Global Optimization Using Hybrid Approach

TING-YU CHEN, YI LIANG CHENG
Department of Mechanical Engineering
National Chung Hsing University
250 Kuo Kuang Road
Taichung, Taiwan 40227
tyc@dragon.nchu.edu.tw http://wwwme.nchu.edu.tw

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 formulas and elites keeping 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. Eight 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, evolution strategy

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.

In general the evolutionary algorithms are though to have a better chance to find the global solution from multiple search points. However, the evolutionary algorithms also have some drawbacks. The first one is that it takes significant number of function evaluations. This may consumes a lot of computational times especially used in structural optimization. The second drawback is that sometimes it only finds near-optimal solution. To reduce the effect of the first drawback some approximate analysis methods such as artificial neural network and response surface methodology may be employed to replace the time-consuming exact analyses. To overcome the second drawback some gradient-based local search method may be use to locate the exact solution.

Taking the advantage of evolutionary algorithms and avoiding its disadvantage, a new hybrid global optimization algorithm GOES(global optimization with evolution strategy) is developed in this paper. This algorithm integrates evolution strategy with the sequential quadratic programming(SQP) to find the exact global solution. Eight widely used test problems 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[5] and extended later by Schwefel[6]. There are three evolutionary steps in ES. The first one is recombination and it is executed by one of the following formulas.

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

where $x_i^{'}$ is the new ith design variable after recombination. $x_{a,i}$ 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 using formulas (B) and (C). $x_{a_i,i}$ and $x_{b_i,i}$ are also the ith design variables of two individuals randomly chosen from μ parent individuals. However, in formulas (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.

To further refine the search space, Chen[7] developed another three formulas for recombination as follows:

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

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

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

Where t_1 is a uniformly distributed random number between 0 and 1. t_2 is a uniformly distributed 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.

$$x_i^{'} = x_i + z_i \sigma_i^{'} \quad (5)$$

and

$$\sigma_i^{'} = \sigma_i e^{(\tau^{'} z + \tau z_i)}, \quad \tau^{'} = \frac{1}{\sqrt{2n}}, \quad \tau = \frac{1}{\sqrt{2\sqrt{n}}} \quad (6)$$

where $x_i^{'}$ is the mutated ith design variable from x_i. x_i is the ith design variable of an individual after 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 step in the last generation. The variable σ_i is also subjected to the same recombination operation. 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.

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

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 from the λ offspring to enter the next generation. 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. The flow chart of ES is shown in Fig. 1.

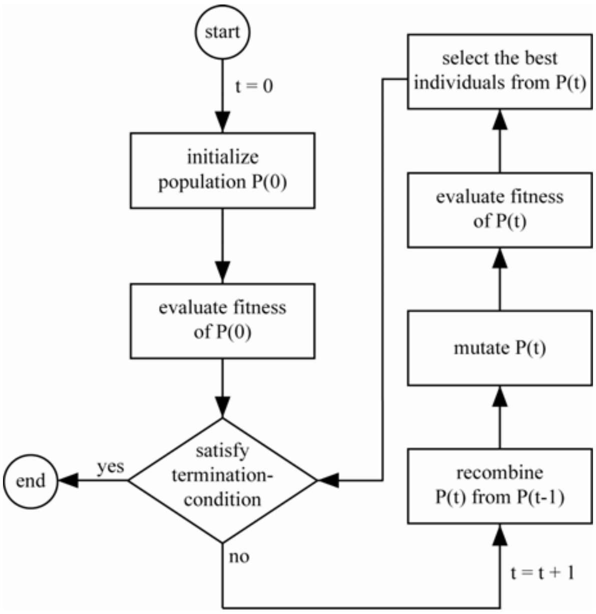

Fig. 1 Flow chart of evolution strategy

3 GOES Algorithm

The GOES algorithm is developed in this paper to find the global optimum solution(s). The algorithm can be divided into three phases. The first phase is basically ES with some modifications. The second phase is SQP search. The last phase is to determine the global solutions(s) from previous two phases. The followings are the steps of GOES algorithm.

(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. For unconstrained problems skip this step.

(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 identify these global solutions during ES search, a criterion to differentiate different solutions is established as follows.

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

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 small value given by users. If the inequality is satisfied, the two individuals i and j are though to be two different solutions and saved in the external pool separately. Otherwise, they are the same solution.

