Article ID:0253-2778(2009)08-0885-09

An invasive weed optimization algorithm for constrained engineering design problems

SU Shou-bao[1,2], WANG Ji-wen[2], ZHANG Ling[2], FANG Jie[1,2], LI Fu-pen[2]

(1. *Department of Computer Science & Technology, West Anhui University, Lu'an 237012, China*;
2. *Ministry of Education Key Lab of Intelligent Computing & Signal Processing, Anhui University, Hefei 230039, China*)

Abstract: A novel swarm intelligence optimization technique for constrained problems was presented. The algorithm was inspired from colonizing weeds, which is used to mimic the natural behavior of weeds in colonizing and occupying suitable places for growth and reproduction. It has the robustness, adaptation and randomness and is simple but effective with an accurate global search ability. Some applications of the new algorithm on constrained engineering design optimization via employing a penalty approach suggest that the experimental results from the proposed algorithm are promising. Also, experimental applications and comparisons show that the presented algorithm is a potential global search technique for solving complex engineering design optimization problems. Extensive simulations are conducted along with statistical tests to yield helpful conclusions regarding the effects of parameter settings on the algorithm's performance.

Key words: global optimization; invasive weed optimization; constrained design optimization; penalty function approach; intelligent optimization

CLC number: TP18; TB237 **Document code**: A

一种约束工程设计问题的入侵性杂草优化算法

苏守宝[1,2],汪继文[2],张 铃[2],方 杰[1,2],李付鹏[2]

(1.皖西学院计算机科学与技术系,安徽六安 237012;2.安徽大学计算智能与信号处理教育部重点实验室,安徽合肥 230039)

摘要:提出了一种新颖的求解约束问题的群智能优化算法.该算法模拟杂草克隆、占地生长与繁殖的自然行为,具有入侵性杂草的鲁棒性、适应性和随机性等特点,算法简单而有效,具有准确的全局搜索能力.结合罚函数方法将提出的算法应用于求解工程设计优化问题,实验结果及比较表明提出的算法获得了更优的结果,同时也显示了它在求解复杂工程设计优化问题时的全局寻优能力.进一步实验与统计分析了关于参数选择对算法性能的影响,得到了有利参数选择的结论.

关键词:全局优化;入侵性杂草优化;约束设计优化;罚函数方法;智能优化

Received: 2007-12-11; **Revised**: 2008-01-21

Foundation item: Supported partly by the Key Project of Provincial Natural Scientific Research Fund from the Educational Bureau of Anhui Province of China (KJ2007A087), National Natural Science Foundation of China (60475017), National Basic Research (973) Program of China (2004CB318108), Natural Science Foundation of Anhui Province of China (090412045, 090412261X).

Biography: SU Shou-bao, male, born in 1965, PhD candidate/associate professor. Research field: Swarm intelligence and control optimization. E-mail: sooshoubo@wxc.edu.cn

Corresponding author: WANG Ji-wen, PhD/Professor. E-mail: wjw@ahu.edu.cn

0 Introduction

Constrained optimization problems are interesting because they arise from naturally engineering, science, operation research, etc. Many practical engineering applications, such as structural optimization, engineering design, automobile cab layout, etc, lead to a paradigm of global optimization[1]. Due to the nature of these applications, the feasible solutions usually need to be constrained in a small subset of the search space that is delimited by linear and/or nonlinear constraints. To tackle such problems different mathematical deterministic as well as stochastic algorithms have been developed. Deterministic approaches such as Feasible Direction and Generalized Gradient Descent make strong assumptions on the continuity and differentiability of the objective function. Therefore, their applicability is limited since these two characteristics are rarely satisfied in problems that arise in real life applications[2,3]. On the other hand, in recent years there have been an extensive research conducted for studying different natural computational methods like genetic algorithms (GAs), ant colony optimization (ACO), particle swarm optimization (PSO), and so on[4~6], which are bio-inspired from some natural systems in physics and biology. These numerical optimization algorithms are categorized with non-gradient based direct search algorithms. The main advantage of these algorithms is that they only use the objective function and constrain values to steer towards the optimal solution. Equipped with numerical stochastic searching ability, they have been successfully applied in many areas and also proved to be a very effective approach in the global optimization for tackling constrained optimization problems.

