KYBERNETIKA — VOLUME *46* (2010), NUMBER 4, PAGES 684–696

DIFFERENTIAL EVOLUTION ALGORITHM COMBINED WITH CHAOTIC PATTERN SEARCH

Yaoyao He, Jian Zhong Zhou, Ning Lu, Hui Qin and Youlin Lu

Differential evolution algorithm combined with chaotic pattern search(DE-CPS) for global optimization is introduced to improve the performance of simple DE algorithm. Pattern search algorithm using chaotic variables instead of random variables is used to accelerate the convergence of solving the objective value. Experiments on 6 benchmark problems, including morbid Rosenbrock function, show that the novel hybrid algorithm is effective for nonlinear optimization problems in high dimensional space. The comparisons with the standard particle swarm optimization (PSO), differential evolution (DE) and other hybrid algorithms verify DE-CPS algorithm has great superiority.

Keywords: hybrid algorithm, differential evolution(DE), chaotic pattern search, global optimization

Classification: 65K10, 49M37,90C30

1. INTRODUCTION

As a relatively new branch of system sciences and mathematics, chaos has provided us important tools for viewing the world we live in and been intensively applied in different fields of sciences, such as chaos control [19], synchronization [20], optimization research [14, 16] and so on. In general, chaos has three important dynamic properties [14]:

1. the sensitive dependence on initial conditions;
2. the quasi-stochastic property;
3. ergodicity.

In recent years, the applications of chaos in various disciplines including optimization research have attracted more and more attention. One way of application with chaos is chaotic optimization algorithm(COA) [14], which utilizes the nature of chaos sequence including the quasi-stochastic property and ergodicity. The experimental studies assert that the benefits by chaotic variables instead of random variables are more obvious although the mathematical theory can't be formulated. However, all of COAs exist two defects for global search. Firstly, it is necessary to choose appropriate initial value due to their sensitivity to the initial conditions. Secondly, simple COA is effective only in lower dimensional space. Owing to the

quasi-stochastic property of chaotic motion, the difference of computing results between two successive iterations is always big, which may result in the failure of global optimization in high dimensional space. Thus, several scholars try to combine other algorithms with chaos to enhance the performance of COA. The hybrid algorithms, such as chaotic genetic algorithm(CGA) [17], chaotic simulated annealing(CSA) [8], chaotic ant swarm optimization(CASO) [2], chaotic particle swarm optimization(CPSO) [5, 6, 15], and chaotic pattern search(CPS) [18] have been applied in different domains, respectively.

In this context, the literature proposes a novel hybrid algorithm for global optimization problem. The new method(DE-CPS) tried to combine differential evolution(DE) algorithm with chaotic pattern search(CPS) so as to achieve global optimum in high dimensional space. DE is employed for local search to optimize the objective function as the current optimum and CPS is used to enhance the optimization effect near the frontal optimal solution with a little chaotic perturbation.

The remainder of this paper is organized as follows. Section 2 describes the basic DE, while Section 3 introduces the pattern search method and proposes DE-CPS algorithm for global optimization. Numerical simulation and comparisons with PSO and DE are provided in Section 4. Finally, we outline the conclusion in Section 5 with a brief summary of results.

2. DIFFERENTIAL EVOLUTION(DE) ALGORITHM

Many global optimization problem can be formulated as the following form:

$$\min f(x),\ x = [x_1, \ldots, x_d],\ \text{s.t.}\ \ x_j \in [a_j, b_j],\ j = 1, \ldots, d. \tag{1}$$

where f is the objective function, and x is a continuous variable vector in d-dimensional space R^d. The feasible domain of variable x is defined by specifying upper (b_j) and lower (a_j) limits of each component j.

Differential evolution (DE) is one of the most recent population- based stochastic evolutionary optimization techniques. Storn and Price first proposed DE in 1995 [11] as a heuristic method for minimizing non-linear and non-differentiable continuous space functions. As other evolutionary algorithms, the first generation is initialized randomly and further generations evolve through the application of certain evolutionary operator until a stopping criterion is reached [12]. The theoretical framework of DE and its modified form [7, 9] are very simple to solve nonlinear continuous optimization problems, and have been applied to various scientific fields [3, 13]. The basic DE algorithm [11] can be depicted in more detail below with reference of following 5 steps.