(7) Based on objective function value and constraint violation, select the best μ individuals to enter the next generation. For unconstrained minimization optimization problems, put the λ individuals in ascending order based on their objective function values. The first μ individuals are chosen to enter the 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 the sequential quadratic programming(SQP) to find the exact solutions. The starting points for SQP are those individuals saved in the external elite pool.

(10)Determine the global solution(s). 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 in GOES 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 step (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 them to other ranks. The higher the rank is, the less the overall constraint violation. Go to step (3).

(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. Fig. 2 is the flow chart of GOES algorithm.

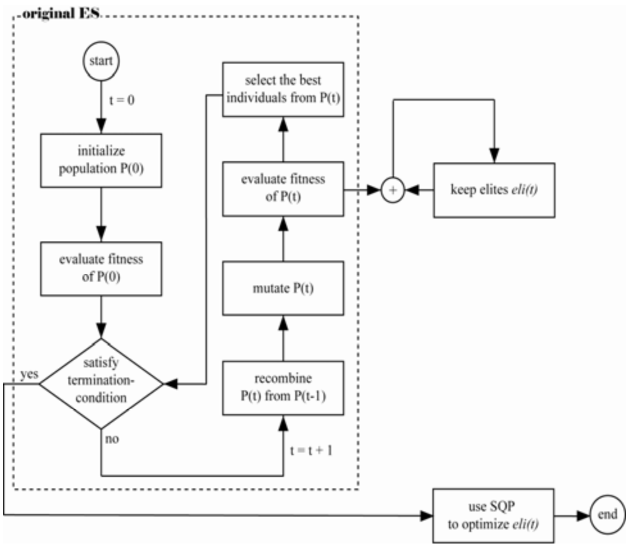

Fig. 2 Flow chart of GOES

5 Numerical Examples

Eight test problems including four unconstrained and four constrained problems are used to test the proposed algorithm. The global solutions are found for all test problems.

Problem 1: Branin RCOS function[8]

This unconstrained optimization problem is formulated as follows:

$$\begin{aligned} \min. \quad & F(\bar{x}) = a(x_2 - bx_1^2 + cx_1 - d)^2 \\ & + e(1-f)\cos(x_1) + e \\ \textit{subject to} \quad & -5 \le x_1 \le 10 \\ & 0 \le x_2 \le 15 \end{aligned} \tag{9}$$

where $a = 1$, $b = 5.1/(4\pi^2)$, $c = 5/\pi$, $d = 6$, $e = 10$, $f = 1/(8\pi)$

Fig. 3 shows the contour of the objective function. Clearly it has three global solutions. Table 1 shows the solutions of this problem. In order to understand the capability of GOES to find all global solutions in a single run, the algorithm is run 100 times with different initial population. In Table 1 the fraction within the parentheses under ES in the second column means that in 29 times GOES algorithm finds all three global solutions. For the other 71 times two of the three global solutions are found. That is for this particular problem GOES has 29% of chance to find all three global solutions in a single run and 71% of chance to find two of the three solutions. The reason for failing to find all three solutions is sometimes ES search fail to cover all three areas that contain the global solutions. The rest data in the table give results obtained by using SQP only. The SQP solver is executed 100 times with different initial pints. The SQP solver successfully finds the global solution from any initial point. However for any single run of SQP it can only find one of the three global solutions. Although GOES can not guarantee finding all global solutions in any single run, its advantage over gradient-based method is apparent.

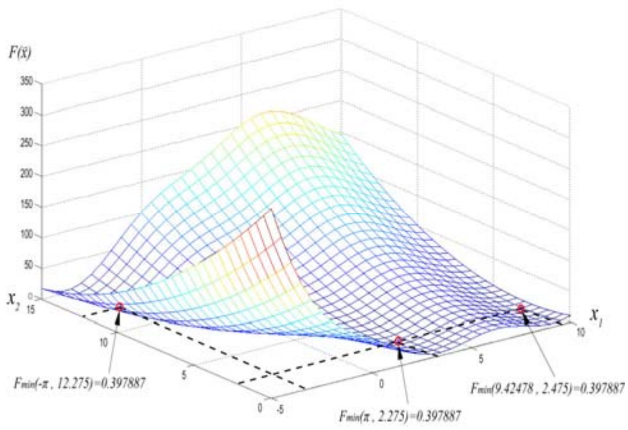

Fig. 3 Branin RCOS function

Table 1 Global solutions of *Branin RCOS function*

	Exact Solution			ES		
	Global			(29/100)		
x_1	- π	π	9.425	-3.129	3.143	9.367
x_2	12.275	2.275	2.475	12.156	2.276	2.437
OBJ	0.398	0.398	0.398	0.407	0.398	0.414

Table 1(continued)

	GOES			SQP		
				(30/100)	(34/100)	(36/100)
x_1	-3.140	3.141	9.425	- π	π	9.425
x_2	12.268	2.276	2.475	12.275	2.275	2.275
OBJ	0.398	0.398	0.398	0.398	0.398	0.398

Problem 2: Bumpy function[9]

This is another unconstrained optimization problem. The mathematical formulation of this problem is given below.

