Global Optimization Using Hybrid Approach

TING-YU CHEN, YI LIANG CHENG
Departmant of Mechanical Engineering
National Chung Hsing University
250 Kuo Kuang Road
Taichung, Taiwan 40227
REPUBLIC OF CHINA

Abstract: - The paper deals with a global optimization algorithm using hybrid approach. To take the advantage of global search capability the evolution strategy(ES) with some modifications in recombination is used first to find the near-optimal solutions. The sequential quadratic programming(SQP) is then used to find the exact solution from the solutions found by ES. One merit of the algorithm is that the solutions for multimodal problems can be found in a single run. Ten popular test problems are used to test the proposed algorithm. The results are satisfactory in quality and efficiency.

Key-Words: -Global optimization algorithm, hybrid approach

1 Introduction

The global optimization has been a hot research topic for a long time. With the progress of evolutionary computation, many global optimization algorithms have been developed using various evolutionary methods. Tu and Lu[1] proposed a stochastic genetic algorithm(StGA) to solve global optimization problems. They divided the search space dynamically and explored each region by generating five offspring. The method was claimed to be efficient and robust. Toksari[2] developed an algorithm based on ant colony optimization(ACO) to find the global solution. In his method each ant searches the neighborhood of the best solution in the previous iteration. Liang et al.[3] used particle swarm optimization (PSO) to find global solutions for multimodal functions. Their method modified the original PSO by using other particles' historical best data to update the velocity of a particle. In doing so, the premature convergence can be avoided. Zhang et al.[4] proposed a method called estimation of distribution algorithm with local search(EDA/L). This method used uniform design to generate initial population in the feasible region. The offspring are produced by using statistical information obtained from parent population. The local search is used to find the final solution.

Chen and Hsu[5] developed an algorithm called rank-niche evolution strategy(RNES) to find Pareto-optimal solutions for multi-objective optimization problems. The algorithm was based on evolution strategy(ES) incorporated with a novel fitness function. Since the algorithm is pretty simple and generates many Pareto optimal solutions in a single run, it is modified to solve global optimization problems with single or multiple solutions in this paper. Ten widely used test functions are employed to test against the algorithm. The global solutions for all test problems are found.

2 Brief Review of ES

The evolution strategy(ES) was developed by Rechenburg[6] and extended later by Schwefel[7]. There are three evolutionary steps in ES. The first one is recombination and it is executed by one of the following formulas.

$$x_i^{'} = \begin{cases} x_{a,i} & \text{(A) no recombination} \\ x_{a,i} \text{ or } x_{b,i} & \text{(B) discrete} \\ 0.5(x_{a,i} + x_{b,i}) & \text{(C) intermediate} \\ x_{a_i,i} \text{ or } x_{b_i,i} & \text{(D) global, discrete} \\ 0.5(x_{a_i,i} + x_{b_i,i}) & \text{(E) global, intermediate} \end{cases} \tag{1}$$

where $x_i^{'}$ is the new ith design variable. $x_{a,i}$ and $x_{b,i}$ are the ith design variables of two individuals a and b randomly chosen from μ parent individuals, respectively. These two parents are used to generate a specific new individual for operations (B) and (C). $x_{a_i,i}$ and $x_{b_i,i}$ are the ith design variables of two individuals randomly chosen from μ parent individuals. In operations (D) and (E) each new design variable may come from two different parents. The number of so generated new individuals is λ and this value is usually several times of μ.

In addition to the five formulas given by Schwefel, Chen[8] developed another three formulas as follows:

$$x_i^{'} = (1 - t_1)x_{a,i} + t_1 x_{b,i}$$
$$t_1 \in [0,1] \tag{2}$$

$$x_i^{'} = (x_{a,i} + t_2 x_{a,i}) \quad \text{or} \quad (x_{b,i} + t_2 x_{b,i}) \qquad t_2 \in [-0.5, 0.5] \tag{3}$$

$$x_i^{'} = (x_{a,i} + x_{b,i} + \cdots + x_{m,i}) / m, \qquad m \in [1, \mu] \tag{4}$$

Where t_1 is a random number between 0 and 1. t_2 is a random number between -0.5 and 0.5. m is an arbitrary integer between 1 and μ.

