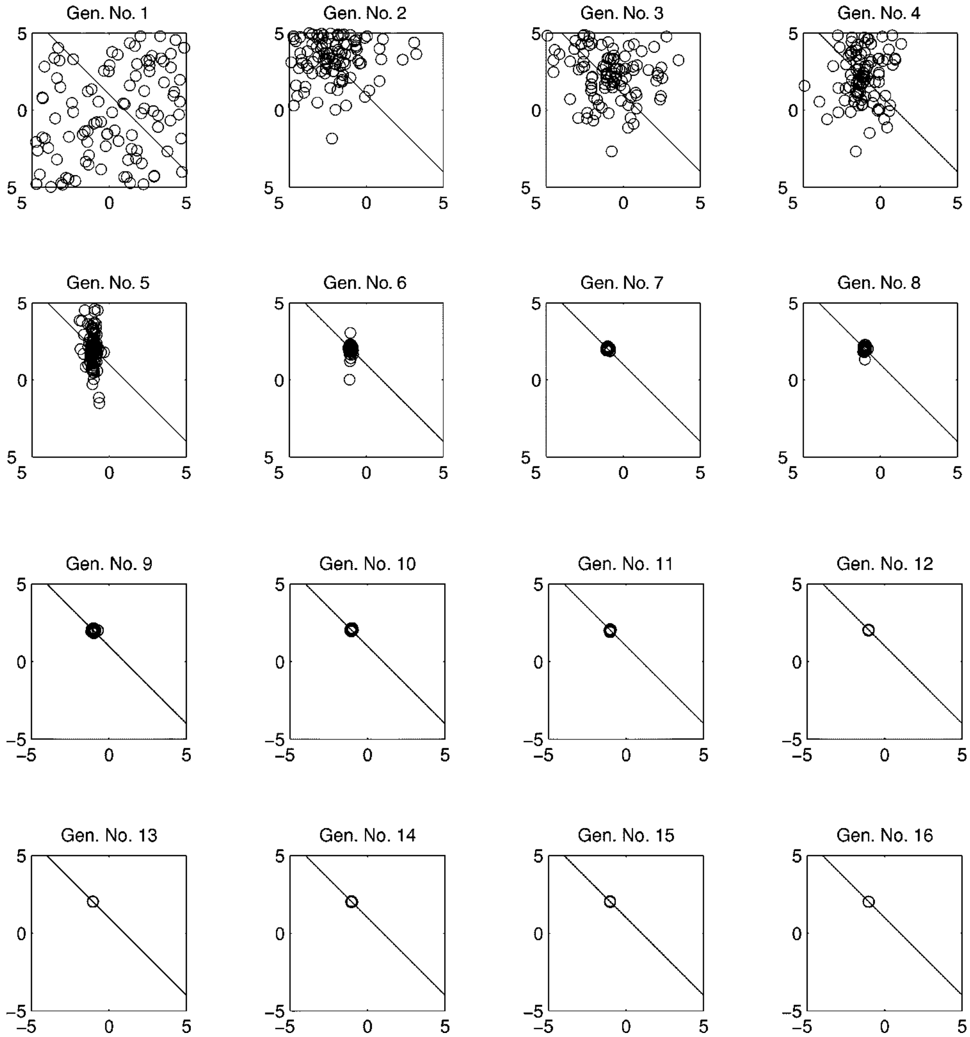


Fig. 4. Evolution of parameters for Example 3 (horizontal axis $= x_1$, vertical axis $= x_2$).

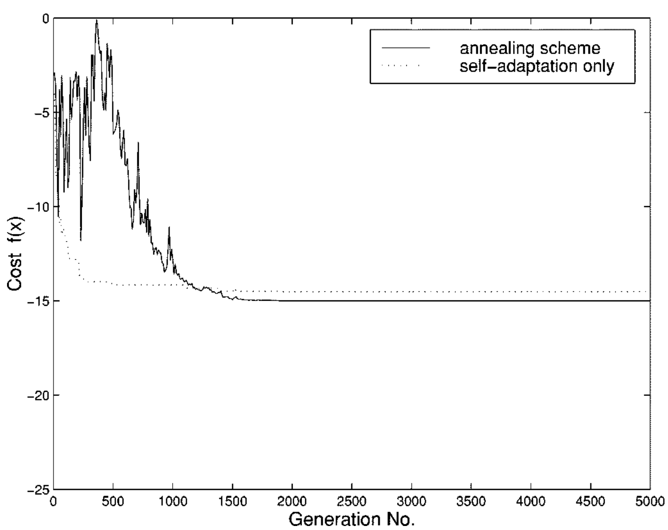


Fig. 5. Best individual's cost value (Problem G1).