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

$$subject\,to \quad 0 \le x_1 \le 10$$

$$0 \le x_2 \le 10$$

(10)

Fig. 4 shows the contour of the function. Although it has only one global solution, it also has many local solutions. To solve this type of problem using gradient-based solver only, at most of time local solutions will be found. Table 2 lists the solutions by GOES and other papers. In this table Lee's approach was called reproducing kernel approximation method using genetic algorithms. The GA solution at the last column in Table 2 was also provided by Lee's paper. The hardware used by Lee was personal computer with Pentium 4 CPU 3GHz and DDR Ram 1 GB which is the same as we use. It is clear that the solution found by GOES is closest to the exact solution. The number of function evaluations and the CPU time is also the least one of the three.

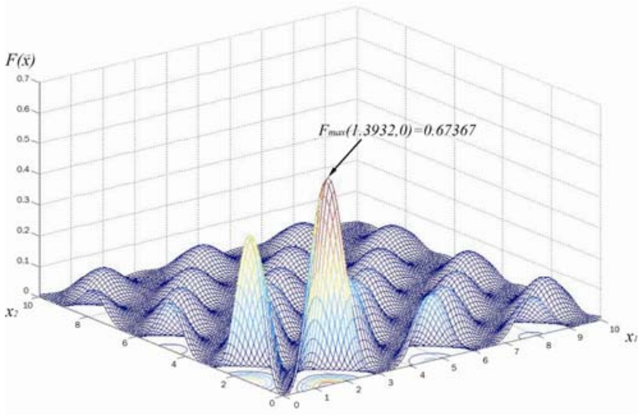

Fig. 4 Bumpy function

Table 2 Global solutions of B*umpy function*

	Exact Solution[9]	GOES	Lee[10]	GA [10]
x_1	1.3932	1.39522	1.3888	1.3942
x_2	0	0	0.00018	0.000153
OBJ	0.67367	0.67366	0.67364	0.67366
No.e	NA*	280	550	2500
time(s)	NA*	<1	34	1

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

Problem 3: Ackley function[11]

This unconstrained optimization problem is defined as

$$\min. \quad F(\vec{x}) = -20e^{-0.2\sqrt{0.5(x_1^{\,2}+x_2^{\,2})}} - e^{0.5[\cos(2\pi x_1)+\cos(2\pi x_2)]} + 20 + e$$

$$subject\ to. \quad -30 \le x_1 \le 30$$

$$-30 \le x_2 \le 30$$

(11)

Fig. 5 is the contour of this function. It is clear that this problem has a single global solution surrounded by many local solutions. This increases the difficulty of finding the global solution. Table 3 gives the solutions of this problem. Again the solution by GOES is closest to the exact solution and the CPU time is the least one compared with the other two solutions.

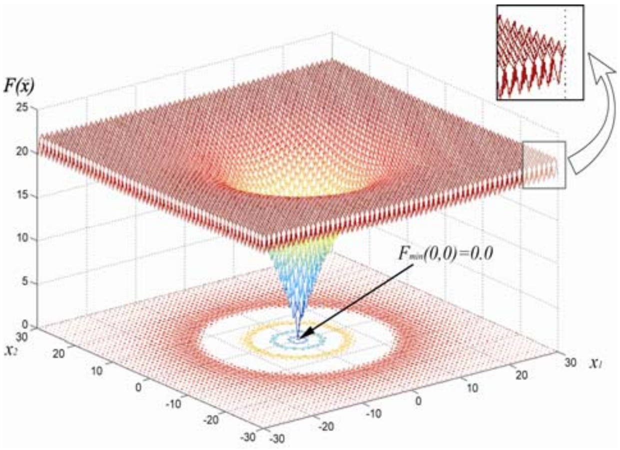

Fig. 5 Ackley function

Table 3 Global solutions of *Ackley function*

	Exact Solution[11]	GOES	Lee[10]	GA [10]
x_1	0	-0.000001	-0.002899	0.000885
x_2	0	0	-0.00119	-0.001373
OBJ	0	0.000004	0.009126	0.004692
No.e	NA*	1500	1200	2500
time(s)	NA*	<1	86	1

