Some Penalty-based Constraint Handling Techniques with Ant Lion Optimizer for Solving Constrained Optimization Problems

Islam S. Fathi
Department of Mathematics
Faculty of Science, Zagazig University
P. O. Box 44519, Zagazig, Egypt

Rasha M. Abo-Bakr
Department of Mathematics
Faculty of Science, Zagazig University
P. O. Box 44519, Zagazig, Egypt

R. M. Farouk
Department of Mathematics
Faculty of Science, Zagazig University
P. O. Box 44519, Zagazig, Egypt

ABSTRACT
In order to solve constraint optimization problems, constraints should be handled. The most common technique is penalty functions. Ant lion optimizer (ALO) is one of meta-heuristic algorithms which used to solve optimization problems. In this paper, the performance of ALO using different penalty-based methods (static penalty, dynamic penalty, and adaptive penalty) is compared and we make sensitivity analysis of tuning important parameters of penalty methods to show their effects on the performance of the penalty methods; six real engineering problems are used as a benchmark in this paper.

Keywords
Constrained optimization problems, ant lion optimizer, penalty functions**,** constraint handling**.**

1. INTRODUCTION
Optimization problems can be written mathematically as:

$$\begin{aligned} &min_x\, f(x)\ \ Subject\ to \\ &g_j(x) \le 0 \qquad ,j = 0,1,\dots,m \\ &h_k(x) = 0 \qquad ,k = 0,1,\dots,p \end{aligned} \tag{1}$$

Equation (1) is called unconstrained optimization problem if $j = k = 0$, but if $j \neq 0$ or $k \neq 0$ then it is called constrained optimization problem.

Where $x = (x_1, x_2, \dots\dots\dots, x_n)$ is the vector of solution such that $x \in S \subseteq R^n$, S is the search space defined as n-dimensional bounded space, m is the number of inequality constraint, p is the number of equality constraint, and F $\subseteq S$ is feasible region which the inequality and equality constraint are satisfied.

Equality constraints $h_k(x)$ are converted into inequality constraint using

$$|h_k(x)| - \varepsilon \le 0 \tag{2}$$

$where\ \varepsilon$ is a very small value.

With respect to constraint optimization problems, there are several approaches to handle constrained problem, one of the most popular approach often used is the penalty function [1].
The idea of penalty function is transform constraint optimization problem into an unconstrained problem by adding or subtracting value to the objective function this value called penalty term. There are many methods in penalty function approach to handle constraint problem such as: static penalty, dynamic penalty, adaptive penalty [2]. Each of this method follows general idea of penalty function approach, but the difference between them is the form of the penalty term.

The Ant lion optimizer (ALO) considered one of the latest nature-inspired algorithms which mimic the intelligent behavior of antlion in hunting ants in nature [3]. ALO is one of stochastic optimization (metaheuristic) algorithm that provides very competitive results in unconstrained or constrained optimization problem, using ALO in constrained engineering problem need to appropriate constraint handling method.

This paper compares the performance of different penalty methods for solving constrained optimization problem with ALO.

2. METHODS OFPENALTY FUNCTION
Penalty function one of constrained handle approaches which proposed by courant in 1940[4]. The main idea of this approach is to convert the constrained problem to unconstrained one through adding or subtracting value to objective function [5].

In general the idea of penalty function can be formulated as

For additive way,

$$eval(\bar{x}) = \begin{cases} f(\bar{x}), & if\ \bar{x} \in F \\ f(\bar{x}) + p(\bar{x}), & otherwise \end{cases} \tag{3}$$

Where $eval(\bar{x})$ is penalized function, $f(\bar{x})$ is objective function and $p(\bar{x})$ is penalty term. Value of $p(\bar{x})$ is zero if no violated constraint and positive otherwise.

For multiplicative way,

$$eval(\bar{x}) = \begin{cases} f(\bar{x}), & if\ \bar{x} \in F \\ f(\bar{x})p(\bar{x}), & otherwise \end{cases} \tag{4}$$

Value of $p(\bar{x})$ is one if no violated constraint and bigger than one otherwise.

In mathematical programming, there are two types of penalty function are used: exterior and interior penalty function. In exterior type, we begins with an infeasible solution and from there we move to the feasible region but in interior type, we begins with an initial point inside the feasible region and the constraint boundaries make the subsequent point generated always lie within the feasible region. One of important drawback of implementation interior penalties is that it needs to begin with an initial feasible solution and it is difficult, so exterior penalties is the most common type used in evolutionary algorithms (ALO) [6].

The general formulation of exterior penalties is [5].

$$\emptyset(\bar{x}) = f(\bar{x}) \pm \left[\sum_{i=1}^{n} r_i\, G_i + \sum_{j=1}^{m} c_j L_j\right] \quad (5)$$

Where $\emptyset(\bar{x})$ shows the new objective function to be optimized, r_i and c_j penalty parameter, $G_i$and L_j the constraint function of $g_i(\bar{x})$ and $h_j(\bar{x})$ respectively.

The general form of $G_i$and L_j is

$$G_i = max[0, g_i(\bar{x})]^{\beta}$$

$$L_j = \left|h_j(\bar{x})\right|^{\gamma}$$

Where β and γ are 1 or 2.

In equation (5) the second term in right side called "penalty term". If the constraint are hold such that $g_i(\bar{x}) \leq 0$ and $h_j(\bar{x}) = 0$ then G_i will be zero and $\emptyset(\bar{x})$ not effected by penalty term, but if there are violation for constraint such that $h_j(\bar{x}) \neq 0$ or $g_i(\bar{x}) > 0$, the value of penalty term add to $\emptyset(\bar{x})$.

The value of penalty term has an important role in getting optimization region. In case of the penalty is too high, algorithm will be pushed inside the feasible region very quickly and not able to explore in the boundary of the feasible region [7, 8]. In case of penalty is too low, algorithm will be spent a lot of search time in exploration of the infeasible region because the penalty will be negligible with respect to the objective function [9] and these is an important issues especially if the optimum of the problem laying on the boundary of the feasible region [10, 11] so the penalty should be kept as low as possible, just above the limit below which infeasible solution are optimal (the minimum penalty rule) [12].