During the past few years, some new ecology-inspired computational models are constantly developed. Recently, a novel global optimization model, invasive weed optimization (IWO) algorithm, was proposed by Mehrabian and Lucas[7,8] in dynamic and control systems theory, which is inspired from a common phenomenon in agriculture: colonization of invasive weeds. The algorithm has some distinctive properties in comparison with traditional GAs and other numerical search algorithms like reproduction, spatial dispersal, and competitive exclusion, but no genetic operators are employed in the IWO algorithm, which makes it more dissimilar to GAs. It is simple but effective in converging to optimal solution by capturing basic properties, e. g. seeding, growth and competition, in a weed colony. The study and application[7~9] have demonstrated that IWO can outperform other numerical stochastic optimization algorithms.

In this paper, a new population-based evolutionary algorithm is presented to incorporate the penalty function into the IWO to solve constrained optimization problems. With no penalty factors to choose from, the algorithm is a simple but effective optimizing method with accurate global search ability, thus extending the invasive weed optimization method to real-world engineering design applications. The performance of the proposed algorithm on some well-known constrained engineering optimization problems is investigated via employing a penalty function approach for constraint-handling, and comparisons with previously reported results are presented. Extensive simulations are conducted along with statistical tests to yield useful conclusions regarding the effects of the parameter setting on the algorithm's performance.

1 Invasive weed optimization algorithm

1.1 Simulating weed colonizing behavior

To model and simulate the colonizing behavior of weeds for introducing a novel optimization algorithm, some basic properties of the process need to be considered, which are initializing population, reproduction, spatial dispersal and competitive exclusion.

1.1.1 Initializing population

A population of initial solutions is dispread over the d-dimensional search space with random positions.

1.1.2 Reproduction

Any individual of the population of weed seeds is allowed to produce seeds according to its own colony's fitness, the lowest fitness and the highest fitness to make sure the increase is linear, as shown in Fig. 1.

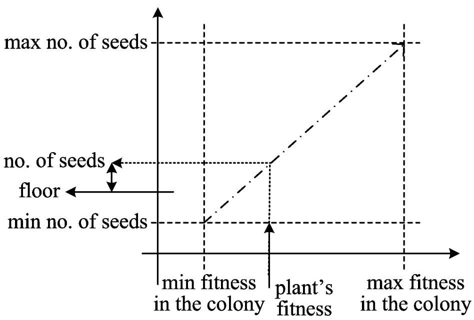

Fig. 1 Seed production procedure in a colony of weeds

1.1.3 Spatial dispersal

The generated seeds are randomly distributed over the d-dimensional search space by normally distributed random numbers with a mean equal to zero; but with a varying variance. This ensures that seeds will be randomly distributed such that they abide near to the parent plant. However, standard deviation (SD), σ, of the random function will be reduced from a previously defined initial value, σ_{initial}, to a final value, σ_{final}, in every generation. In simulations, a nonlinear alteration has shown satisfactory performance, given as follows

$$\sigma_i = \frac{(\text{iter}_{\max} - i)^n}{(\text{iter}_{\max})^n}(\sigma_{\text{initial}} - \sigma_{\text{final}}) + \sigma_{\text{final}} \quad (1)$$

where $\text{iter}_{n\max}$ is the maximum number of iterations, σ_{iter} is the SD at the present time step and n is the nonlinear modulation index. Obviously, when n is set to 3, SD is linearly decreased from the maximum to the minimum.

1.1.4 Competitive exclusion

There is a need for some kind of competition between plants due to the limiting maximum number of plants in a colony. After passing some iterations, the number of plants in a colony will reach its maximum by fast reproduction. However, it is hoped that the fitter plants reproduce better than undesirable plants. When the maximum number of plants in the colony is reached, $p_{\max}$, a mechanism for eliminating the plants with poor fitness in the generation, is activated. This mechanism works as follows: when the maximum number of weeds in a colony is reached, each weed is allowed to produce seeds. The produced seeds are then allowed to spread over the search area. When all seeds have found their position in the search area, they are ranked together with their parents' (as a colony of weeds). Next, weeds with lower fitness are eliminated to reach the maximum allowable population in a colony. In this way, plants and their offspring are ranked together and the ones with better fitness survive and are allowed to replicate. The population control mechanism also is applied to their offspring to the end of a given run, realizing competitive exclusion.