The purpose of adding formula (2) is to provide the chance of generating any value between $x_{a,i}$ and $x_{b,i}$. Formula (3) gives the chance to generate a value neighboring $x_{a,i}$ or $x_{b,i}$. Formula (4) finds the centroid of some randomly selected individuals. The adding of the three formulas to the original five formulas can increase the search area in the design space.

The second step in ES is the mutation operation. The mutation is done by the following formulas.

$$x_i^{'} = x_i + z_i \sigma_i^{'} \tag{5}$$

and

$$\sigma_i^{'} = \sigma_i e^{(\tau^{'} z + \tau z_i)}$$

$$\tau^{'} = \frac{1}{\sqrt{2n}} \tag{6}$$

$$\tau = \frac{1}{\sqrt{2\sqrt{n}}}$$

where $x_i^{'}$ is the mutated ith design variable from x_i. x_i is the ith design variable of an individual from recombination. $z_i \sigma_i^{'}$ is the change for the ith design variable of that individual. $\sigma_i^{'}$ is the updated self-adaptive variable associated with the ith design variable. σ_i is the self-adaptive variable used for the previous mutation in the last generation. The variable σ_i is also subjected to the same recombination operation in equation (1). n is the number of design variables. z and z_i are two random numbers from a normal distribution $N(0,1)$ with mean zero and standard deviation one.

Equation (7) is the probability density function of the normal distribution.

$$P(z_i) = \frac{1}{\sqrt{2\pi}} e^{\frac{-(z_i - 0)^2}{2}} \tag{7}$$

where the mean value of the normal distribution is 0 and the standard deviation is 1.

The last step in ES is the selection operation which is used to choose some best individuals resulted from mutation operation to enter the next generation. Two approaches are available. One is called (μ, λ) selection and the other one is named $(\mu + \lambda)$ selection. For (μ, λ) selection, the best μ individuals are chosen to enter the next generation from the λ offspring. The $(\mu + \lambda)$ selection combines λ offspring with μ parents in current generation first and then chooses the best μ individuals from the combined pool to be parents in the next generation. The (μ, λ) selection may have better chance to find the global solution while the $(\mu + \lambda)$ selection may accelerate the convergence rate.

3 RNES Algorithm

The rank-niche evolution strategy(RNES) was developed by Chen and Hsu[5] to solve multi-objective optimization problems using evolution strategy. For finding global optimum solution or solutions, the RNES is modified as follows:

(1) Use random numbers to generate μ individuals in the design space as the initial population. Establish an external elite pool that contains some best individuals.
(2) Perform recombination operation using equation (3) to produce λ temporary offspring.
(3) Perform mutation operation using equation (5).
(4)Compute objective function values for all λ individuals.
(5) Compute constraint function values. If the problem has constraints, compute all constraint function values for all λ individuals.
(6) Select elites using (μ, λ) approach and update the external elite pool. For unconstrained optimization problems if the individual with smallest objective function value is better than the one in the elite pool, replace the one in the pool by the best one obtained in this generation. For constrained optimization problems, choose the best feasible solution and update the one in the external pool if necessary. If no feasible solution is found, no updating is performed. For multimodal problems multiple global solutions may exist. In order to find these solutions in a single run, several different elites are saved in the external pool. To avoid ES search converging to a single solution, these elites must be separated from each other by a distance given by equation (8).

$$d^{eli}(i, j) = \sqrt{\sum_{k=1}^{n} \left(\frac{x_{i,k}^{eli} - x_{j,k}^{eli}}{x_k^U - x_k^L} \right)^2} \geq \varepsilon_{eli} \tag{8}$$

where $d^{eli}(i,j)$ is the normalized distance between elite i and elite j. $x_{i,k}^{eli}$ and $x_{j,k}^{eli}$ are the kth design variable for the ith and the jth elites, respectively. x_k^U and x_k^L are the upper and lower bound for the kth design variable, respectively. ε_{eli} is a user defined smallest distance between two different elites.