2.1 Static Penalty

In static penalty method the penalty parameter doesn't depend on the current generation of the algorithm that employs the function and remain constant during evolutionary process. In this method optimization process starts with a random population using both feasible and infeasible individuals [13].

In this method we evaluate the individual using

$$fitness(\bar{x}) = f(\bar{x}) + d\sum_{i=1}^{m}\{\max[0, g_i(\bar{x})]\} \quad (6)$$

Where d is penalty parameter, $f(\bar{x})$ is the unpenalized objective function, m is the number of constraints.

2.2 Dynamic Penalty

In this method, penalty parameter depends on current generation number. The dynamic function evaluates individuals at each generation t which proposed by Joines and Houck [14] is defined as:

$$fitness(\bar{x}) = f(\bar{x}) + (c \times t)^{\propto} \times SVC(\beta, \bar{x}) \quad (7)$$

Where c,$\propto$ and β are constant determined by the user (c=0.5, $\propto$= 1 or 2, β= 1 or 2) and $SVC(\beta, \bar{x})$ is defined as:

$$SVC(\beta, \bar{x}) = \sum_{i=1}^{m} D_i{}^{\beta}(\bar{x}) + \sum_{j=1}^{q} D_j(\bar{x}) \quad (8)$$

Where

$$D_i(\bar{x}) = \begin{cases} 0, & g_i(\bar{x}) \leq 0,\ 1 \leq i \leq m \\ |g_i(\bar{x})|, & otherwise \end{cases} \quad (9)$$

$$D_j(\bar{x}) = \begin{cases} 0, & -\varepsilon \leq h_j(\bar{x}) \leq \varepsilon, \quad 1 \leq i \leq q \\ |h_j(\bar{x})|, & otherwise \end{cases} \quad (10)$$

The value of penalty function increase as generation grows. In Joines [10] and Houck approaches, there found that no explanation regarding the sensitivity of the method to different values of c, but the quality of the solution is very sensitive to changes in value of $\propto$ andβ , and the values indicated above for this parameter is good selection, Michalewicz [15] state that this values cause premature convergence in some example. He also state that the method converges either to infeasible solution or solution that is far away from an optimal solution.

2.3 Adaptive Penalty

The method of adaptive penalty depends on updating the penalty parameter at every generation (iteration) according to individual in last generation. In Hadj-Alouane and Bean [16] approach the individual are evaluated by following formula:

$$fitness(\bar{x}) = f(\bar{x}) + \mathrm{k(t)}\left[\sum_{i=1}^{n} g_i{}^2(\bar{x}) + \sum_{j=1}^{p}\left|h_j(\bar{x})\right|\right]$$
(11) (11)

Where the penalty parameter k(t) is updated at every generation by:

$$\mathrm{k(t+1)} = \begin{cases} \beta_1.\mathrm{k(t)} & \text{if case\#1,} \\ \beta_2.\mathrm{k(t)} & \text{if case\#2 ,} \\ \mathrm{k(t)} & otherwise, \end{cases}$$

$(\beta_1 \neq \beta_2\ , \beta_2 > \beta_1\ )$

Where case#1 indicates that all the best individual in last generation are feasible (i.e. the penalty parameter is decreased), case#2 indicates that all the best individual in last generation are infeasible (i.e. the penalty parameter is increased) and if some of the best individual in last generation are feasible and some infeasible the penalty parameter does not change.

3. THE ANT LION OPTIMIZER ALGORITHM

3.1 Main Inspiration

Ant lion algorithm inspired from intelligent behavior of antlion's larvae, where it digs pit in the form of cone using its strong jaw [17] as illustrated in fig.1. Antlion hides in cone waiting for ant or insects to slip on the sand and fall in [18]. The edge of pit is sharp leading to the fall of the prey easily. When prey fall in pit ant lion slide it into the bottom of the pit as illustrated in fig.2. Finally antlion amend the trap to next hunt.

It has been observed that whenever the antlion hungrier whenever the trap bigger and it has the higher chance of catching ant [19].

3.2 Mathematical Formula For The ALO

Ant and Antlion are main elements over search space, ant move to search food and antlion await to hunt it.

Firstly, modeled the position of ant in search space in the following matrix:

$$\boldsymbol{H_{ANT}} = \begin{bmatrix} \boldsymbol{H_{1,1}} & \boldsymbol{H_{1,2}} & \ldots\ldots & \boldsymbol{H_{1,m}} \\ \vdots & \vdots & \vdots\ \vdots & \vdots \\ \boldsymbol{H_{n,1}} & \boldsymbol{H_{n,2}} & \ldots\ldots & \boldsymbol{H_{n,m}} \end{bmatrix}$$

Where H_{ANT} is matrix of position of each ant over search space, $H_{i,j}$ is position of i-th ant in j-th dimension(variable), n shows the number of ants and m shows number of dimensions(variable) in space.

The fitness value of all ants can be expressed as:

$$f\ (H_{ANT}) = \begin{bmatrix} f(H_{1,1} & ,H_{1,2} & ,\dots\cdots & ,H_{1,m)} \\ & & \vdots & \\ f(H_{n,1} & ,H_{n,2} & ,\dots\dots & ,H_{n,m)} \end{bmatrix}$$

Where f(H_{ANT}) is the matrix for the fitness value for all ants and f is the objective function.

Besides ant, antlion take position over search space.

$$H_{ANTLION} = \begin{bmatrix} HL_{1,1} & HL_{1,2} & \dots\dots & HL_{1,m} \\ \vdots & \vdots & \vdots\ \vdots & \vdots \\ HL_{n,1} & HL_{n,2} & \dots\dots & HL_{n,m} \end{bmatrix}$$

Where $H_{ANTLION}$ is the matrix of the position of each antlion over search space, $HL_{i,j}$ is the position of i-th antlion in j-th dimension(variable), n shows the number of antlions and m shows the number of dimensions(variables) in space.