Problem 4: Rastrigin function[12]

This unconstrained optimization problem is formulated as follows.

min.

$$F(\bar{x}) = 20 + x_1^2 + x_2^2 - 10[\cos(2\pi x_1) + \cos(2\pi x_2)]$$

$$subject\ to \quad -5 \le x_1 \le 5$$

$$-5 \le x_2 \le 5$$

(12)

Fig. 6 shows the multimodal nature of the problem. Table 4 gives the solutions found by GOES and other papers. It is clear that GOES finds the best solution compared with other methods. Also the CPU time spent by GOES is less than those of the other two methods.

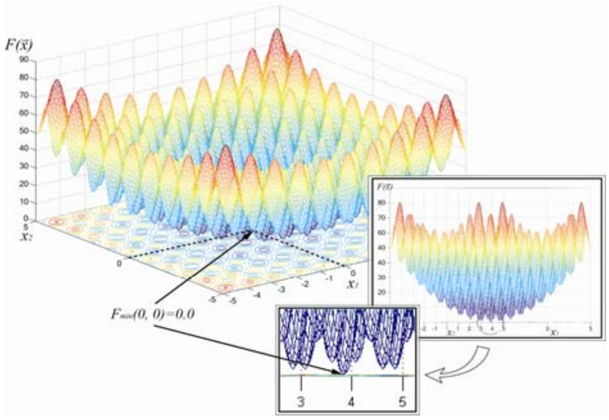

Fig. 6 Rastrigin function

Table 4 Global solutions of *Rastrigin Function*

	Exact Solution[12]	GOES	Lee[10]	GA [10]
x_1	0	-1.50E-09	-0.002167	-0.000153
x_2	0	3.70E-08	-0.000214	0.00058
OBJ	0	0.00E+00	0.00094	0.00007
No.e	NA*	2000	1400	2500
time(s)	NA*	<1	109	1

Problem 5:

This constrained optimization problem is formulated as follows:

$$\min.\quad F(\bar{x}) = 5x_1 + 5x_2 + 5x_3 + 5x_4 - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=1}^{9} y_i$$

subject to

$$\begin{aligned}
& g_1(\bar{x}) \equiv 2x_1 + 2x_2 + y_6 + y_7 \le 10, \\
& g_2(\bar{x}) \equiv -8x_1 + y_6 \le 0, \\
& g_3(\bar{x}) \equiv 2x_1 + 2x_3 + y_6 + y_8 \le 10, \\
& g_4(\bar{x}) \equiv -8x_2 + y_7 \le 0, \\
& g_5(\bar{x}) \equiv 2x_2 + 2x_3 + y_7 + y_8 \le 10, \\
& g_6(\bar{x}) \equiv -8x_3 + y_8 \le 0, \\
& g_7(\bar{x}) \equiv -2x_4 - y_1 + y_6 \le 0, \\
& g_8(\bar{x}) \equiv -2y_2 - y_3 + y_7 \le 0, \\
& g_9(\bar{x}) \equiv -2y_4 - y_5 + y_8 \le 0, \\
& 0 \le x_i \le 1,\ i = 1,\ 2,\ 3,\ 4,
\end{aligned} \qquad (13)$$

This problem containing 13 design variables and 9 constraints is from Floudas and Pardalos's book[13]. Two sets of solutions are given in Table 5.

Part (A) in Table 5 shows the exact solution of the problem and the solution by GOES. Apparently GOES didn't find the global solution of this problem. However, if the recombination formula used in GOES is changed from equation (3) to equation (4) or (H) in equation (1), GOES still can find the global solution shown in Part (B). Therefore the recombination formulas in evolution strategy may produce different results for different problems. Further researches on recombination formulas may be needed.