1.2 Invasive weed optimization algorithm

Invasive weeds have shown to be very robust and adaptive to changes in the environment. Thus, capturing their properties would lead to a powerful optimization algorithm, which is a numerical stochastic, population-based search algorithm. It attempts to mimic natural behavior of weed colonizing in opportunity spaces for function optimization. The procedure of the algorithm can be summarized as follows:

// Procedure of IWO algorithm

Step 1 A finite number of seeds are spread out over the search space;

Step 2 Every seed grows to flowering plants and produces seeds depending on its fitness;

Step 3 The produced seeds are randomly dispersed over the search area and grow to new plants;

Step 4 This process continues until the maximum number of iterations of plants is reached; now only the plants with lower fitness can survive and produce seeds, others are eliminated.

Step 5 The process continues until the maximum number of iterations is reached and hopefully the plant with the best fitness is closest to the optimal solution.

2 Problem formulations and constraint handling

Generally there are several types of constraint handling methods, such as methods that preserve solution feasibility, penalty-based methods, methods that clearly distinguish between feasible and unfeasible solutions, and hybrid methods. As we all know, the penalty function method has been the most popular constraint-handling technique due to its simple principle and easy implementation. A general engineering optimization problem can be described as follows

$$(\mathrm{P}) \qquad \min_{\boldsymbol{X}\in S\subset \mathrm{R}^n} f(x) \tag{2}$$

$$\text{s. t.} \quad \left.\begin{aligned} & g_i(x) \leqslant 0,\ i=1,\cdots,m \\ & S=\{x_1,x_2,\cdots,x_n\} \xrightarrow{f} \mathrm{R}^1 \\ & x_i^l \leqslant x_i \leqslant x_i^u,\ i=1,2,\cdots,n \end{aligned}\right\} \tag{3}$$

where m is the number of constraints. Different inequality and equality constraints can be easily transformed into the form of Eq. (3). Constraints define the feasible region, meaning that if the vector $\boldsymbol{X}$ complies with all constraints g_i, then it belongs to the feasible region. Traditional methods relying on calculus demand that the functions and constraints have very particular characteristics; those based on evolutionary algorithms don't have such limitations. For this reason, many constraint handling strategies have been proposed[5,10~12]. The most popular approach to constrained optimization is the application of penalty functions. In this approach, a constrained problem is transformed into a non-constrained one. The function under consideration is transformed as follows

$$F(x) = f(x) + \text{penalty}(x) \tag{4}$$

The function penalty(x) is a penalty function denoted by $P(x)$, which has been tackled in different strategies. Penalty functions with static, dynamic, annealing and adaptive penalties have been proposed and successfully applied in different applications. From the experiments and analysis of Kuri[10], the following penalty function method is the simplest but has the best performances among all the methods considered and here it is denoted by

$$P(x) = \begin{cases} \left[K - \sum_{i=1}^{s} \frac{K}{p}\right] - f(x),\ s \neq p \\ 0,\ \text{otherwise} \end{cases} \tag{5}$$

where K is a large constant $[O(10^9)]$, p is the number of constraints and s is the number of those which have been satisfied. K's only restriction is that it should be large enough to insure that any non-feasible individual is graded much more poorly than any feasible one. Here the algorithm receives information as to how many constraints have been satisfied but is not otherwise affected by the strategy. Notice, however, that in this method the penalty is not added to $f(x)$ as in Eq. (2) but, rather, it replaces $f(x)$

$$F(x) = K - \sum_{i=1}^{s} \frac{K}{p} \tag{6}$$

when any of the constraints is not met.