Algorithm DE

Step 1: Let a population size N, crossover probability P_c, evolution generation T, scaling factor F, and initial evolution generation $t = 0$ in the d-dimensional space. Initialize the population with random vector $X(0) = (x_1(0), \ldots, x_N(0))$, in which the ith individual $x_i(0) = (x_{i1}(0), \ldots, x_{id}(0))$. The initial objective value is $f(x_b(0))$ in the optimal position $x_b(0) = (x_{b1}(0), \ldots, x_{bd}(0))$.

Step 2: Choose three distinct random integer r_1, r_2, r_3 from the set $\{1, \ldots, N\}$ and random integer $j_{ra} = 1, \ldots, d$. The new point is found from nearby the best position using the following crossover rule:

$$x'_{ij}(t) = \begin{cases} x_{r_1,j}(t) + F(x_{r_2,j}(t) - x_{r_3,j}(t)), & \text{if } r_{ij} < P_c \text{ or } j = j_{ra} \\ x_{ij}(t) & \text{else} \end{cases} \tag{2}$$

where r_{ij} are random numbers from (0,1) with the uniform distribution.

Step 3: Select each trial vector $x'_{ij}(t)$ for the $t+1$ iteration using the acceptance criterion:

$$x_{ij}(t+1) = \begin{cases} x'_{ij}(t), & \text{if } f(x'_{ij}(t)) < f(x_{ij}(t)) \\ x_{ij}(t) & \text{else.} \end{cases} \tag{3}$$

Step 4: Evaluate the objective function value $f(x_i(t))$ of each individual $x_i(t)$,in which the optimal individual of optimal value f_{best} is $x_b(t) = (x_{b1}(t), \ldots, x_{bd}(t))$.

Step 5: Update the current evolution generation $t = t + 1$. If $t = T$ is satisfied then stop, else go back to step 2.

In conventional DE, the fitness of an offspring competes one-to-one with its parent that gives rise to a faster convergence rate. However, this faster convergence also leads to a higher probability of obtaining a local optimum because the diversity of the population descends faster during the solution progress. Hence, the attempt that mixes other global optimal technique is a considerable field.

3. DE-CPS

3.1. Pattern search algorithm(PSA)

For a direct search method, pattern search algorithm(PSA) do not need the derivatives of objective functions [1, 4, 18]. Alternatively, only the function values are compared to choose the new iteration. Hence, we only need the order of function values. According to the nonlinear unconstrained global optimization problem, the steps of Hooke–Jeeves PSA [18] may be described as follows:

Algorithm PSA

Step 1: Set the initial evolution generation $t = 0$, initial vector $x(0) = (x_1(0), \ldots, x_d(0))$, axis direction vector $e = (e_1, \ldots, e_d)$, step size δ, acceleration factor $\alpha \geq 1$, compression factors $\beta \in (0, 1)$ and let $y = (y_1, \ldots, y_d) = x(t), j = 1$.

Step 2: If $f(y + \delta e_j) < f(y)$, let $y = y + \delta e_j$ and go to step 4; else go to step 3.

Step 3: If $f(y - \delta e_j) < f(y)$, let $y = y - \delta e_j$ else let $y = y$.

Step 4: If $j < d$, then $j = j + 1$ and go to step 2.

Step 5: Set $x(t+1) = y$, If $f(x(t+1)) < f(x(t))$, let $y = x(t+1) + \alpha(x(t+1) - x(t))$, then $\delta = \delta, t = t + 1, j = 1$ and go to step 2; else go to step 6.

Step 6: If $\delta \leq \varepsilon$, then stop; else let $\delta = \delta$, when $x(t+1) \neq x(t)$; let $\delta = \beta\delta$ when $x(t+1) = x(t)$, where the ε is a predefined constant according to the precision requirement.

Step 7: Let $y = x(t+1), t = t+1, j = 1$, and go to step 2.

Hooke–Jeeves PSA directly explores in feasible region along the potent direction. Although simplicity, the implementation of this algorithm is easy to trap in local optimization because of the influences of initial values and step size. In order to overcome this limitation, many researchers modified the PSA by adopting some techniques [4, 18].