Numerical experiments show that the proposed annealing scheme is quite effective in avoiding local solutions as well as premature freezing. A proper t_d can be found as follows. First, the coevolution algorithm is executed several times without the annealing feature. If the solution is not consistent, then premature freezing or local solutions are suspected. In that case, the proposed annealing scheme with a small t_d is taken. The value of t_d may have to be increased if consistent solutions are not produced. Using a small t_d at first is recommended since the algorithm takes more time to obtain convergence with a larger t_d.

V. BENCHMARK PROBLEMS

The following four benchmark problems are taken from [1], where several evolutionary algorithms developed for constrained optimization are compared. To evaluate the efficiency of the coevolution approach for nonlinear programming problems, these benchmark problems are solved by using the CEALM described in the previous sections. A (μ, λ)-evolution strategy is employed for recombination, mutation, and selection with population sizes of $\mu = 8$ and $\lambda = 40$. Results of a parametric study on the population sizes will be discussed later. Selection of the penalty weight ρ may need some numerical experiments since the convexity property is totally unknown *a priori*. We need to increase the value of ρ if there is no sign of convergence. The choice of ρ is not crucial for convergence as

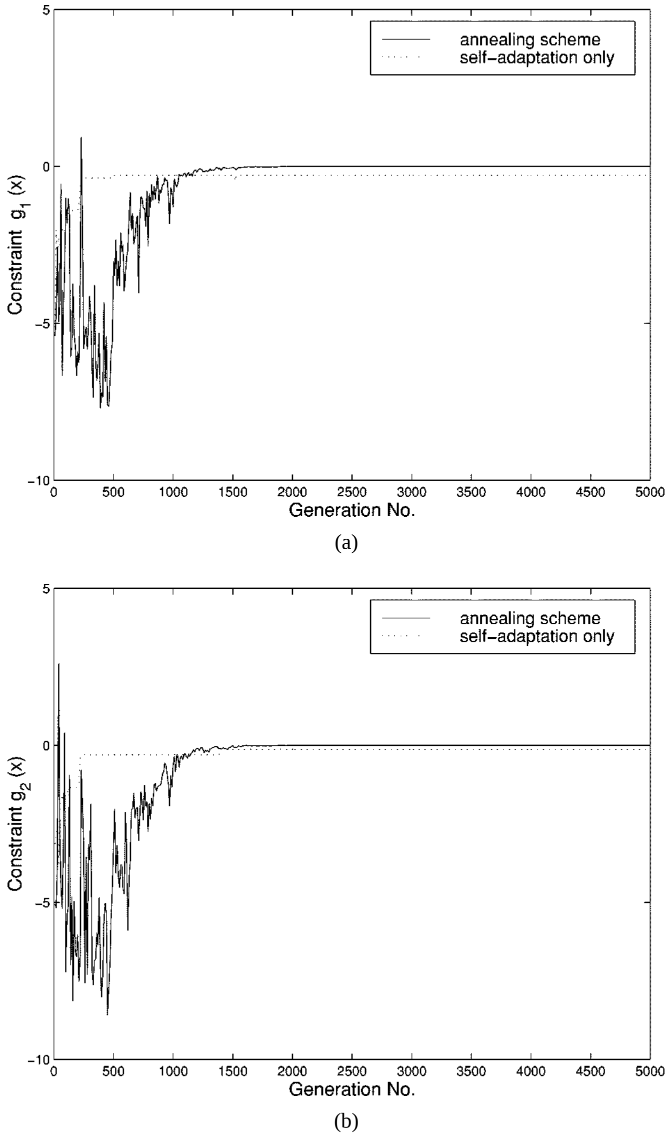


Fig. 6. Best individual's constraint variances (Problem G1).