In the current study, we employed the penalty function mentioned above, of which selection is based on the promising results obtained by using such penalty functions with evolutionary algorithms.

3 The proposed algorithm for constrained engineering design optimization problems using IWO with penalty function (CDIWO)

The original IWO algorithm is not a natural constrained optimizer but it does provide a new paradigm for our development of a robust, optimization tool for constrained problems. In this section, we employ the penalty function approach into the IWO algorithm for constrained design optimization problems, which is denoted by CDIWO algorithm. It is a universal method for constrained optimization problems because of no penalty factor to choose. The algorithm starts with an initial population of weeds in a certain initialization area, and then reproduction shows us the process of colonizing weeds around the point with best fitness value for a given problem function. For each individual of weed population, count the number of constraints which have been satisfied by evaluating $g_i(x)$. When the number of weeds in the population does not equal the given maximum number of plant population, the weed population is sorted in the order of better fitness, which is a lower objective function value for a given problem. In the population the first 5 individuals produce 3 seeds, the second 5 individuals produce 2 seeds, the third 5 individuals produce 1 seed and the other individuals produce no seeds. The seeds are generated with a normal (Gaussian) distribution function with standard deviation, σ and mean, μ, where μ is the constant of parent plant initialized by an experiential value for the given problem. In each iteration, the standard deviation has been calculated with the above nonlinear Eq. (1). In this process, the plants grow towards the optimal point from the initialization area, in which plants with lower (worse) fitness are excluded, and only weeds with higher (better) fitness are allowed to reproduce, which leads approximately to the optimal point. The final value of the fitness function is found when the specified maximum generation is reached. The pseudo code for the CDIWO algorithm is given as follows:

```
// CDIWO algorithm
Setup initial parameter values listed in Tab. 1;
Generate randomly a population of N0 weeds:
  wi=(wi1,wi2,…,wid), i=1,…,N0, j=1,…,dim;
For i = 1 to itermax
  if calculating gi(x)≤0, count s in Eq. (5);
  Evaluate F(wi) according to Eqs. (4~6);
  Update σi by Eq. (1);
  For each individual wi∈N0;
    Compute number of seeds of wi, corresponding to its
       fitness, s. t. [smax, smin];
    Generate seeds over the search space with N(0, σi)
       around the parent plant wi;
    Add the generated seeds into the solution set, N0;
  End;
  If N0>Pmax;
     Sort the population N0 in descending order of smaller
       F(wi);
     Truncate population of weeds with higher fitness until
       N0 = Pmax;
  End If;
Next i, until a stop criterion is satisfied or a maximum
number of iterations achieved
```

Tab. 1 Notations and definitions of initial parameters for the CDIWO algorithm

notation	definition
N_0	number of initial population of weeds
$\text{iter}_{\max}$	maximum number of iterations
dim	problem dimension
p	number of constraints
$p_{\max}$	maximum number of plant population
$[s_{\max}, s_{\min}]$	maximum/minimum number of seeds
n	nonlinear modulation index
σ_{initial}	initial value of standard deviation
σ_{final}	final value of standard deviation
x_{initial}	initial search area

4 Applications and analysis

4.1 Three engineering design optimization problems

Problem 4.1 Design of a tension spring. This problem consists of the minimization of the weight of the tension/compression spring taken from Ref. [11], subject to constraints on the minimum deflection, shear stress, surge frequency, diameter and design variables. The design variables are the wire diameter, $d(x_1)$, the mean coil diameter, $D(x_2)$, and the number of active coils, $N(x_3)$. The problem is mathematically formulated as

$$\min_{\boldsymbol{X}} f(x) = x_2(x_3+2)x_1^2$$

$$\text{s. t.} \quad g_1(x) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leqslant 0$$

$$g_2(x) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \leqslant 0$$

$$g_3(x) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leqslant 0$$

$$g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leqslant 0$$

where $\boldsymbol{X}=(x_1, x_2, x_3)^{\mathrm{T}}$. The desired ranges of the design variables are

$$0.05 \leqslant x_1 \leqslant 2.0,\ 0.25 \leqslant x_2 \leqslant 1.3,$$
$$2.0 \leqslant x_3 \leqslant 15.0$$