The fitness value of all antlions can be expressed as:

$$f(\ H_{ANTLION}) = \begin{bmatrix} f(HL_{1,1} & ,HL_{1,2} & ,\dots\cdots & ,HL_{1,m)} \\ & & \vdots & \\ f(HL_{n,1} & ,HL_{n,2} & ,\dots\dots & ,HL_{n,m)} \end{bmatrix}$$

Where f ($H_{ANTLION}$) is the matrix for the fitness value for all antlions and f is the objective function

3.2.1 *Ants random walk*

Since the ants moves randomly in nature when searching for food, This movement is modeled as follows:

R^t =[0, cumsum (2r (t_1) -1), cumsum (2r (t_2) -1),., cumsum (2r (t_{max}) -1)] (12)

Where cumsum indicates the cumulative sum , t_{max} is the maximum number of iteration, t denotes the current iteration, and r (t) is defined as:

$$r(t) = \begin{cases} 1\ if\ rand > 0.5 \\ 0\ if\ otherwise \end{cases} \quad (13)$$

Where rand is random number between 0 and 1

3.2.2 *Convergence of Ant towards antlion*

When ant trapped in the pit, it sliding towards antlion, ant getting close to antlion, so the boundary of search space decreased adaptively. This may be expressed as:

$$u^t = \frac{u^t}{I} \quad (14)$$

$$l^t = \frac{l^t}{I} \quad (15)$$

Where u^t is upper bounded at t-th iteration, l^t is lower bounded at t-th iteration, I denotes ratio defined as $I = 10^W \frac{t}{t_{max}}$, where w is a constant defined as: (w = 2 when t > $0.1t_{max}$, w = 3when t > $0.5t_{max}$, w = 4 when t > $0.75t_{max}$, w = 5 when t > $0.9t_{max}$, and w=6 when t > $0.95t_{max}$).

3.2.3 *Ants are affected by antlions' traps*

Each ant affected by traps of antlion either antlion selected by roulette wheel or elite antlion in each iteration.
This affected may be expressed in simple form as:

$$u_i^t = {H_{ANTLION}}_j^t + u^t \quad (16)$$

$$l_i^t = {H_{ANTLION}}_j^t + l^t \quad (17)$$

Where u_i^t indicates upper bounded for i-th ant in iteration t, l_i^t indicates lower bounded for i-th ant in t-th iteration, u^t is upper bounded, l^t is lower bounded, ${H_{ANTLION}}_j^t$ is position of j-th antlion in t-th iteration.

3.2.4 *Random walks around elite and selected antlion*

As explained above the better anlion the higher chance of catching ant, roulette wheel used to select the fittest antlion. On the other hand, in each iteration produce the best antlion .so the movement of ant affected by elite antlion and a selected antlion by roulette wheel. Ant walks randomly around $H(elite)_{ANTLION}$ and$H(selected)_{ANTLION}$ as:

$${H_{ANT}}_i^t = \frac{R_{selected}^t + R_{elite}^t}{2} \quad (19)$$

Where ${H_{ANT}}_i^t$ shows position of i-th ant at t-th iteration, $R_{selected}^t$ shows the random walk around selected antlion using roulette wheel at t-th iteration, R_{elite}^t shows the random walk around the elite at t-th iteration.

3.2.5 *Catching ants*

After antlion catch ant and pulls it inside the sand, antlion update their position with corresponding ant. This occurs when fitness of ant is more than fitness of antlion, this expressed as:

$${H_{ANTLION}}_j^t = {H_{ANT}}_i^t\ if\ f({H_{ANT}}_i^t) > f({H_{ANTLION}}_j^t) \quad (20)$$

Where ${H_{ANTLION}}_j^t$ indicates the position of j-th antlion at t-th iteration, Where ${H_{ANT}}_j^t$ indicates the position of i-th ant at t-th iteration.

4. NUMERICAL EXPERIMENTS

To evaluate the performance of the ALO algorithm with the different penalty methods mentioned above in solving constraint optimization problems, we conducted experimental study in which six classical engineering problems.

Each of the test problems is solved by ALO, with each of penalty-based methods (static penalty, dynamic penalty, and adaptive penalty) 30 runs over 1000 iterations and the result are reported in tables 1-12. Best indicates to the best solution, Ave indicates to the average solution and S.D indicates to standard deviation. The specific parameters of each penalty method are used as following:

- In static penalty (d=10^{10}).
- In dynamic penalty (c=0.5, $\propto$= 2, β= 1).
- In adaptive penalty ($\lambda(0) = 10^4$, $\beta_1 = 0.95$, $\beta_2 = 1.1$)..

Test problem 1: Pressure Vessel Design Problem

The objective is to minimize the total cost, including the cost of material, forming, and welding. A cylindrical vessel is capped at both ends by hemispherical heads as shown in Figure 1. They are four design variables in this problem: Ts (thickness of the shell), T_h (thickness of the head), R (inner radius), and L (length of the cylindrical section of the vessel). Among the four design variables, T_s and T_h are expected to be integer multiplies of 0.0625 inch, and R and L are continuous variables [20].

The problem formulation is as follows:

Minimize f (x) = $0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$

Subject to: $g_1(x) = -x_1 + 0.0193x_3 \le 0$,
$g_2(x) = -x_2 + 0.00954x_3 \le 0$,
$g_3(x) = -\pi x_3^2x_4 - (4/3)\,\pi x_3^3 + 1,296,000 \le 0$,
$g_4(x) = x_4 - 240 \le 0$.

Variable rang: $0 \le xi \le 100$, i = 1, 2,
$10 \le xi \le 200$, $i = 3, 4$.