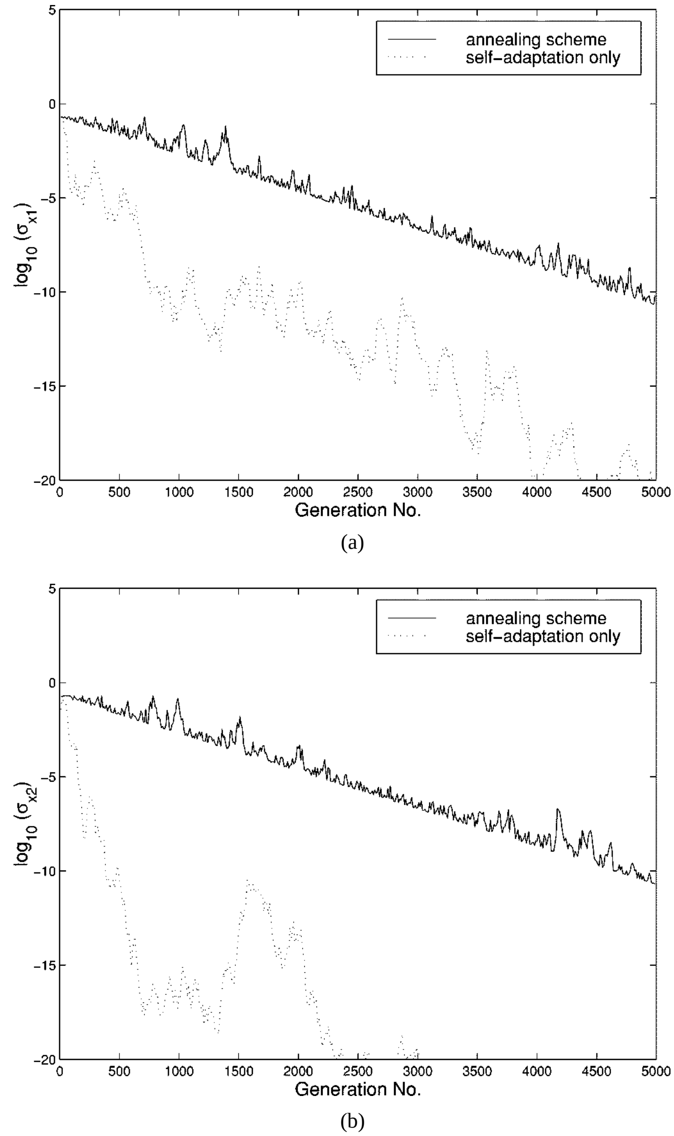


Fig. 7. Best individual's variances of x_1 and x_2 (Problem G1).

long as the convexity of the augmented Lagrangian is provided. For all four benchmark problems treated here, $\rho = 100$ turns out to be a reasonable choice.

Problem G1: Minimize

$$f(\boldsymbol{x}) = 5x_1 + 5x_2 + 5x_3 + 5x_4 - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

subject to

$$\begin{aligned} 2x_1 + 2x_2 + x_{10} + x_{11} &\le 10 \\ 2x_1 + 2x_3 + x_{10} + x_{12} &\le 10 \\ 2x_2 + 2x_3 + x_{11} + x_{12} &\le 10 \\ -8x_1 + x_{10} &\le 0 \\ -8x_2 + x_{11} &\le 0 \\ -8x_3 + x_{12} &\le 0 \\ -2x_4 - x_5 + x_{10} &\le 0 \\ -2x_6 - x_7 + x_{11} &\le 0 \end{aligned}$$

$$-2x_8 - x_9 + x_{12} \le 0$$

and bounds $0 \le x_i \le 1$, $i = 1, \cdots, 9$; $0 \le x_i \le 100$, $i = 10, 11, 12$; and $0 \le x_{13} \le 1$. The global minimum is known as

$$\boldsymbol{x}^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$$

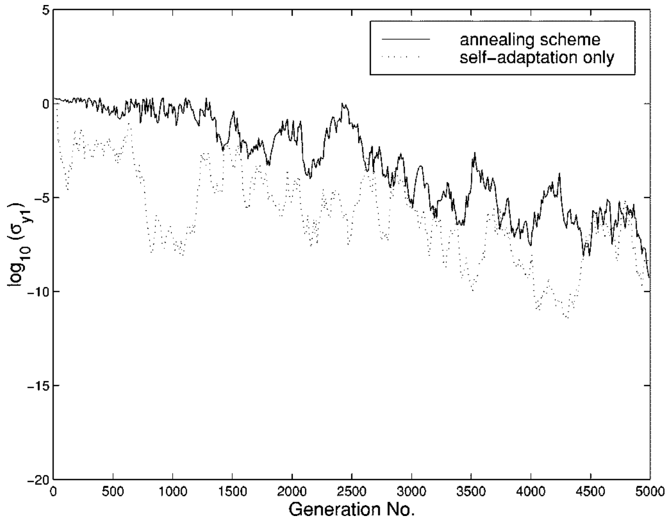
where $f(\boldsymbol{x}^*) = -15$.