3.2. Improved logistic chaotic map

The simple logistic map is a well-known chaotic map [14, 16, 17]. It is often cited as an example of how complex behavior can arise from a simple deterministic dynamic system without any stochastic disturbance. This map is written as:

$$y(k+1) = c \cdot y(k)(1 - y(k)) \quad \text{for } 0 < c \leq 4,\ y(k) \in (0, 1). \tag{4}$$

in which c is a control parameter and determines whether chaotic variable y stabilizes at a constant value. Setting $y(k) = (z(k) + 1)/2$ when $c = 4$, the equation (6) is changed to $z(k+1) = 1 - 2(z(k))^2$. Suppose a control parameter r, then we can define improved logistic map as follows:

$$z(k+1) = 1 - r(z(k))^2 \quad \text{for } 0 < r \leq 2,\ z(k) \in (-1, 1). \tag{5}$$

where the distribution of $z(k)$ with different r are depicted in Figure 1.

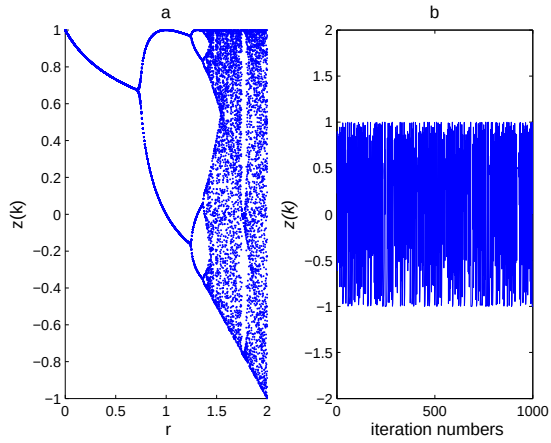

Fig. 1. The distribution of improved logistic map (a)$0 \leq r \leq 2$ (b) $r = 2$.

From Figure 1 (b), we can find improved logistic map can distribute in the region of $(-1, 1)$ except several unstable cycle points. According to this property, chaotic variables can be applied in PSA as axis direction vector. Hence, we will propose a DE based on PS using chaotic variable(DE-CPS) in the next subsection.

3.3. DE-CPS

In this subsection, DE and PS using chaotic variable, these two kinds of techniques, are both applied to numerical optimization as a way to accelerate convergence and improve the efficiency of the novel hybrid algorithms. In allusion to the problem of trapping in local optimization in the simple evolution method, we combine DE with CPS, and establish a new frame-based DE-CPS algorithm for comparison with conventional PSO [10] and DE as mentioned above. DE can propose a approximate global optimal point for PS using chaotic variable. After this, the process of CPS includes two major steps. The first one is chaotic exploration motion nearby the current optimal value with a little chaotic perturbation. The other one is the second order pattern motion includes coarse search and fine search. The procedure of DE-CPS is described by the following steps.

Algorithm DE-CPS

Step 1: Set the initial evolution generation $t = 0$, and initial all parameters of DE-CPS as the step 1 of basic DE in the Section 2.

Step 2: Execute the basic DE algorithm as a subprogram of the DE-CPS algorithm.

Step 2.1: Choose three distinct random integer r_1, r_2, r_3 from the set $\{1, \ldots, N\}$ and random integer $j_{ra} = 1, \ldots, d$. The new point is found from nearby the best position using the following crossover rule:

$$x'_{ij}(t) = \begin{cases} x_{r_1,j}(t) + F(x_{r_2,j}(t) - x_{r_3,j}(t)), & \text{if } r_{ij} < P_c \text{ or } j = j_{ra} \\ x_{ij}(t) & \text{else} \end{cases} \tag{6}$$

where r_{ij} are random numbers from (0,1) with the uniform distribution.

Step 2.2: Select each trial vector $x'_{ij}(t)$ for the $t+1$ iteration using the acceptance criterion:

$$x_{ij}(t+1) = \begin{cases} x'_{ij}(t), & \text{if } f(x'_{ij}(t)) < f(x_{ij}(t)) \\ x_{ij}(t) & \text{else.} \end{cases} \tag{7}$$

Step 2.3: Evaluate the objective function value $f(x_i(t))$ of each individual $x_i(t)$,in which the current optimal individual of optimal value f_{best} is $x_b(t)$.

Step 3: Chaotic exploration motion. Initial the chaotic iteration counter $k = 0$ and d-dimensional chaotic variable $z(0) = (z_1(0), \ldots, z_d(0))$, in which each $z_j(0) = \text{rand}(0, 1), (j = 1, \ldots, d)$. The maximum number of chaotic iteration is $C_{\max}$. Let $y(0) = (y_1(0), \ldots, y_d(0)) = x_b(t)$ and $x^* = y(0)$ for chaotic exploration motion.

Step 3.1: Run chaotic exploration motion using improved logistic map by means of equation (5).