Problem 4.2 The welded beam design

problem is taken from Ref. [12], in which a welded beam is designed for minimum cost subject to constraints on shear stress τ, bending stress in the beam θ, buckling load on the bar P_c, end deflection of the beam δ, and side constraints. There are four design variables, $h(x_1)$, $l(x_2)$, $t(x_3)$, and $b(x_4)$. The problem can be mathematically formulated as

$$\min_{\boldsymbol{x}} f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$$

$$\text{s. t.} \quad g_1(x) = \tau(x) - 136000 \leqslant 0$$

$$g_2(x) = \sigma(x) - 30000 \leqslant 0$$

$$g_3(x) = x_1 - x_4 \leqslant 0$$

$$g_4(x) = 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leqslant 0$$

$$g_5(x) = 0.125 - x_1 \leqslant 0$$

$$g_6(x) = \delta(x) - 0.25 \leqslant 0$$

$$g_7(x) = 6000 - P_c(x) \leqslant 0$$

where

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{6000}{\sqrt{2}x_1x_2},\ \tau'' = \frac{MR}{J}$$

$$M = 6000\left(14 + \frac{x_2}{2}\right),\ R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_2}{2}\right)^2}$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$

$$\sigma(x) = \frac{784000}{x_4x_3^2},\ \delta(x) = \frac{4 \times 6000 \times 14^3}{30 \times 10^6 x_3^3 x_4}$$

$$P_c(x) = \frac{2.0065\sqrt{x_3^2x_4^6}}{14^2}\left(1 - \frac{50x_3}{28}\right)$$

and the desired ranges of the design variables are

$$0.1 \leqslant x_1, x_4 \leqslant 2.0,\ 0.1 \leqslant x_2,\ x_3 \leqslant 10.0$$

4.2 Parameters setting

From the experiences and analysis of Mehrabian and Lucas[7], in a colony with population of 10 to 20 weeds, and n is set to 3, the numbers of seeds are set to 3 and 0, the convergence and the performance are better. Parameters of this algorithm are summarized as Tab. 2.

Tab. 2 Parameters setup for two problems

notation	initial values of	
	tension spring	welded beam
N_0	20	20
$\text{iter}_{\max}$	500	500
dim	3	4
$p_{\max}$	10	10
$[s_{\max}, s_{\min}]$	[3,0]	[3,0]
n	3	3
σ_{initial}	6.5	6.5
σ_{final}	0.001	0.001
x_{initial}	$x_1 \in [0.05, 2.0]$	$x_1 \in [0.1, 2]$
	$x_2 \in [0.25, 1.3]$	$x_2 \in [0.1, 10]$
	$x_3 \in [2.0, 15.0]$	$x_3 \in [0.1, 10]$
		$x_4 \in [0.1, 2]$

4.3 Numerical results

In our experiment , the proposed algorithm was allowed to perform 50 runs independently for three problems, and the best solutions are found, of which computational results are reported in Tab. 3, respectively. Some of the important criteria of heuristic search techniques need to be discussed for solving the constrained engineering design optimization problems. For this purpose, some statistical results about the mean, standard deviation (SD), best and worst solutions obtained in 50 independent runs for each problem are recorded in Tab. 4.

Tab. 3 The best solutions for two problems obtained from CDIWO

variables	tension spring	welded beam
x_1	0.051 727 8	0.203 135 7
x_2	0.357 643 7	3.542 989 1
x_3	11.244 539 6	9.033 488 3
x_4		0.206 181 2
$g_1(x)$	−8.378 519 1E−4	−1.224 444 5E+5
$g_2(x)$	−1.510 473 9 E−5	−44.918 833 7
$g_3(x)$	−80.693 486 3	−3.045 500 0E−3
$g_4(x)$	−0.727 085 7	−3.423 711 9
$g_5(x)$		−0.078 135 7
$g_6(x)$		−0.235 556 9
$g_7(x)$		−38.217 289 7
$f(x)$	1.267 463 5E−2	1.733 473 9