Problem G7: Minimize

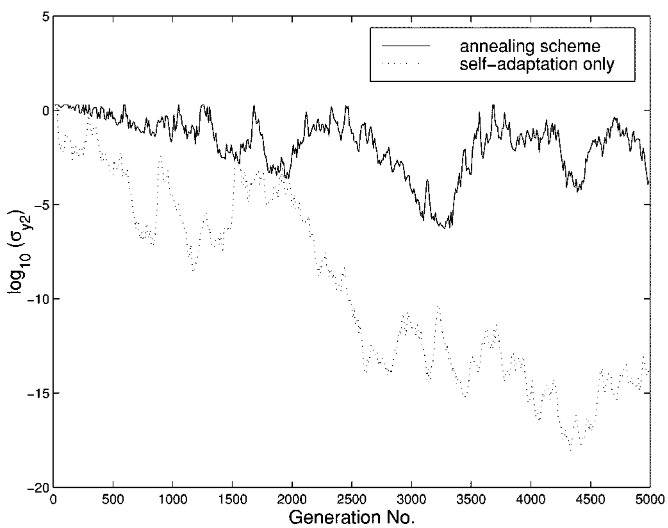
$$\begin{aligned} f(x) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 \\ + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 \\ + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45 \end{aligned}$$

subject to

$$\begin{aligned} 105 - 4x_1 - 5x_2 + 3x_7 - 9x_8 &\ge 0 \\ -3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4 + 120 &\ge 0 \\ -10x_1 + 8x_2 + 17x_7 - 2x_8 &\ge 0 \end{aligned}$$



(a)



(b)

Fig. 8. Best individual's variances of μ_1 and μ_2 (Problem G1).

$$
\begin{aligned}
-x_1^2 - 2(x_2-2)^2 + 2x_1x_2 - 14x_5 + 6x_6 &\geq 0\\
8x_1 - 2x_2 - 5x_9 + 2x_{10} + 12 &\geq 0\\
-5x_1^2 - 8x_2 - (x_3-6)^2 + 2x_4 + 40 &\geq 0\\
3x_1 - 6x_2 - 12(x_9-8)^2 + 7x_{10} &\geq 0\\
-0.5(x_1-8)^2 - 2(x_2-4)^{-}3x_5^2 + x_6 + 30 &\geq 0
\end{aligned}
$$

and bounds $-10 \leq x_i \leq 10$, $i = 1, \cdots, 10$. The global minimum is known as

$$
\begin{aligned}
\boldsymbol{x}^* = &(2.171996,\ 2.363683,\ 8.773926,\ 5.095984,\ 0.9906548,\\
&1.430574,\ 1.321644,\ 9.828726,\ 8.280092,\ 8.375927)
\end{aligned}
$$

where $f(\boldsymbol{x}^*) = 24.3062091$.

Problem G9: Minimize

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

TABLE I
RESULTS OF CEALM WITHOUT ROTATION ANGLE PARAMETERS (20 RUNS)

Problem	G1	G7	G9	G10
Exact Solution	-15.000	24.306	680.630	7049.331
$t_d = 1$				
(best)	-15.000	24.394	680.651	7074.600
(median)	-15.000	26.358	680.921	8920.235
(worst)	-11.281	30.823	681.666	14452.255
$t_d = 500$				
(best)	-15.000	24.307	680.630	7073.062
(median)	-15.000	24.313	680.632	7296.322
(worst)	-13.000	24.373	680.637	8007.965
$t_d = 1000$				
(best)	-15.000	24.306	680.630	7058.352
(median)	-15.000	24.310	680.630	7205.976
(worst)	-14.500	24.317	680.630	7430.667
$t_d = 2000$				
(best)	-15.000	24.306	680.630	7061.708
(median)	-15.000	24.307	680.630	7159.883
(worst)	-14.750	24.313	680.630	7256.905
$t_d = 3000$				
(best)	-15.000	24.306	680.630	7050.475
(median)	-14.999	24.307	680.630	7095.852
(worst)	-14.750	24.309	680.630	7217.698