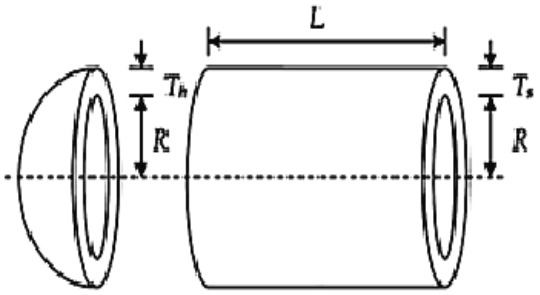


Fig 1: Pressure Vessel Design Problem

This test problem has been solved by ALO algorithm with three penalty-based methods (static penalty, dynamic penalty, and adaptive penalty) and the results represented in tables 1 and 2. The comparison results for this problem show that the adaptive penalty method outperforms on static and dynamic methods in terms of best solution, but dynamic penalty method is outperform in term of Ave and S.D.

Table 1. Compression the best solution for pressure vessel design problem by three penalty-based methods

penalty methods	x_1	x_2	x_3	x_4	f_{min}
Static penalty	7.7920E-01	3.8516E-01	4.0373E+01	1.9926E+02	5.8871E+03
Dynamic penalty	7.8064E-01	3.8587E-01	4.0448E+01	1.9823E+02	5.8897E+03
Adaptive penalty	7.7825E-01	3.8469E-01	4.0324E+01	1.9997E+02	5.8859E+03

Table 2. Statistical performance of three penalty-based methods for pressure vessel design problem

Results	Static penalty	Dynamic penalty	Adaptive penalty
Best	5.8871E+03	5.8897E+03	5.8859E+03
Ave	6.7932E+03	6.2888E+03	6.5141E+03
S.D.	2.1554E+03	3.4791E+02	7.4712E+02

Test problem 2: Cantilever Beam Design Problem

The cantilever beam design is made of five elements, each having a hollow cross-section with constant thickness as shown in figure 2. There is a total of 5 structure parameters and also a vertical load applied to the free end of the beam (node 5) and the right side of the beam (node1) is rigidly supported. The aim is to minimize the Weight of the beam. In the final optimal design there is one vertical displacement constraint that should not be violated [21].The mathematical formulation of this problem is as follows:

Minimize f (x) =$0.6224(x_1 + x_2 + x_3 + x_4 + x_5)$,

Subject to: $g(x) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} \le 1$,

Variable rang: $0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100$,

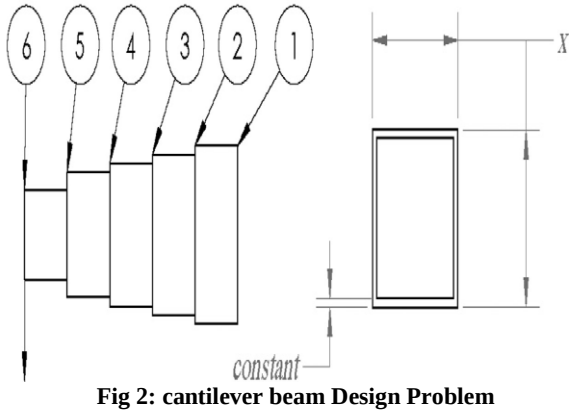


Fig 2: cantilever beam Design Problem

The optimal value of the objective function, average and standard deviation for this problem which solved by ALO algorithm with their penalty methods reported in tables 3 and 4. The results show that the static penalty method is better than dynamic and adaptive in terms of best, but in terms of Ave and S.D the adaptive penalty method outperforms on static and dynamic.

Table 3. Compression the best solution for cantilever beam design problem by three penalty-based methods

penalty methods	x_1	x_2	x_3	x_4	x_5	f_{min}
Static penalty	6.016998	5.309663	4.492589	3.501620	2.152792	13.365207
Dynamic penalty	6.020794	5.301333	4.499755	3.499036	2.152790	13.365235
Adaptive penalty	6.008809	5.315777	4.493109	3.502588	2.153413	13.365228

Table 4. Statistical performance of three penalty-based methods for cantilever beam design problem

Results	Static penalty	Dynamic penalty	Adaptive penalty
Best	13.365207	13.365235	13.365399
Ave	13.365437	13.365421	13.365399
S.D.	0.000205	0.000173	0.000140

Test problem 3: Gear Train Design Problem

In gear train problem the objective is to minimize the cost of the gear ratio of the gear train as shown in figure 3. The constraints are only limits on design variables (side constraints). Design variables to be optimized are in discrete form since each gear has to have an integral number of teeth. Constrained problems with discrete variables may increase the complexity of the problem [3]. The decision variables of the problem are n_A, n_B, n_C, and n_D which are denoted as x_1, x_2, x_3, and x_4, respectively. This problem is given by:

Minimize $f(x) = (\frac{1}{6.931} - \frac{x_3 x_2}{x_1 x_4})^2$,

Subject to: $12 \leq x_1, x_2, x_3, x_4 \leq 60$,

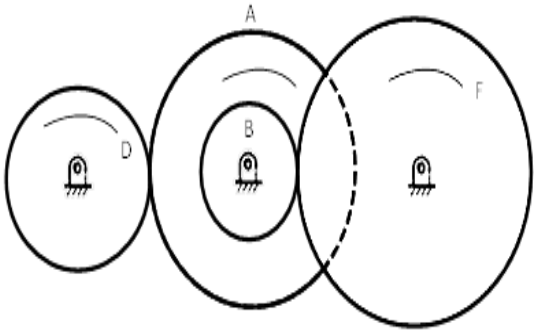


Fig 3: Gear train Design Problem

Tables 5 and 6 represent the result of solving this problem by ALO using three penalty-based methods. The adaptive penalty method in obtained the best results from the static and dynamic penalties in best, Ave, S.D.