Step 3.2: The chaotic variables, which are proportional to contraction factor ω, alter each component nearby the optimal position by following equation (addition and subtraction are operated, respectively.)

$$y_j(k+1) = y_j(k) \pm \omega\delta \cdot z_j(k) \tag{8}$$

where contraction factor $\omega = e^{\frac{-\gamma \cdot t}{T}}$ is a monotone decreasing function with the increase of evolution generation. γ is a predefined constant according to the precision requirement.

Step 3.3: Evaluate the value of $f(y(k+1))$ with twice. If the new result $f(y(k+1)) < f(y(k))$, then let $x^*(t) = y(k+1)$ and $f(x^*(t)) = f(y(k+1))$, else the values of $x^*(t)$ and $f(x^*(t))$ keep invariant.

Step 3.4: If $k = c_{\max}$ then go to step 4, else increase k by 1 and go to step 3.1.

Step 4: Second order pattern motion. The coarse/fine search strategy presented here is a parallel two-stage acquisition technique [18]. For improving the computing precision, a litter constant ε_1 is set to choose one of search strategy according to the effect of chaotic exploration. If $\|x^*(t) - x_b(t)\| \geq \varepsilon_1$, then adopt coarse search and go to step 5; otherwise, adopt fine search and go to step 6.

Step 5: (coarse search) Let $x'(t) = x^*(t) + \alpha_1(x^*(t) - x_b(t))$, $(\alpha_1 > 1)$, and evaluate the fitness value $f(x'(t))$. If $\min\big(f(x'(t))\big) < f(x^*(t))$, then replace the value of $x^*(t)$ with $x'(t)$.

Step 6: (fine search) Let $x'(t) = x^*(t) + \alpha_2(x^*(t) - x_b(t))$, $(0 < \alpha_2 \leq 1)$, and evaluate the fitness value $f(x'(t))$. If $\min\big(f(x'(t))\big) < f(x^*(t))$, then replace the value of $x^*(t)$ with $x'(t)$.

Step 7: To prevent the fitness value from trapping in local optima, a estimation criteria is adopted for identifying it, which is described as follows.

$$\triangle f(x^*) = \begin{cases} (f(x^*(t-1)) - f(x^*(t)))/f(x^*(t)), & \text{if } f(x^*(t)) \neq 0 \\ f(x^*(t-1)) - f(x^*(t)) & \text{else.} \end{cases} \tag{9}$$

If $\triangle f(x^*) < \varepsilon_2$ is satisfied for K times in succession, then reset the initial value of population and return to step 1 (namely population have trapped in local optima, and need to substitute it using new initial value), else directly go to step 8, where ε_2 is a little constant according to the precision requirement and K is a count constant.

Step 8: Let $t = t + 1$ and go back to step 2 until $t = T$ is satisfied. The value of $x^*(T)$ is the global solution that we need.

The process of DE-CPS algorithm includes three major steps, namely basic DE, chaotic exploration and second order pattern motion, in which DE is used to perform global exploration and CPS is employed to perform locally oriented search. The preliminary optimum $x_b(t)$ is obtained by basic DE, and it is viewed as the center with a little chaotic perturbation. Repeat the chaotic exploration until the maximum

chaotic iterative number $c_{\max}$ is satisfied, then the current optimum $x^*(t)$ is gained. Second order pattern motion is applied to decide to adopt a coarse search strategy or a fine search strategy according to the difference of $x_b(t)$ and $x^*(t)$. In fact, coarse search strategy is adopted more frequently in the early phase of hybrid algorithm due to keeping away from the global optimum; fine search strategy is adopted more frequently in the later stage for coming near to the global optimum. In order to avoid being trapped into local optimum, a estimation criteria is used to reset the initial population based on the required precision(namely chooses suitable ε_2).

4. NUMERICAL EXPERIMENT

4.1. Test functions and experimental setup

For evaluating the efficiency and effectiveness of the algorithm, six prevalent benchmark functions that are commonly employed in the natural computation literature [15],are used. f_1 and f_2 are unimodal functions (i. e., they have a unique local optimum that is also the global optimum) and the remaining four functions are multimodal (i. e., they have several local optima).