subject to

$$
\begin{aligned}
127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 &\geq 0\\
282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 &\geq 0\\
196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 &\geq 0\\
-4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 &\geq 0
\end{aligned}
$$

and bounds $-10 \leq x_i \leq 10$, $i = 1, \cdots, 7$. The global minimum of this problem is located at

$$
\begin{aligned}
\boldsymbol{x}^* = &(2.330499,\ 1.951372,\ -0.4775414,\ 4.365726,\\
&-0.6244870,\ 1.038131,\ 1.594227)
\end{aligned}
$$

where $f(\boldsymbol{x}^*) = 680.6300573$.

Problem G10: Minimize

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

subject to

$$
\begin{aligned}
1 - 0.0025(x_4 + x_6) &\geq 0\\
1 - 0.0025(x_5 + x_7 - x_4) &\geq 0\\
1 - 0.01(x_8 - x_5) &\geq 0\\
x_1x_6 - 833.33252x_4 - 100x_1 + 83333.333 &\geq 0\\
x_2x_7 - 1250x_5 - x_2x_4 + 1250x_4 &\geq 0\\
x_3x_8 - 1250000 - x_3x_5 + 2500x_5 &\geq 0
\end{aligned}
$$

TABLE II
RESULTS OF CEALM WITH ROTATION ANGLE PARAMETERS (20 RUNS)

Problem	G1	G7	G9	G10
Exact Solution	-15.000	24.306	680.630	7049.331
$t_d = 1$				
(best)	-15.000	24.312	680.630	7126.113
(median)	-15.000	24.353	680.633	7513.907
(worst)	-15.000	24.516	680.659	11561.223
$t_d = 500$				
(best)	-15.000	24.306	680.630	7069.703
(median)	-15.000	24.309	680.630	7286.974
(worst)	-15.000	24.323	680.631	8153.088
$t_d = 1000$				
(best)	-15.000	24.306	680.630	7058.712
(median)	-15.000	24.308	680.630	7176.479
(worst)	-15.000	24.311	680.630	7390.182
$t_d = 2000$				
(best)	-15.000	24.306	680.630	7050.169
(median)	-15.000	24.307	680.630	7077.557
(worst)	-15.000	24.310	680.630	7136.600
$t_d = 3000$				
(best)	-15.000	24.306	680.630	7051.622
(median)	-15.000	24.306	680.630	7059.055
(worst)	-15.000	24.308	680.630	7089.063

TABLE III
COMPARISON OF VARIOUS EVOLUTIONARY METHODS (COURTESY OF M.I.T. PRESS FOR RESULTS OTHER THAN CEALM)

Problem		G1	G7	G9	G10
Exact		-15.000	24.306	680.630	7049.331
CEALM, $t_d = 3000$ (without rotation angles)	b	-15.000	24.306	680.630	7050.475
	m	-14.999	24.307	680.630	7095.852
	w	-14.750	24.309	680.630	7217.698
CEALM, $t_d = 3000$ (with rotation angles)	b	-15.000	24.306	680.630	7051.622
	m	-15.000	24.306	680.630	7059.055
	w	-15.000	24.308	680.630	7089.063
Dynamic Penalty	b	-15.000	25.486	680.787	N/A
	m	-15.000	26.905	681.111	
	w	-14.999	42.358	682.798	
Behavior Memory	b	-15.000	N/A	680.836	7485.667
	m	-15.000		681.175	8271.292
	w	-14.998		685.640	8752.412
Annealing Penalty	b	-15.000	N/A	680.642	7477.976
	m	-15.000		680.718	8206.151
	w	-15.000		680.955	9652.901
Superiority of Feasible Solutions	b	-15.000	N/A	680.805	N/A
	m	-15.000		682.682	
	w	-14.999		685.738	
Death Penalty	b	-15.000	25.653	680.847	7872.948
	m	-14.999	27.116	681.826	8559.423
	w	-13.616	32.477	689.417	8668.648

and bounds $100 \le x_1 \le 10\,000$; $1000 \le x_i \le 10\,000$, $i = 2, 3$; $10 \le x_i \le 1000$, $i = 4, \cdots, 8$. The problem has its global minimum at

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

where $f(\boldsymbol{x}^*) = 7049.330923$.