Table 5. Compression the best solution for Gear train design problem by three penalty-based methods

penalty methods	x_1	x_2	x_3	x_4	f_{min}
Static penalty	43.4493	14.9087	12.0095	28.5615	4.0653E-23
Dynamic penalty	43.4493	14.9087	12.0095	28.5615	4.0653E-23
Adaptive penalty	55.1986	38.6950	38.6950	58.3125	3.4239E-25

Table 6. Statistical performance of three penalty-based methods for Gear train design problem

Results	Static penalty	Dynamic penalty	Adaptive penalty
Best	4.0653E-23	4.0653E-23	3.4239E-25
Ave	6.8853E-20	6.8853E-20	5.2360E-20
S.D.	1.1554E-19	1.1554E-19	8.6603E-20

Test problem 4: Welded Beam Design Problem

A welded beam is designed for minimum cost subject to constraint on share stress (τ), buckling load on the bare (p_c), bending stress in the beam (σ), end deflection of the beam (δ), and side constraints [22], as shown in figure 4 there are four design variable (x_1), $(x_2), (x_3)$ *and* (x_4), the mathematical formulation of this problem is as follows:

Minimize $f(x) = 1.10471x_1^2 x_2 + 0.04811x_3x_4(14.0+x_2)$

Subject to: $g_1(x) = \tau(x) - \tau_{max} \leq 0$,

$g_2(x) = \sigma(x) - \sigma_{max} \leq 0$,

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

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

$g_5(x) = 0.125 - x_1 \leq 0$,

$g_6(x) = \delta(x) - \delta_{max} \leq 0$,

$g_7(x) = p - p_c(x) \leq 0$,

Where

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

$\tau' = \frac{p}{\sqrt{2}x_1x_2}$, $\tau'' = \frac{MR}{J}$,$M = p(L + \frac{x_2}{2})$,

$R = \sqrt{\frac{x_2^2}{4} + (\frac{x_1+x_3}{2})^2}$,

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

$\sigma(x) = \frac{6pL}{x_4x_3^2}$, $\delta(x) = \frac{4pL}{Ex_3^3x_4}$,

$p_c(x) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}})$

Variable rang: $0.1 \leq xi \leq 2$, $i = 1, 4$,
$0.1 \leq xi \leq 10$, $i = 2, 3$.

P=6000 lb, L=4 in, $\delta_{max} = 0.25$, E= 30×10^6 psi, G= 12×10^6 psi, $\tau_{max} = 13600$,
$\sigma_{max} = 30000$ psi

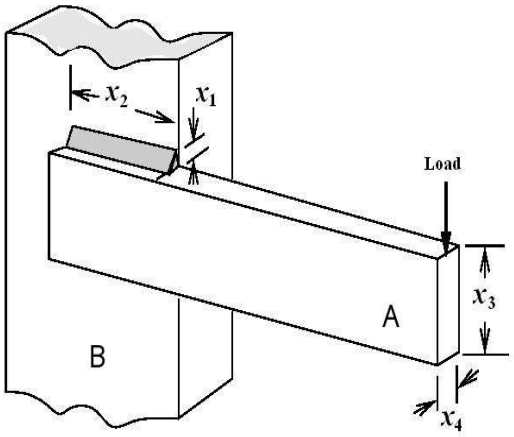


Fig 4: Welded Beam Design Problem

In Tables 7 and 8 the results of this problem are represented, it shows that the adaptive penalty methods have better solution in term of best but not better in the terms of Ave, S.D.

Table 7. Compression the best solution for welded beam design problem by three penalty-based methods

penalty methods	x_1	x_2	x_3	x_4	f_{min}
Static penalty	0.2054	7.1076	9.0367	0.2057	2.2192
Dynamic penalty	0.2054	7.0702	9.0749	0.2055	2.2203
Adaptive penalty	0.2056	7.1006	9.0367	0.2057	2.2187

Table 8. Statistical performance of three penalty-based methods for welded beam design problem

Results	Static penalty	Dynamic penalty	Adaptive penalty
Best	2.2192	2.2203	2.2187
Ave	2.3010	2.3231	2.3251
S.D.	0.0748	0.0925	0.0910

Test problem 5: Three-Bar Truss Design Problem
In a three-bar truss problem the objective is to minimize its weight [3]. The structural design problem has a large number of constraints and the objective function is very simple. The constraints here are stress, deflection, and buckling constraints as shown in figure 5.

The mathematical formulation of this problem is as follows:

Minimize $f(x) = (2\sqrt{2}\,x_1 + x_2) * l$,

Subject to: $g_1(x) = \frac{\sqrt{2}\,x_1 + x_2}{\sqrt{2}\,x_1^2 + 2x_1x_2}\,p - \sigma \leq 0$,

$g_2(x) = \frac{x_2}{\sqrt{2}\,x_1^2 + 2x_1x_2}\,p - \sigma \leq 0$,

$g_3(x) = \frac{1}{\sqrt{2}x_2 + x_1}\,p - \sigma \leq 0$,

Variable rang: $0 \leq x_1, x_2 \leq 1$,

Where $l = 100cm$, $p = 2$ kN/cm^2 , $\sigma = 2$ kN/cm^2

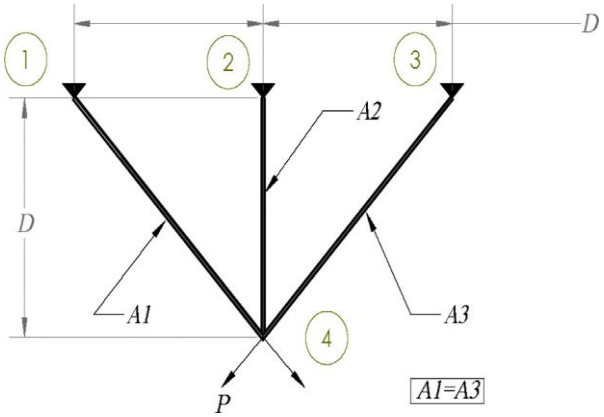


Fig 5: Three-Bar Truss Design Problem

The results of the best solution and Statistical performance for this problem are reported in tables 9 and10; the static and dynamic methods are better than adaptive penalty in terms of best but the adaptive penalty method outperforms on Ave, S.D.