(7) Based on objective function value and constraint violation, select the best μ individuals to enter the next generation. For unconstrained optimization problems, put the λ individuals in ascending order based on their objective function values. The first μ individuals are chosen to enter next generation. For constrained optimization problems, the selection rules will be discussed in the next section.

(8) If the maximum number of generation is reached, go to step (9). Otherwise, go to step (2).

(9)Use sequential quadratic programming(SQP) to find the final solutions. The starting points for SQP are those individuals saved in the external elite pool. The best solution or solutions resulted from SQP or ES search are taken as the global solutions.

4 Selection Steps for Constrained Problems

The selection rules for constrained problems are executed in the following order.

(1)Select feasible solution to enter the next generation first. If the number of feasible solution is greater than μ, select the best μ individuals according to their objective function values. If the number of feasible solution is less than μ, select all feasible solutions first and go to rule (2).

(2)For infeasible solutions compute the normalized violation for each violated constraint. Divide the infeasible solutions into several ranks based on the domination check of constraint violation. The domination check proceeds as follows: For any two individuals A and B, if every constraint violation of A is less than that of B, then B is dominated by A. Otherwise, A and B do not dominate each other.

Perform domination check on all infeasible solutions using the normalized violations to find the non-dominated ones. These infeasible solutions are assigned to the first rank. Repeat the domination check for the rest infeasible solutions to allocate individuals to other ranks. The higher the rank is, the less the overall constraint violation.

(3)Select infeasible individuals from rank one first. If the number of individuals in rank one is less than the required number to fill up μ, go to rank two and repeat this process until the required number μ is reached. If the number of individuals in the lowest rank used to fill up μ is greater than the required number, use objective function values to determine the ones to be selected.

5 Numerical Examples

Ten test problems including five unconstrained and five constrained problems are used to test the proposed algorithm. The global solutions are found for all test problems. Due to limited space only four of them are shown in this paper. The first one is the Rastrigin function [9]. The second one is the Bumpy equation[10]. The first two problems are for unconstrained optimization. The third one is C-Bumpy equation[10]. The last one is Himmeblau equation[11]. The last two problems are for constrained optimization.

Problem 1: Rastrigin function[9]

The optimization problem is formulated as follows.

$$\begin{aligned} &\min. \\ &F(\bar{x}) = 20 + {x_1}^2 + {x_2}^2 - 10[\cos(2\pi x_1) + \cos(2\pi x_2)] \\ &subject\ to \quad -5 \le x_1 \le 5 \qquad (9) \\ &\qquad\qquad\quad -5 \le x_2 \le 5 \end{aligned}$$

Fig. 1 shows the multimodal nature of the problem. Table 1 lists the solutions found by the proposed algorithm and other papers. It is clear that the proposed method finds the best solution compared with other methods. Also the cpu time spent for the proposed method is less than those of the other two methods.

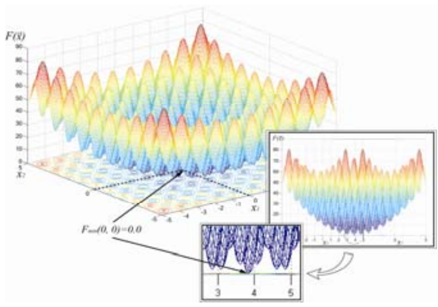

Fig. 1 Rastrigin function

Table 1 Optimum solutions of problem 1

	Exact Solution[9]	ES+SQP	Lee[12]	GA [12]
x_1	0.0	-0.15E-08	-0.002167	-0.000153
x_2	0.0	0.37E-07	-0.000214	0.000580
OBJ	0.0	0.00E+00	0.00094	0.00007
No.e	NA*	2000	1400	2500
time(s)	NA*	<1	109	1