The function f_1 is the Sphere function:

$$\begin{aligned} &f_1(x) = \sum_{i=1}^{30} x_i^2 \\ &-100 \le x_i \le 100, \min(f(0,0,\ldots,0)) = 0. \end{aligned} \tag{10}$$

The function f_2 is the Rosenbrock function:

$$\begin{aligned} &f_2(x) = \sum_{i=1}^{29} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2] \\ &-30 \le x_i \le 30, \min(f(1,1,\ldots,1)) = 0. \end{aligned} \tag{11}$$

The function f_3 is the Ackley function:

$$\begin{aligned} &f_3(x) = -20\exp\Big(-0.2\sqrt{\tfrac{1}{n}\textstyle\sum_{i=1}^{30} x_i^2}\Big) - \exp\Big(\tfrac{1}{n}\sum_{i=1}^{30}\cos(2\pi x_i)\Big) + 20 + e \\ &-32 \le x_i \le 32, \min(f(0,0,\ldots,0)) = 0. \end{aligned} \tag{12}$$

The function f_4 is the Griewank function:

$$\begin{aligned} &f_4(x) = 1 + \sum_{i=1}^{30} \tfrac{x_i^2}{4000} - \prod_{i=1}^{30} \cos(\tfrac{x_i}{\sqrt{i}}) \\ &-400 \le x_i \le 400, \min(f(0,0,\ldots,0)) = 0. \end{aligned} \tag{13}$$

The function f_5 is the Rastrigrin function:

$$\begin{aligned} &f_5(x) = \sum_{i=1}^{30} (x_i^2 - 10\cos(2\pi x_i) + 10) \\ &-5.12 \le x_i \le 5.12, \min(f(0,0,\ldots,0)) = 0. \end{aligned} \tag{14}$$

The function f_6 is the Schaffer function:

$$f_6(x) = 0.5 + \frac{(\sin\sqrt{x_1^2+x_2^2})^2 - 0.5}{(1+0.001(x_1^2+x_2^2))^2} \\ -100 \le x_1, x_2 \le 100, \min(f(0,0)) = 0. \tag{15}$$

In our simulation, we first compared the DE-CPS with the stand PSO [10] and DE [11]. Each algorithm was implemented in Matlab 7.7.0(R2008b). All the programs were run on the Intel pentium T2370 processor with 1GB of random access momory. To PSO, the acceleration constant c_1, c_2 are set to 2.0, and the velocity is clamped to be 30% of search space, and the inertia weight decreases linearly from 0.9 to 0.4. In both DE and DE-CPS, the parameters were defined with crossover probability $P_c = 0.5$ and scaling factor $F = 0.5$. All the parameters of DE-CPS were set to $C_{\max} = 800$, $\delta = 0.01$, $\gamma = 20$, $\varepsilon_1 = e^{-10}$, $\varepsilon_2 = e^{-5}$, and $K = 5$.

4.2. Computational results

4.2.1. Fixed iterative number of evaluating results

Table 1. Computational results in 1000 iterations.

	alg	min	max	aver	std	CPU
f_1	PSO	7.1019e-008	2.1371e-005	3.6402e-006	4.7810e-006	4.1846
	DE	6.1231e-013	4.4680e-012	1.9459e-012	9.5136e-013	2.4853
	DE-CPS	3.1614e-021	5.1220e-021	4.1984e-021	5.0914e-022	10.037
f_2	PSO	15.856	256.66	70.901	51.896	3.6712
	DE	24.0997	25.3366	24.6876	0.3017	2.4694
	DE-CPS	2.3217e-015	3.1329e-015	2.8477e-015	1.5065e-016	10.32
f_3	PSO	5.5932e-005	1.9709e-003	6.6965e-004	5.0071e-004	5.165
	DE	2.4048e-007	5.4664e-007	3.7811e-007	7.2372e-008	4.1238
	DE-CPS	4.0576e-011	5.2087e-011	4.6711e-011	2.7963e-012	29.633
f_4	PSO	1.8794e-006	5.8951e-002	1.1465e-002	1.2884e-002	5.4563
	DE	1.4240e-012	7.7782e-011	9.5855e-012	1.1455e-011	4.7128
	DE-CPS	0	0	0	0	31.587
f_5	PSO	11.293	55.846	30.351	7.1162	3.9432
	DE	48.7046	80.2083	65.0621	7.3930	2.6632
	DE-CPS	8.0637	37.0862	17.2017	5.7842	17.543
f_6	PSO	0	0	0	0	2.8107
	DE	0	0.0097	0.0020	0.0039	1.5923
	DE-CPS	0	0	0	0	20.152