Table 9. Compression the best solution for three-bar truss design problem by three penalty-based methods

penalty methods	x_1	x_2	f_{min}
Static penalty	0.788646	0.408330	263.895844
Dynamic penalty	0.788652	0.408314	263.895844
Adaptive penalty	0.788737	0.408072	263.895846

Table 10. Statistical performance of three penalty-based methods for three-bar truss design problem

Results	Static penalty	Dynamic penalty	Adaptive penalty
Best	263.89584	263.89584	263.89585
Ave	263.89629	263.89629	263.89617
S.D.	0.0006397	0.0007456	0.0003781

Test problem 6: Tension/compression spring Design Problem

This problem minimize The weight of a tension/compression spring as shown in Figure 6, subject to constraints on minimum deflection, shear stress, surge frequency, limits on outside diameter, and design variables. The design variables are wire diameter (x1), mean coil diameter (x2), and number of active coils (x3) [23].

The formal statement is:

$$\text{Minimize} \quad f(x) = (x_3 + 2)x_2x_1^2$$

$$\text{Subject to:} \quad g_1(x) = 1 - \frac{x_2^3x_3}{71{,}785x_1^4} \leq 0$$

$$g_2(x) = \frac{4x_2^2 - x_1x_2}{12{,}566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0$$

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

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

$$\text{Variable rang:} \quad 0.05 \leq x_1 \leq 2.00, 0.25 \leq x_2 \leq 1.30, 2.00 \leq x_3 \leq 15.00$$

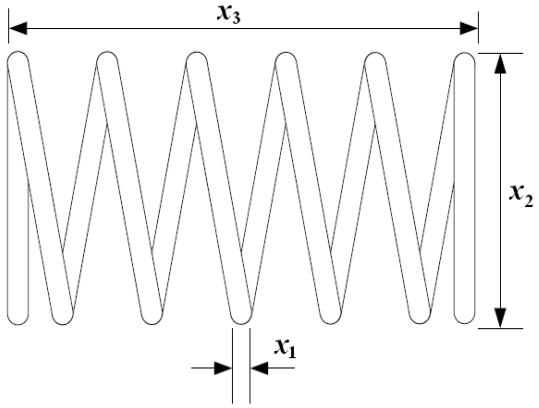


Fig 6: Tension/compression spring Design Problem

Tension/compression spring Design problem is solved by ALO with each of penalty-based methods; the results of the best solution and Statistical performance which represented in tables 11and 12 indicates that the adaptive penalty method is obtained the best result from static and dynamic in best term and dynamic penalty have better solutions in Ave and S.D.

Table 11. Compression the best solution for Tension/compression spring design problem by three penalty-based methods

penalty methods	x_1	x_2	x_3	f_{min}
Static penalty	0.0514103	0.3500494	11.690909	0.012666
Dynamic penalty	0.0520775	0.3661357	10.757475	0.012668
Adaptive penalty	0.0518434	0.3604415	11.073951	0.012665

Table 12. Statistical performance of three penalty-based methods for Tension/compression spring design problem

Results	Static penalty	Dynamic penalty	Adaptive penalty
Best	0.012667	0.012668	0.012666
Ave	0.013669	0.013450	0.013600
S.D.	0.001613	0.001341	0.001726

The convergence curves of ALO with static, dynamic and adaptive penalties on engineering problems are illustrated in Fig 7.

5. ANALYSIS OF PARAMETER VALUES FOR PENALTY METHODS

In this section, we have provided sensitively analysis for the parameter of penalty methods and its effects the performance of results of test problems. The static and adaptive penalties are implemented but in dynamic penalty parameter t dependent on current generation number (iteration).

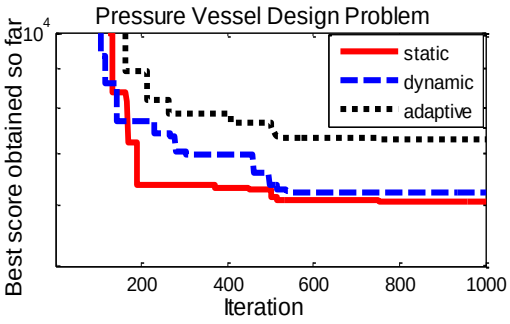
5.1 Static Penalty

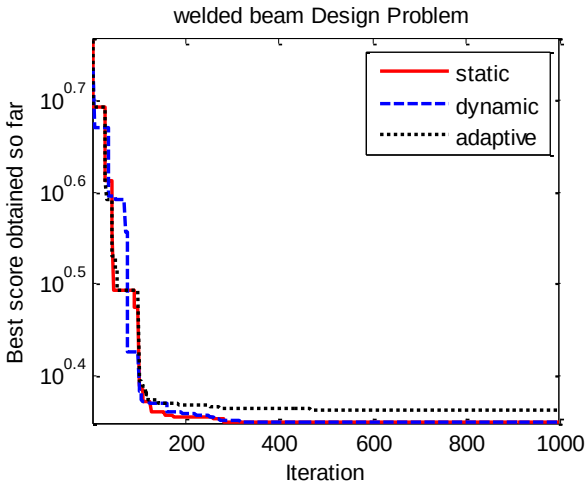
In Static penalty we study the effects of parameter d on the average value of test problems, the comparative results of the average value of 10 runs are illustrated in table 13 and figure 8.

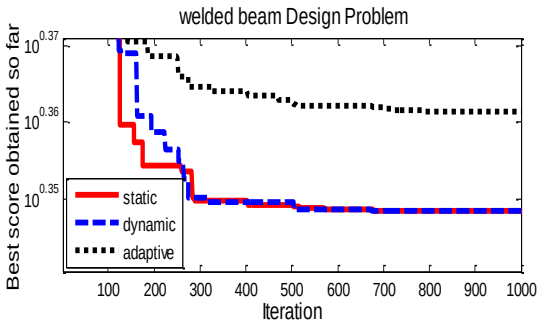
Table13 and Figure 8 shows that the average values sometimes improves and sometimes worsen with the increase of parameter d.