These problems have already been solved by several researchers, including Parsopoulos[11] and Hu et al.[5], etc. In order to compare our method with those mentioned above in terms of performance on engineering optimization problems, we have taken their solutions and listed them in Tab . 4 . There are

Tab. 4　Statistical results and comparisons

problems		CDIWO	PSO	UPSO	GA	ACH
tension spring	mean	1.273 128 4E−2	4.673 51E−2	2.294 78E−2	1.274 20E−2	1.088 285E−2
	SD	2.647 231 6E−4	2.145 05E−1	7.205 71E−3	5.900 00E−5	1.027 939E−3
	best	1.267 463 5E−2	1.281 58E−2	1.312 00E−2	1.268 10E−2	9.872 521E−3
	worst	1.302 581 4E−2	1.579 98	5.036 51E−2	1.297 30E−2	1.378 785E−2
welded beam	mean	1.776 146 3	1.968 20	2.837 21	1.792 654	N/A
	SD	0.081 307 2	1.554 15E−1	6.829 80E+1	0.074 713	
	best	1.733 473 9	1.765 58	1.921 99	1.728 226	
	worst	1.892 713 6	2.844 06	4.883 60	1.993 408	

solutions by different methods, including genetic algorithm(GA)[2], original PSO, united PSO[5] and ant colony algorithm(ACH)[6].

4.4 Discussions

Tab. 3 shows that the "best" feasible solutions are found by our proposed algorithm, where the positions are (0.051 727 8, 0.357 643 7, 11.244 539 6) on the spring problem, (0.203 135 7, 3.542 989 1, 9.033 488 3, 0.206 181 2) on the welded beam problem, respectively. From Tab. 4, it can be seen that the best feasible solutions found by IWO is better than the best solutions obtained from other methods. It has the overall best performance with respect to the mean objective function value of the best solutions as well as the smaller standard deviation.

To gain a deeper understanding of the performances of our proposed algorithm on constrained engineering design optimization problems, we compare the box plots of five methods on three problems as shown in Figs. 2～3 respectively. The box-plots provided a clearer and more detailed overview of all the experimental data. For the spring problem shown in Fig. 2, although the box and whisker of IWO is higher than that of GA, the box plot shows that the IWO found slightly a better "best" solution. By examining the box plot, we can observe that for other problems, the boxes and whiskers of IWO are lower than those of other methods. This means the proposed approach worked very robustly and effectively on these problems. For the problems, the box plots show that two methods, IWO and GA, had very close performances. Also the box plots show that the average searching quality of IWO is also better than those of other approaches, and even the worst solution found by IWO is better than the best solution obtained from original PSO and the best solution found by UPSO. Moreover, the standard deviations of the result by our proposed algorithm for two problems in 50 independent runs are very small.

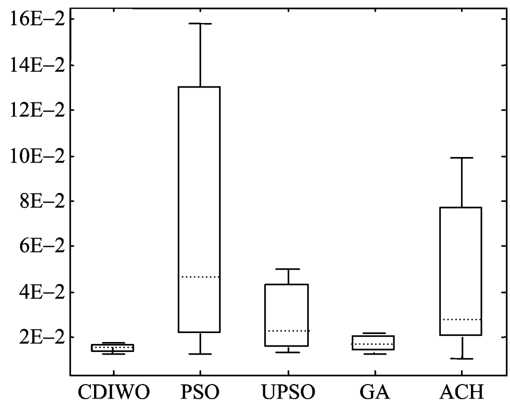

Fig. 2　Box plot for tension spring

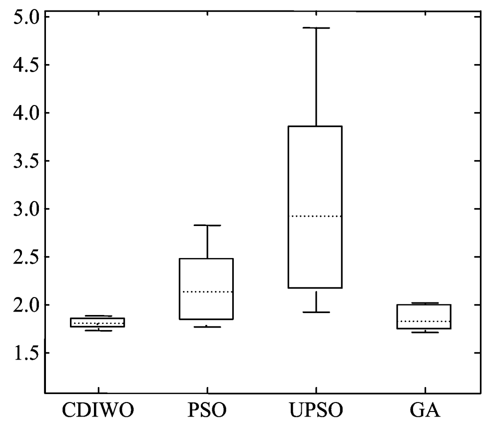

Fig. 3　Box plot for welded beam

Based on the above experimental results and comparisons, our proposed method proved to be one of the most promising methods. It can be concluded that IWO is of superior searching quality and robustness for constrained engineering design optimization problems.