For Problem G1, Fig. 5 shows the history of the best individual's cost for two cases: evolution with and without an annealing scheme of $t_d = 500$ generation. When the annealing scheme is not used, the initial convergence rate is fast, but the global minimum is not achieved. Fig. 6 clearly shows that two active constraints (g_1 and g_2) at the exact solution remain inactive due to premature freezing. The variances of x_1 and x_2 for the best individual of X and those of μ_1 and μ_2 for the best individual of Y are shown in Figs. 7 and 8, respectively. In Fig. 7, we observe that the annealing scheme strictly controls the descent rates of the mutation variances to avoid premature freezing. As shown in Fig. 8, however, the variance of μ_2 floats well above the lower bound dictated by the annealing scheme; some multipliers may not converge, while the parameter vector x does. Once the whole population of the parameter vector converges to a solution satisfying all of the constraints, the augmented Lagrangian is no longer sensitive to the multiplier vector. Hence, we may not obtain a convergence for the multiplier vector in this case.

Finally, various combinations of μ and λ are tested to find out how crucial the selection of the population sizes is for convergence performance. Table IV shows the evolution results obtained without using rotation angle parameters. To remove any effects of the annealing scheme, $t_d = 1$ is used. It is observed that the population ratio μ/λ is not a critical parameter for coevolution since noticeable performance degradation is not observed for any value of μ/λ between 0.1 and 0.3. Hence, the default setting $\mu/\lambda = 0.15$ recommended in [11] seems to be reasonable for coevolution as well as evolution of a single population group. On the other hand, comparing the case of $\mu = 8$, $\lambda = 40$ with the case of $\mu = 16$, $\lambda = 80$, we observe that convergence characteristics are not improved by merely increasing the population size. However, the case of a smaller population size ($\mu = 4$, $\lambda = 20$) produces solutions (median values) that are a little worse than the cases of larger population. Hence, reducing λ below 20 to save computation time is not recommended.

In summary, the numerical experiments with the benchmark problems suggest that ρ and t_d are the only parameters that are problem dependent. Fortunately, these parameters can be adjusted without much difficulty, as previously explained.

VI. CONCLUDING REMARKS

In this paper, a novel coevolution method based on the augmented Lagrangian formulation is proposed for solving constrained optimization problems. This method uses the par-

TABLE IV
EVOLUTION RESULTS FOR VARIOUS POPULATION SIZES (20 CEALM RUNS WITHOUT ROTATION ANGLE PARAMETERS, $t_d = 1$)

Problem		G1	G7	G9	G10
Exact		-15.000	24.306	680.630	7049.331
$\mu = 4$	b	-15.000	25.096	680.648	7059.649
$\lambda = 40$	m	-14.999	26.523	680.839	10397.686
$(\mu/\lambda = 0.10)$	w	-14.750	48.886	683.281	20241.768
$\mu = 6$	b	-15.000	25.008	680.648	7100.527
$\lambda = 40$	m	-13.827	25.888	680.823	8411.921
$(\mu/\lambda = 0.15)$	w	-9.701	37.583	686.510	13023.181
$\mu = 8$	b	-15.000	24.394	680.651	7074.600
$\lambda = 40$	m	-15.000	26.358	680.921	8920.235
$(\mu/\lambda = 0.20)$	w	-11.281	30.823	681.666	14452.255
$\mu = 10$	b	-15.000	24.432	680.641	7116.462
$\lambda = 40$	m	-13.231	26.608	680.719	8560.488
$(\mu/\lambda = 0.25)$	w	-9.707	37.584	693.217	22710.479
$\mu = 12$	b	-15.000	24.791	680.656	7186.935
$\lambda = 40$	m	-13.828	26.630	680.828	8577.437
$(\mu/\lambda = 0.30)$	w	-9.963	50.383	686.510	13203.963
$\mu = 16$	b	-15.000	24.404	680.631	7194.791
$\lambda = 80$	m	-13.827	25.566	680.657	8647.929
$(\mu/\lambda = 0.20)$	w	-11.484	38.128	681.395	20681.262
$\mu = 4$	b	-15.000	24.951	680.643	7246.762
$\lambda = 20$	m	-12.562	27.942	681.056	9223.630
$(\mu/\lambda = 0.20)$	w	-10.571	37.300	691.740	14773.082