Table 5 Global solutions of problem 5
(A) (B)

Exact Solution[36]		ES	GOES	ES		GOES	
	GLOBAL		Eqn(3)	Eqn(B)	Eqn(4)	Eqn(B)	Eqn(4)
x_1	1	0.032	1	0.991	1	1	1
x_2	1	0.169	1	0.978	1	1	1
x_3	1	0.047	1	0.983	1	1	1
x_4	1	0	0	0.983	1	1	1
y_1	1	0.896	1	0.989	1	1	1
y_2	1	0.718	1	0.96	1	1	1
y_3	1	0.996	1	0.917	1	1	1
y_4	1	0.672	1	0.878	1	1	1
y_5	1	0.815	1	0.926	1	1	1
y_6	3	0.101	1	2.725	3	3	3
y_7	3	0.732	3	2.813	3	3	3
y_8	3	0.299	3	2.679	3	3	3
y_9	1	0.599	1	0.773	1	1	1
OBJ	-15	-4.75	-13	-13.3	-15	-15	-15
No.e	NA*	3500	3500	15000	15000	22500	22500
time(s)		<1	<1	1	1	1	1

Problem 6:

The formulation of the problem is given below.

$$Min \quad F(\bar{x}) = e^{x_1 x_2 x_3 x_4 x_5}$$

Subject to

$$h_1(x) \equiv x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 = 10,$$
$$h_2(x) \equiv x_2 x_3 - 5 x_4 x_5 = 0,$$
$$h_3(x) \equiv x_1^3 + x_2^3 = -1,$$
$$-2.3 \le x_i \le 2.3, \ i = 1, \ 2$$
$$-3.2 \le x_i \le 3.2, \ i = 3, \ 4, \ 5$$

(14)

This problem was provided by Hock and Schittkowsi[14]. Three recombination formulas (3), (4) and (B) are used to test GOES and the results are shown in Table 6. Part (A) in Table 6 contains the exact global solution and the solution from GOES by using recombination formula (3). Part (B) lists the results by using recombination formulas (B) and (4). It is seen that all three formulas find the global solution. The main difference between this problem and other test problems is it has three equality constraints. In general equality constraints are hard to satisfy. Therefore it is observed that at the end of ES search none of the three solutions are close to the global solution. But SQP search eventually manages to lead the way to the global solution. This example problem further proves that the integration of evolutionary computation with gradient-based search method can have a better chance to find the exact global solution.

Table 6 Global solutions of problem 6
(A) (B)

Exact Solution[14]		ES	GOES	ES		GOES	
	GLOBAL		Eqn(3)	Eqn(B)	Eqn(4)	Eqn (B)	Eqn(4)
x_1	2.3305	1.2775	2.3305	2.1571	2.331	-0.386	2.3303
x_2	1.9514	2.0609	1.9514	1.9399	1.951	1.8731	1.9514
x_3	-0.478	0.4073	-0.4778	-0.6348	-0.476	-0.744	-0.4786
x_4	4.3657	4.1129	4.3657	4.444	4.366	4.7045	4.3658
x_5	-0.625	-0.017	-0.6245	-0.6309	-0.624	-0.014	-0.6243
x_6	1.0381	0.0144	1.038	1.1218	1.038	0.655	1.0383
x_7	1.5942	0.9042	1.5942	1.4834	1.594	1.3584	1.5942
OBJ	680.63	705.58	680.63	681.18	680.6	725.28	680.63
No.e	NA*	4500	4500	15000	6800	15000	6800
time(s)		1	1	1	1	1	1

Problem 7: C-Bumpy function[9]

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

max.

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

subject to

$$\begin{aligned} &g_1(\bar{x}) \equiv x_1 x_2 > 0.75, \\ &g_2(\bar{x}) \equiv x_1 + x_2 \le 15, \\ &0 \le x_1 \le 10, \\ &0 \le x_2 \le 10, \end{aligned} \qquad (15)$$

Fig. 7 shows the global and local solutions of the problem. Table 7 gives the solutions obtained by various approaches. Again GOES yields better solution than solutions by other methods. The computational time is also the least one.

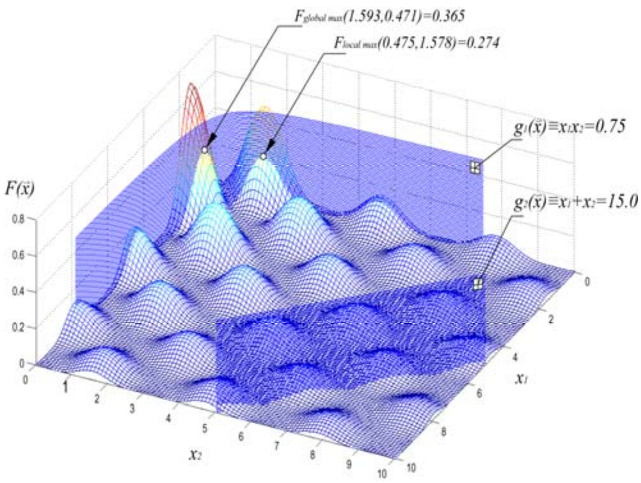

Fig. 7 C-Bumpy function

Table 7 Global solutions of *Bumpy function*

Exact Solution[9]		GOES	Lee[10]	DPF[10]	APF[10]
x_1	1.593	1.601	1.639	1.65	1.563
x_2	0.471	0.468	0.459	0.456	0.48
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 8: Himmeblau problem[15]

This constrained optimization problem having five design variables is defined as

min.