4.5 Effects of parameter setting

In unconstrained functions, original IWO algorithm has been well tested with different classes of benchmark functions. In order to take full advantage of the CDIWO, it is necessary to understand the impact on the performances of the proposed algorithm with different parameter tunings. Hence, the tension spring problem is investigated as a test benchmark additionally.

Based on Columns 7 in Tab. 2, for each case with different initial sizes (10,20,30,40,50,60) and different nonlinear modulation indices (1,2,3,4) shown in Fig. 4, 50 independent traits were conducted for CDIWO algorithm, and the statistical results were recorded in Tab. 5. The performance is considered according to the following two criteria:

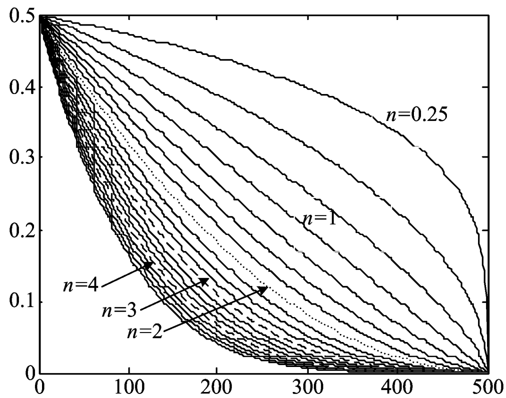

Fig. 4 Varying SDs. with $iter_{max} = 500$, $\sigma_{initial} = 0.5$, $\sigma_{final} = 0.01$, **under many nonlinear modulation index** n

Tab. 5 Averages for spring problem

initial weeds	nonlinear modulation index (n)			
	1	2	3	4
10	2.283 1E−2	1.562 2E−2	1.318 2E−2	1.692 2E−2
20	2.031 6E−2	1.313E−2	1.273 3E−2	1.493 3E−2
30	1.974 3E−2	1.296 8E−2	1.273 7E−2	1.275 6E−2
40	2.261 1E−2	1.276 4E−2	1.282 6E−2	2.211 7E−2
50	1.968 7E−2	1.566 7E−2	1.296 7E−2	1.923 2E−2
60	2.179 8E−2	1.778 4E−2	1.374 2E−2	1.368 8E−2

(Ⅰ) The success rate, as represented by the number of traits required for the colonizing weeds to hit its feasible regions.

(Ⅱ) The average of the solution obtained from all runs. In all cases, the solution stopped when a maximum number of 500 iterations were reached.

From experiments it can be found that increasing the iterations leads to a lower mean for solution; however, basically, it does not increase the success rate, which is increased by decreasing population size of weed in a colony. From Tab. 5 and Fig. 5, it can be seen that the proposed algorithm is guaranteed to find feasible solutions in all cases, but no all optimum. A colony with a population of 20 to 30 weeds has shown satisfactory means. Hence, increasing the initial size of weeds does not essentially lead to satisfactory results (lower mean or higher success rate). Rather, it would require a large number of objective and constraint function evaluations. Also, it is observed that nonlinear modulation index has a key control on the performance of the proposed algorithm. It makes the weed colony change its behavior in time and softly switch from a high value of standard deviation to a lower one as shown in Fig. 4, which results in grouping fitter plants and elimination of inappropriate plants, representing transformation from some selection mechanism. In our simulation, when a suitable value for the nonlinear modulation index, n, is found equal to 3, much lower means are obtained from CDIWO algorithm.