The Table 1 shows the simulation results of 50 independent runs including 6 benchmark functions. The population size of PSO and DE is 100, but DE-CPS's population size is 30 so as to reduce CPU time. The evolution generation of each algorithm is 1000. 'min', 'max', 'aver', 'std' and 'CPU' represent the minimum, maximum, average values, standard deviation of the proposed methods and the average CPU processing time of each independent run, respectively. It can be easily observed that the precision of DE-CPS outperforms PSO and DE to all problems,

and DE-CPS can be close to or achieve the global optima except function f_5. The std of DE is better than PSO except function f_6. Especially, It can also identify that DE-CPS has more obvious optimal performance in Rosenbrock function (f_2), simultaneously. So, the effect of hybrid algorithm is perfect for DE-CPS can search nearby the optimal value based on the results of simple DE. However, DE-CPS increases the CPU time for chaotic motion improves the of precision of algorithm, but reduces the evaluation speed of programs.

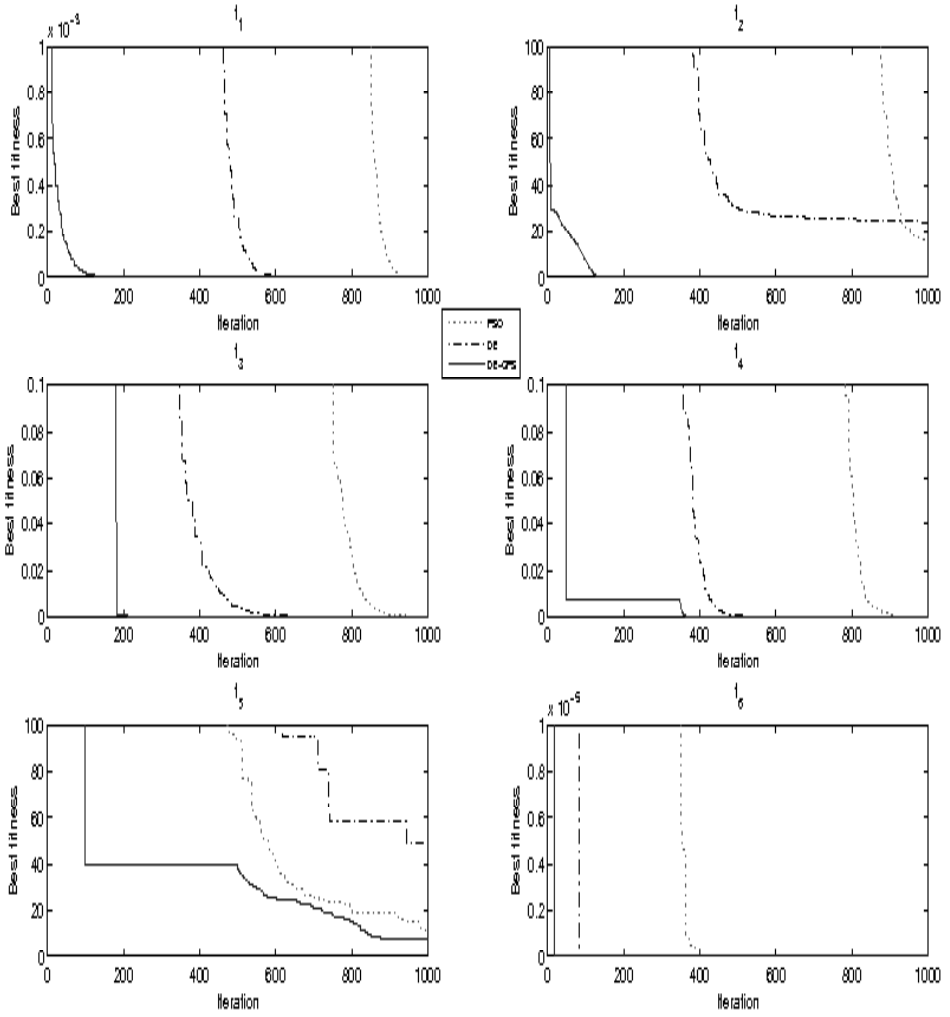

Fig. 2. The best results trend lines of PSO, DE and DE-CPS on different function.

Form Figure 2, it can be seen that the varying curves of optimal solutions using DE-CPS descend much faster than those by PSO and DE, and DE-CPS algorithm can gain best results compared with PSO and DE, simultaneously. Hence, it is concluded that DE-CPS is more efficient and accurate than PSO and DE.