allel evolution of two population groups: one is for the parameter vector, and another for the multiplier vector. The fitness of each individual is evaluated according to the security strategy of each group. For selection, recombination, and mutation of each group, the features of an evolution strategy are adopted without modification. In addition, an annealing scheme is introduced to reduce the possibility of premature freezing (or premature self-adaptation). Numerical results with typical benchmark problems of nonlinear programming show that the proposed coevolution method gives very consistent solutions with a numerical accuracy comparable to that of current deterministic methods. Equipped with a number of advantages of the evolutionary approach, such as capability of finding the global solution and robustness to the initialization procedure, the proposed method is believed to have a great potential as a practical and reliable optimization tool for various engineering problems. To be established as a standard optimization tool, the coevolution method needs further study on convergence analysis and reduction in computation time.

ACKNOWLEDGMENT

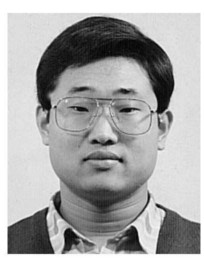
The authors are grateful to the reviewers and D. B. Fogel for their helpful comments and valuable suggestions during the revision of this paper.

REFERENCES

[1] Z. Michalewicz and M. Schoenauer, "Evolutionary algorithms for constrained parameter optimization problems," *Evol. Comput.*, vol. 4, no. 1, pp. 1–32, 1996.
[2] J. H. Kim and H. Myung, "Evolutionary programming techniques for constrained optimization problems," *IEEE Trans. Evol. Comput.*, vol. 1, no. 2, pp. 129–140, 1997.
[3] J. Paredis, "Co-evolutionary constraint satisfaction," in *Proc. 3rd Conf. Parallel Problem Solving from Nature*. New York: Springer-Verlag, 1994, pp. 46–55.
[4] C. S. Park and M. J. Tahk, "A numerical solver based on evolutionary computation for minimax problems," in *Proc. JSASS Aircraft Symp.*, Tokyo, Japan, Oct. 1997.
[5] ——, "A co-evolutionary minimax solver and its application to autopilot design," in *Proc. AIAA Guidance, Navigation, Contr. Conf.*, Boston, MA, Aug. 1998.
[6] J. Hur and M. J. Tahk, "Parameter-robust control design using co-evolution," in *Proc. JSASS Aircraft Symp.*, Yokosuka, Japan, Oct. 1998.
[7] M. J. Tahk, "Co-evolution for engineering optimization problems: Minimax design and constrained optimization," in *Special Lecture, JSASS Aircraft Symp.*, Yokosuka, Japan, Oct. 1998.
[8] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*, 2nd ed. New York: Wiley, 1993.
[9] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*. New York: Academic, 1981.
[10] T. Basar and G. J. Olsder, *Dynamic Noncooperative Game Theory*. New York: Academic, 1982.
[11] T. Bäck, *Evolutionary Algorithms in Theory and Practice*. Oxford, U.K.: Oxford Univ. Press, 1996.

Min-Jea Tahk received the B.S. degree from Seoul National University, Seoul, Korea, in 1976 and the M.S. and Ph.D. degrees from The University of Texas at Austin, in 1983, and 1986, all in aerospace engineering.

From 1976 to 1981, he was a Research Engineer at the Agency for Defense Devlopment, Taejon, Korea, and from 1986 to 1989, he was employed by Integrated Systems, Inc., Santa Clara, CA. He is presently an Associate Professor of Aerospace Engineering at Korea Advanced Institute of Science and Technology, Taejon, Korea. His research interests include evolutionary computation with its application to flight control design and missile guidance using neural networks.



Byung-Chan Sun received the B.S. degree from Seoul National University, Seoul, Korea, in 1991 and the M.S. and Ph.D. degrees from Korea Advanced Institute of Science and Technology, in 1993, and 1997, all in aerospace engineering.

Since then, he has been a postdoctoral student at the Automatic Control Research Center of Seoul National University, and at Shinshu University, Nagano, Japan. He is currently a Research Engineer at Korea Aerospace Research Institute, Taejon, Korea. His research interests include evolutionary optimization techniques, robust control theory, and flight control of launch vehicles.