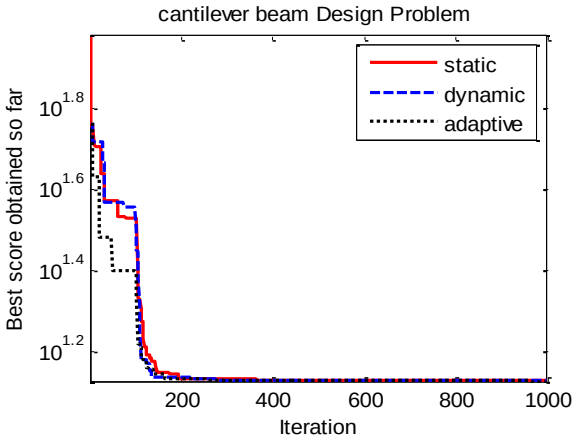
5.2 Adaptive Penalty

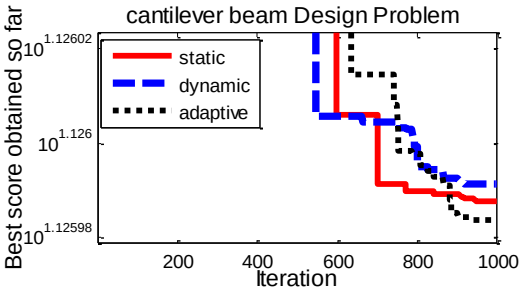
In the adaptive penalty, we study k (1) the initial value of parameter k(t) and its effects on the average value, result of the average value of 10 runs are presented in table 14 and figure 9 indicates that also the average value sometimes improve and sometimes worsen with the increase of parameter k (1), so in each penalty method, some appropriate values of penalty parameters should be selected.

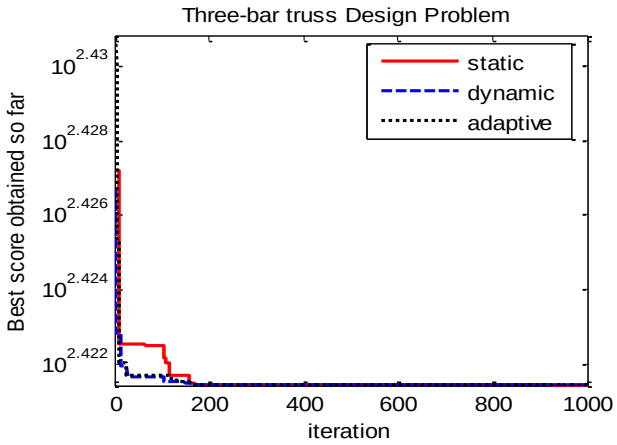












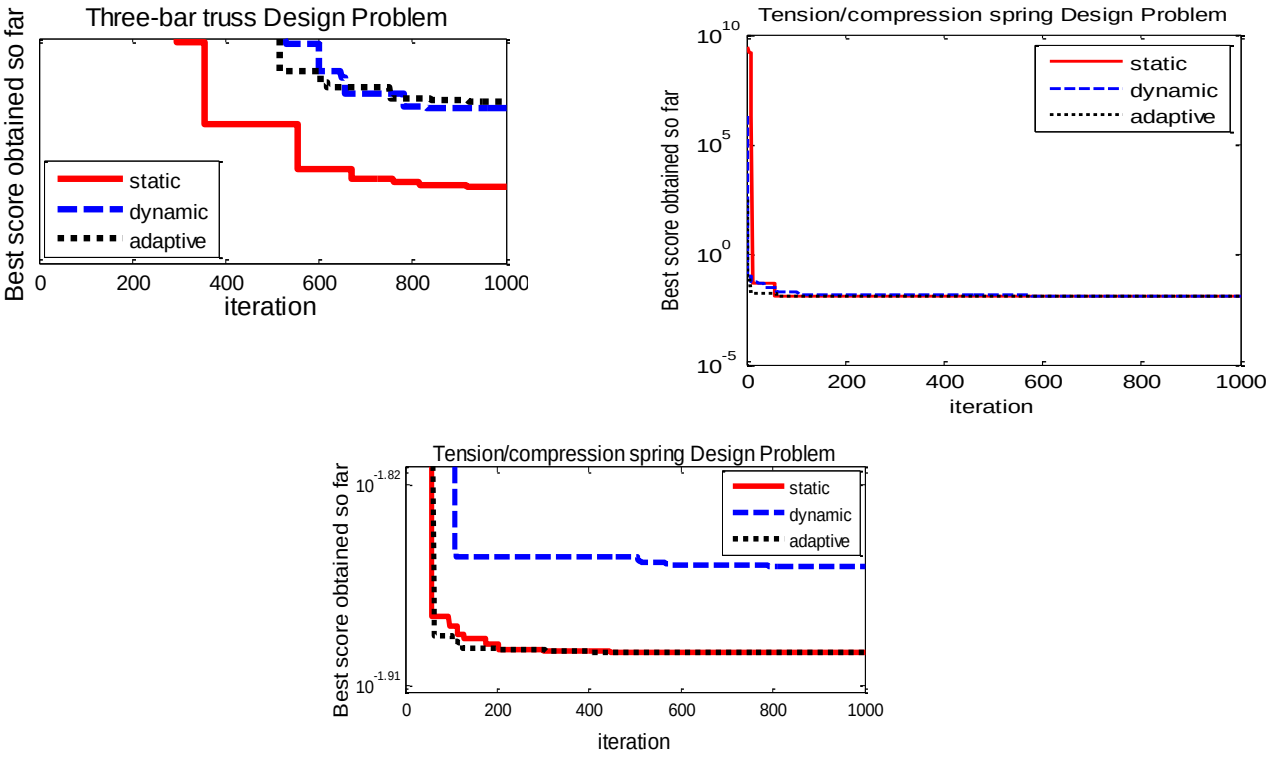
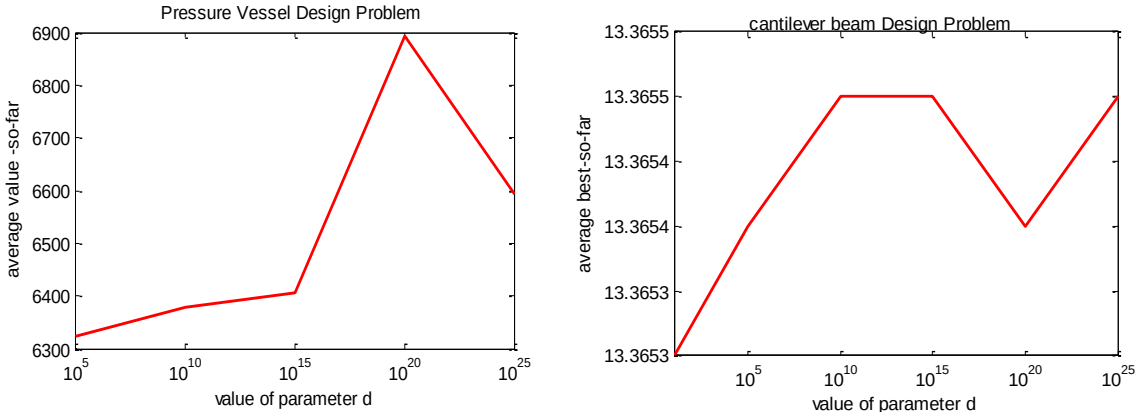