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

subject to

$$\begin{aligned} &g_1(\bar{x}) \equiv 85.334407 + 0.0056858 x_2 x_5 + 0.0006262 x_1 x_4 - 0.0022053 x_3 x_5 \\ &g_2(\bar{x}) \equiv 80.51249 + 0.0071317 x_2 x_5 + 0.0029955 x_1 x_2 + 0.0021813 x_3^{\ 2} \\ &g_3(\bar{x}) \equiv 9.300961 + 0.0047026 x_3 x_5 + 0.0012547 x_1 x_3 + 0.0019085 x_3 x_4 \\ &0 \le g_1(\bar{x}) \le 92,\ 90 \le g_2(\bar{x}) \le 110,\ 20 \le g_3(\bar{x}) \le 25 \\ &78 \le x_1 \le 102,\ 33 \le x_2 \le 45,\ 27 \le x_3 \le 45,\ 27 \le x_4 \le 45,\ 27 \le x_5 \le 45 \end{aligned}$$

(16)

The optimum solutions are listed in Table 8. The objective function value from Coello's solution is the smallest one. But one of the constraints is not satisfied, the solution is an infeasible solution. The best feasible solution is obtained by Homaifar. His approach used genetic algorithm with penalty function approach. The solution by GOES is the second best one and the result is very close to Homaifar's solution. The CPU time for GOES is also the least one in the known data.

Table 8 Global solutions of *Himmeblau function*

GOES		Lee[10]	DPF[10]	APF[10]	Homaifar[16]
x_1	78	79.293	82.681	79.473	78
x_2	33	34.186	34.502	34.163	33
x_3	29.995	31.186	31.573	31.576	29.995
x_4	45	39.92	40.07	43.267	45
x_5	36.776	36.195	33.78	33.86	36.776
OBJ	-30665.5	-30225.7	-30033.6	-30237.5	-30665.6
No.e	800	1650	5000	5000	NA*
time(s)	<1	138	2	2	NA*

Table 8(continued)

	Gen [17]	Himmelblau[15]	Coello[18]
x_1	81.49	78.62	78.05
x_2	34.09	33.44	33.007
x_3	31.24	31.07	27.081
x_4	42.2	44.18	45
x_5	34.37	35.22	44.94
OBJ	-30183.5	-30373.9	-31020.9
No.e	NA*	NA*	NA*
time(s)	NA*	NA*	NA*

6 Conclusion

The proposed global optimization algorithm GOES using hybrid approach of ES plus SQP has been proved to be successful in solving 8 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]I. Rechenberg, Evolutionsstrategie: Optimierung technisher systeme nach prinzipien der biologischen evolution, Frommann-Holzboog Verlag, Stuttgart, 1973.

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

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

[8] Z. Michalewicz, Genetic algorithms + Data structures = Evolution program, third edition, Springer-Verlag, 1996.

[9] A. J. Keane, Experiences with optimizers in structural design, Proc. Conf. Adaptive Computing in Engineering Design and Control, 1994, pp. 14-27,

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

[11] J. Branke and C.Schmidt, Faster convergence by means of fitness estimation, Soft Computing, Vol. 9, 2005, pp. 13-20.

[12]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.

[13] C. A. Floudas and P. M. Pardalos, Recent advances in global optimization, Princetion Series in Computer Science, Princeton University Press, Princeton, NJ, 1992.

[14] W. Hock and K. Schittkowsi, Test examples for nonlinear programming codes, Lecture Notes in Economics and Mathematical Systems, Vol. 187, Springer-Verlag, 1981.

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

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

[17] M. Gen and R. Cheng, A survey of penalty techniques in genetic algorithms, In Toshio Fukuda and Takeshi Furuhashi, editors, Proceedings of the 1996 International Conference on Evolutionary Computation IEEE, Nagoya, Japan, 1996, pp. 804-809.

[18] C. A. C. Coello , Self-adaptive penalties for GA-based optimization, Evolutionary Computation, Vol. 1, 1999, pp. 573-580.