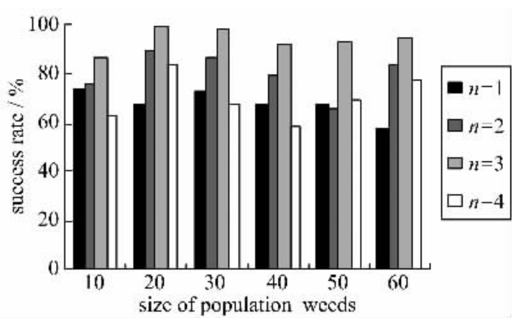

Fig. 5 Chart of the success rates for spring problem

Choice of the parameters for the implemented algorithm has guaranteed convergence to a feasible solution in the search space confirmed by numerical experiences. In addition, effects of other parameter setting on the behavior of colony, and the theoretical convergence will be reported by the authors in another article.

5 Conclusion

This paper has introduced an improved invasive weed optimization algorithm and reported for the first time to incorporate the penalty function into IWO to solve constrained engineering optimization problems. The algorithm is inspired from colonizing weeds, and is used to mimic natural behavior of weeds in colonizing and occupying suitable places for growth and reproduction. It has the robustness, adaptation and randomness of colonizing weeds and is simple but effective with accurate global search ability. Simulation results based on some well-known constrained engineering design problems demonstrate the effectiveness, efficiency and robustness of the proposed algorithm. Also the comparisons with previously reported results suggest that the robust characteristics make IWO ideally suitable as an encouraging solver of successive approximate sub-problems, when a successive approximation procedure is to be adopted in optimizing large and complicated systems. Extensive simulations are conducted along with two statistical criteria to yield useful conclusions regarding the effects of the parameter settings on the algorithm's performance. There is much more work to be done in practice. The aim of further research is to incorporate suitable local spatial dispersal models and reproduction mechanisms of seeds into IWO to further enhance and balance the exploration and exploitation abilities so as to achieve better performance.

References

[1] Papalambros P Y. The optimization paradigm in engineering design: promises and challenges [J]. Computer-Aided Design, 2002, 34 (12): 939-951.

[2] Deb K. An efficient constraint handling method for genetic algorithms[J]. Computer Methods in Applied Mechanics Engineering, 2000, 186 (2-4): 311-338.

[3] Herskovits J, Mappa P, Goulart E, et al. Mathematical programming models and algorithms for engineering design optimization[J]. Computer Methods in Applied Mechanics Engineering, 2005, 194 (30-33): 3 244-3 288.

[4] Lee K S, Geem Z W. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice[J]. Computer Methods in Applied Mechanics Engineering, 2005, 194 (36-38): 3 902-3 933.

[5] Hu X H, Eberhart R C, Shi Y H. Engineering optimization with particle swarm[C]// Proceedings of the 2003 IEEE on Swarm Intelligence Symposium. Indianapolis: IEEE Neural Networks Society, 2003: 53-57.

[6] Coelho L D S, Mariani V C. Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization [J]. Expert Systems with Applications, 2008, 34(3): 1 905-1 913.

[7] Mehrabian A R, Lucas C. A novel numerical optimization algorithm inspired from weed colonization [J]. Ecological Informatics, 2006, 1(4): 355-366.

[8] Jarchi S, Rashed-Mehassel J, Neshati M H, et al. A dual resonance three segment rectangular dielectric resonator antenna[C]// PIER Proceedings. Prague, Czech Republic: IEEE Press, 2007: 516-520.

[9] Zhang X C, Wang Y L, Cui G Z, et al. SIWO: a hybrid algorithm combined with the conventional SCE and novel IWO[J]. Journal of Computational and Theoretical Nanoscience, 2007, 4(7/8):1 316-1 323.

[10] Kuri-Morales A F, Guti rrez-Gar J. Penalty function methods for constrained optimization with genetic algorithms: a statistical analysis[C]// Proceedings 6th European Congress on Intelligent Techniques & Soft Computing. 1998: 518-522.

[11] Parsopoulos K E, Vrahatis M N. Unified particle swarm optimization for solving constrained engineering optimization problems[J]. Lecture Notes in Computer Science, 2005, 3612: 582-591.

[12] Coello Coello C A, Montes E M. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection[J]. Advanced Engineering Informatics, 2002, 16(3): 193-203.