4.2.2. Comparison with other hybrid algorithms

For demonstrating the superiority of DE-CPS algorithm, we try to compare it with other hybrid algorithms. In 2007, Xiang et. al proposed a particle swarm optimization using piecewise linear chaotic map(PWLCPSO), and compared the efficiency and robustness of five chaotic particle swarm optimization (CPSO)[15] according to the above-mentioned six test functions. In our study, the 'Goal' denotes the function value should be achieved in each run, and the success ratio γ which indicates the robustness of proposed hybrid algorithms is the ratio of times of successful run to total runs. Table 2 shows the goals of test functions that have to be achieved by all hybrid algorithms. In Table 3 and Table 4, it is easily identified that DE-CPS's efficiency and robustness outperforms other hybrid algorithms reported in the literature [15] on six benchmark functions and even obtains 100 % success ratio.

Table 2. The goals of test functions.

Function	f_1	f_2	f_3	f_4	f_5	f_6
Goal	0.01	100	0.1	0.1	100	10^{-5}

Table 3. Comparison in 5000 iterations($f_1 - f_3$).

	f_1		f_2		f_3	
	min	γ	min	γ	min	γ
rCPSO	368.32	0	49179	0	14.57	0
wCPSO	0.0019	0.96	328.32	0.28	8.57	0.46
CPSO'04	5.25e-077	1	13.003	1	0.32	0.74
CPSO'05	0.00078	0.98	131.2	0.72	1.3	0.32
PWLCPSO	4.049e-079	1	12.455	1	0.28	0.78
DE-CPS	1.3707e-109	1	9.2938e-16	1	4.4409e-015	1

Table 4. Comparison in 5000 iterations($f_4 - f_6$).

	f_4		f_5		f_6	
	min	γ	min	γ	min	γ
rCPSO	4.33	0	158.03	0.02	0.0058	0
wCPSO	0.031	0.96	43.01	1	7.5e-6	0.98
CPSO'04	0.042	1	11.8	1	0.0029	0.7
CPSO'05	0.059	0.86	56.39	0.96	0.0016	0.84
PWLCPSO	0.0075	1	47.28	1	0	1
DE-CPS	0	1	3.9798	1	0	1

4.2.3. Influence of dimension

Rosenbrock function(f_2) is a morbid in the feasible region and is very hard to be optimized for residing inside a long, narrow,and parabolic-shaped flat valley nearby the global optimum. General algorithm can't almost obtain a convergent result to it. Hence, most optimal algorithms set '100' as the goal of evaluation to it [15]. From the frontal context, it can be easily found that the evaluation of all algorithms can't be close to the minimal value '0' to 30-dimensional Rosenbrock function except DE-CPS. Thus, we try to evaluate the performance of DE-CPS in higher dimension space.

Table 5. DE-CPS to solve high dimensional Rosenbrock function in 1000 and 2000 iterations.

dim	iteration	min	max	aver	std	CPU	γ
50	1000	2.6173e-014	98.0032	4.3813	15.4557	13.5839	1
50	2000	9.0875e-005	41.6975	1.7802	6.0312	27.3907	1
80	1000	7.9657e-005	215.8463	35.5503	35.3754	15.6992	0.92
80	2000	9.3009e-0014	94.0201	5.9351	19.3441	31.6717	1
100	1000	21.2744	136.3613	58.892	14.8942	18.3851	0.96
100	2000	2.1799e-006	32.6063	26.1584	7.9107	36.575	1

Table 5 illustrates the evaluation results of DE-CPS to 50, 80, 100 dimensional Rosenbrock function in 1000 and 2000 iterations. It is easy to find that DE-CPS shows good effect and robustness to 50 dimensional problem. DE-CPS can achieve 100 % success ratio in 2000 iterations though it has general performance to the 80-dimensional and 100-dimensional problem in 1000 iterations. So, we can conclude that the increase of dimension may lead to the depreciation of performance of DE-CPS. But, we can solve this problem by means of increasing the evolution generations.

5. CONCLUSION

This paper has presented the differential evolution algorithm based on chaotic pattern search(DE-CPS). With the evolution of generations, pattern search using chaotic variable instead of random variable is incorporated into the basic DE algorithm to prevent local optimization. Experiment results shows that the proposed DE-CPS is superior in term of effectiveness and robustness including morbid Rosenbrock function. In the future, we will try to apply the DE-CPS for several complex engineering optimization problems.