No.e is No. of function evaluations, *time(s)* is CPU time(sec), NA* is not available

Problem 2: Bumpy equation[10]

The unconstrained problem is defined as

$$\max. \quad F(\bar{x}) = \left| \frac{\cos^4 x_1 + \cos^4 x_2 - 2\cos^2 x_1 \cos^2 x_2}{\sqrt{x_1^2 + 2x_2^2}} \right|$$

$$\textit{subject to} \quad 0 \le x_1 \le 10, \qquad 0 \le x_2 \le 10 \tag{10}$$

The results of various methods are shown in Table 2. The proposed method spends the least computational time to find the global solution.

Table 2 Optimum solutions of problem 2

	Exact Solution[10]	ES+SQP	Lee[12]	GA [12]
x_1	1.3932	1.39522	1.3888	1.3942
x_2	0	0.0	0.000182	0.000153
OBJ	0.67367	0.673663	0.67364	0.67366
No.e	NA	280	550	2500
time(s)	NA	<1	34	1

Problem 3: C-Bumpy equation[10]

The objective function of this problem is the same as problem 2. But two constraints are added. The optimization problem is defined as

$$\max. \quad F(\bar{x}) = \left| \frac{\cos^4 x_1 + \cos^4 x_2 - 2\cos^2 x_1 \cos^2 x_2}{\sqrt{x_1^2 + 2x_2^2}} \right|$$

$$\textit{subject to} \quad \begin{aligned} & g_1(\bar{x}) \equiv x_1 x_2 > 0.75, \\ & g_2(\bar{x}) \equiv x_1 + x_2 \le 15, \\ & 0 \le x_1 \le 10, \\ & 0 \le x_2 \le 10, \end{aligned} \tag{11}$$

Fig. 2 shows the local solutions of the problem. Table 3 gives the solutions obtained by various approaches. Again the proposed method provides better solution. The computational time is also the least one.

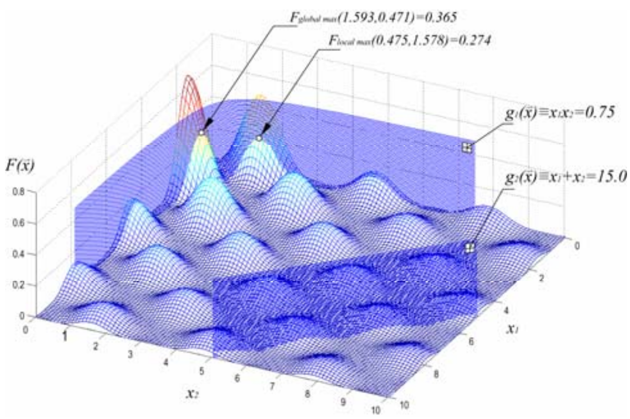

Fig. 2 C-Bumpy function

Table 3 Optimum solutions of problem 3

	Exact Solution[10]	ES+SQP	Lee[12]	DPF[12]	APF[12]
x_1	1.593	1.601	1.639	1.650	1.563
x_2	0.471	0.468	0.459	0.456	0.480
OBJ	0.365	0.365	0.362	0.361	0.363
No.e	NA	1900	900	2500	2500
time(s)	NA	<1	61	1	1

Problem 4: Himmeblau function[11]