ACKNOWLEDGEMENT

This work was partially supported by National Basic Research Program of China(No. 2007CB 714107),the National Natural Science Foundation (No.70631003),and the Special Research Foundation for the Public Welfare Industry of the Ministry of Science and Technology and the Ministry of Water Resources (No. 200701008).

(Received March 22, 2009)

REFERENCES

[1] C. Audet and J. E. Dennis: Pattern search algorithms for mixed variable programming. SIAM J. Optim. *11* (2001), 3, 573–594.

[2] J. J. Cai, X. Q. Ma, X. Li et. al: Chaotic ant swarm optimization to economic dispatch. Electron. Power Systems Research *77* (2007), 10, 1373–1380.

[3] H. Y. Fan and J. Lampinen: A trigonometric mutation operation to differential evolution. J. Global Optim. *27* (2003), 1, 105–129.

[4] W. E. HART: Evolutionary pattern search algorithms for unconstrained and linearly constrained optimization. IEEE Trans. Evol. Comput. *5* (2001), 4, 388–397.

[5] Y. Y. He, J. Z. Zhou, C. S. Li et. al: A precise chaotic particle swarm optimization algorithm based on improved tent map. ICNC *7* (2008), 569–573.

[6] Y. Y. He, J. Z. Zhou, X. Q. Xiang et. al: Comparison of different chaotic maps in particle swarm optimization algorithm for long term cascaded hydroelectric system scheduling. Chaos Solitons Fractals *42* (2009), 5, 3169–3176.

[7] Y. Y. He, J. Z. Zhou, H. Qin et. al: Flood disaster classification based on fuzzy clustering iterative model and modified differential evolution algorithm. FSKD *3* (2009), 85–89.

[8] M. J. Ji and H. W. Tang: Application of chaos in simulated annealing. optimization. Chaos Solitons Fractals *21* (2004), 933–941.

[9] P. Kaelo and M. M. Ali: A numerical study of some modified differential evolution algorithms. European J. Oper. Res. *169* (2006), 1176–1184.

[10] J. Kennedy and R. J. Eberhan: Particle swarm optimization. In: IEEE Internat. Conf on Neural Networks 1995, Vol. 4, pp. 1942–1948.

[11] R. Storn and K Price: Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. Technical Report TR-95-012, International Computer Science Institute, Berkeley 1995.

[12] R. Storn and K. Price: Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. *11* (1997), 341–359.

[13] R. Storn and K. Price: Differential evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. University of California, Berkeley 2006.

[14] M. S. Tavazoei and M. Haeri: Comparison of different one-dimensional maps as chaotic search pattern in chaos optimization algorithms. Appl. Math. Comput. *187* (2007), 1076–1085.

[15] T. Xiang, X. F. Liao, and K. W. Wong: An improved particle swarm optimization algorithm combined with piecewise linear chaotic map. Appl. Math. Comput. *190* (2007), 1637–1645.

[16] D. X. Yang, G. Li, and G. D. Cheng: On the efficiency of chaos optimization algorithms for global optimization. Chaos Solitons Fractals *34* (2007), 1366–1375.

[17] X. H. Yuan, Y. B. Yuan, and Y. C. Zhang: A hybrid chaotic genetic algorithm for short-term hydro system scheduling. Math. Comput. Simul. *59* (2002), 4, 319–327.

[18] X. F. Yuan, Y. N. Wang, and L. H. Wu: Pattern search algorithm using chaos and its application. J. of Hunan University (Natural Sciences) *34* (2007), 9, 30-33.

[19] L. Zhang and C. J. Zhang: Hopf bifurcation analysis of some hyperchaotic systems with time-delay controllers. Kybernetika *44* (2008), 1, 35–42.

[20] Z. L. Zhu, S. P. Li, and H. Yu: A new approach to generalized chaos synchronization based on the stability of the error System. Kybernetika *44* (2008), 4, 492–500.

Yaoyao He, School of Management, Hefei University of Technology,Hefei 230009, China, School of Hydropower and Information Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, 430074. China.
e-mail: hy-342501y@163.com

Jianzhong Zhou, School of Hydropower and Information Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, 430074. China.
e-mail: jz.zhou@hust.edu.cn

Ning Lu, School of Automation, Wuhan University of Technology, Wuhan, Hubei, School of Hydropower and Information Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, 430074. China.
e-mail: luning3000@163.com

Hui Qin, School of Hydropower and Information Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, 430074. China.
e-mail: hqin.hust@gmail.com

Youlin Lu, School of Hydropower and Information Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, 430074. China.
e-mail: youlin_lu@yahoo.cn