This constrained optimization problem having five design variables is defined as

$$\min. \quad F(\bar{x}) = 5.3578547 x_3^2 + 0.8356891 x_1 x_5 + 37.29329 x_1 - 40792.141$$

$$\textit{subject to}$$

$$g_1(\bar{x}) \equiv 85.334407 + 0.0056858 x_2 x_5 + 0.0006262 x_1 x_4 - 0.0022053 x_3 x_5$$

$$g_2(\bar{x}) \equiv 80.51249 + 0.0071317 x_2 x_5 + 0.0029955 x_1 x_2 + 0.0021813 x_3^2$$

$$g_3(\bar{x}) \equiv 9.300961 + 0.0047026 x_3 x_5 + 0.0012547 x_1 x_3 + 0.0019085 x_3 x_4$$

$$0 \le g_1(\bar{x}) \le 92, \ 90 \le g_2(\bar{x}) \le 110, \ 20 \le g_3(\bar{x}) \le 25$$

$$78 \le x_1 \le 102, \ 33 \le x_2 \le 45, \ 27 \le x_3 \le 45, \ 27 \le x_4 \le 45, \ 27 \le x_5 \le 45 \tag{12}$$

The optimum solutions are listed in Table 4. The proposed method finds the second best solution and spends the least computational time.

Table 4 Optimum solutions of problem 4

	ES+SQP	Lee[12]	Himmeblau[11]	Homaifar[13]
x_1	78.000	79.293	78.62	78.000
x_2	33.000	34.186	33.44	33.000
x_3	29.995	31.186	31.07	29.995
x_4	45.000	39.920	44.18	45.000
x_5	36.776	36.195	35.22	36.776
OBJ	-30665.45	-30225.7	-30373.9	-30665.6
No.e	800	1650	NA	NA
time(s)	<1	138	NA	NA

6 Conclusion

The proposed global optimization algorithm using hybrid approach of ES plus SQP has been proved to be successful in solving 10 test problems. For most test problems the proposed method not only finds the best solution compared with other methods but also spends the least computational time.

References:

[1]Z. Tu and Y. Lu, A robust stochastic genetic algorithm(StGA) for global numerical optimization, IEEE Transactions on Evolutionary Computation, Vol. 8, No. 5, 2004, pp. 456-470.

[2]M. D. Toksari, Ant colony optimization for finding the global minimum, Applied Mathematics and Computation, vol. 176, 2006, pp. 308-316.

[3]J.J. Liang, A.K. Qin, P.N. Suganthan and S. Baskar, comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE transactions on Evolutionary Computation, Vol. 10, No. 3, 2006, pp. 281-295.

[4] Q. Zhang, J. Sun, E. Tsang and J, Ford, Hybrid estimation of distribution algorithm for global optimization, Engineering Computations, Vol. 21, No. 1, 2004, pp. 91-107.

[5]T.Y. Chen and Y.S. Hsu, A Multiobjective optimization solver using rank-niche evolution strategy, Advances in Engineering Software, Vol. 37, 2006, pp. 684-699.

[6]I. Rechenberg, Evolutionsstrategie: Optimierung technisher systeme nach prinzipien der biologischen evolution, Frommann-Holzboog Verlag, Stuttgart, 1973.

[7]H.-P. Schwefel, Numerical optimization of computer models, John Wiley & sons Ltd, Chichester, U.K, 1977.

[8]H.C. Chen, Discrete and mixed-variable evolution strategy, Master's thesis, Department of Mechanical Engineering, National Chung Hsing University, Taiwan, 2006.

[9]D. Buche, N.N. Schraudolph and P. Koumoutsakos, Accelerating evolutionary algorithms with Guassian process fitness function models, IEEE transactions on Systems, Man, and cybernetics-Part C: applications and Reviews, Vol. 35, No. 2, 2005, pp. 183-194.

[10]A.J. Keane, Experiences with optimizers in structural design, Proceeding of Adaptive Computing in Engineering Design and control, 1994, pp. 14-27.

[11]D.M. Himmebleau, Applied nonlinear programming, McGraw-Hill, New York, 1972.

[12]C.C. Lee, Reproducing kernel approximation method for structural optimization using genetic algorithms, Ph D dissertation, National Taiwan University, 2006.

[13]A. Homaifar, S.H.Y. Lai and X. Qi, Constrained optimization via genetic algorithms, Simulation, Vol. 62, No. 4, 1994, pp. 242-254.