Fig.7: Convergence curves of ALO with static, dynamic and adaptive penalties on the test problems



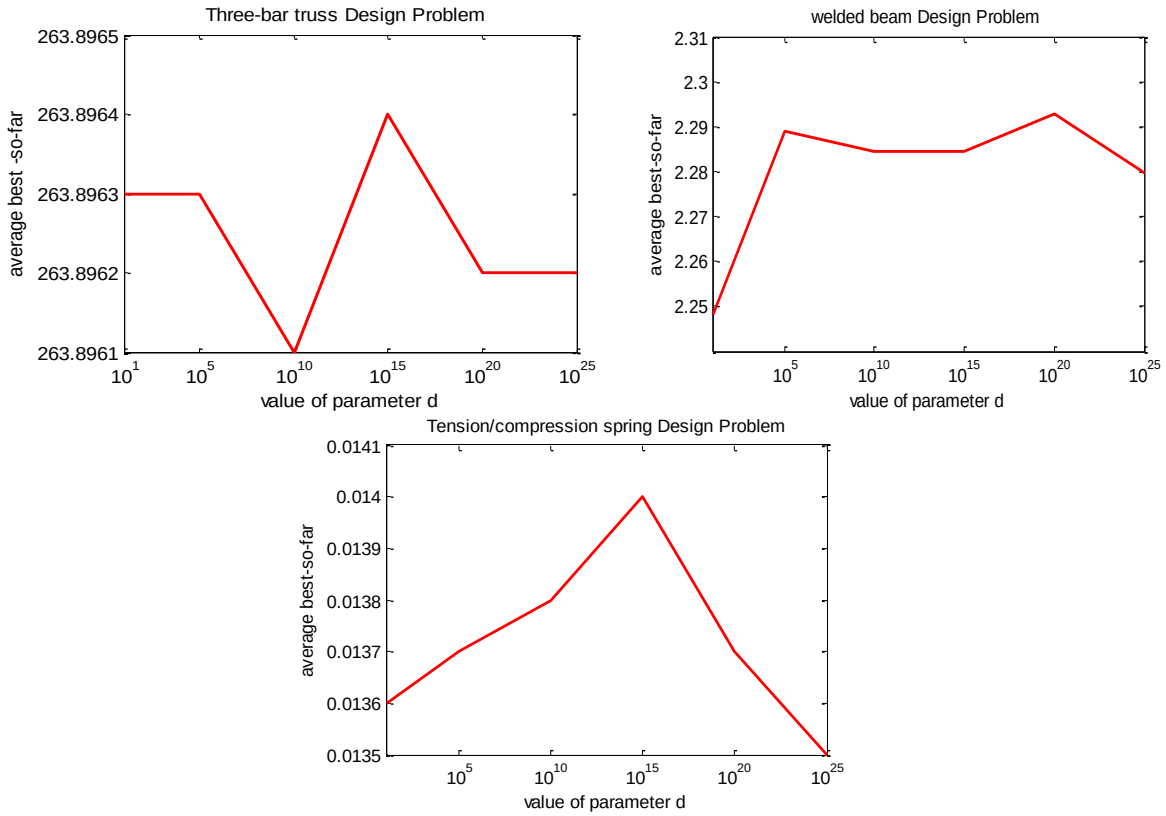


Fig 8: effects of change value of parameter d on the average best value for test problems

Table 13. Results of test problems with different values of parameter d in static penalty

d	Pressure Vessel Design Problem	cantilever beam Design Problem	welded beam Design Problem	Three-bar truss Design Problem	Tension/compression spring Design Problem
10	infeasible	13.3653	2.2483	263.8963	0.0136
10^5	6.3225e+003	13.3654	2.2889	263.8963	0.0138
10^{10}	6.3774e+003	13.3655	2.2844	263.8961	0.0137
10^{15}	6.4055e+003	13.3655	2.2844	263.8964	0.0140
10^{20}	6.8926e+003	13.3654	2.2927	263.8962	0.0137
10^{25}	6.5939e+003	13.3655	2.2796	263.8962	0.0135

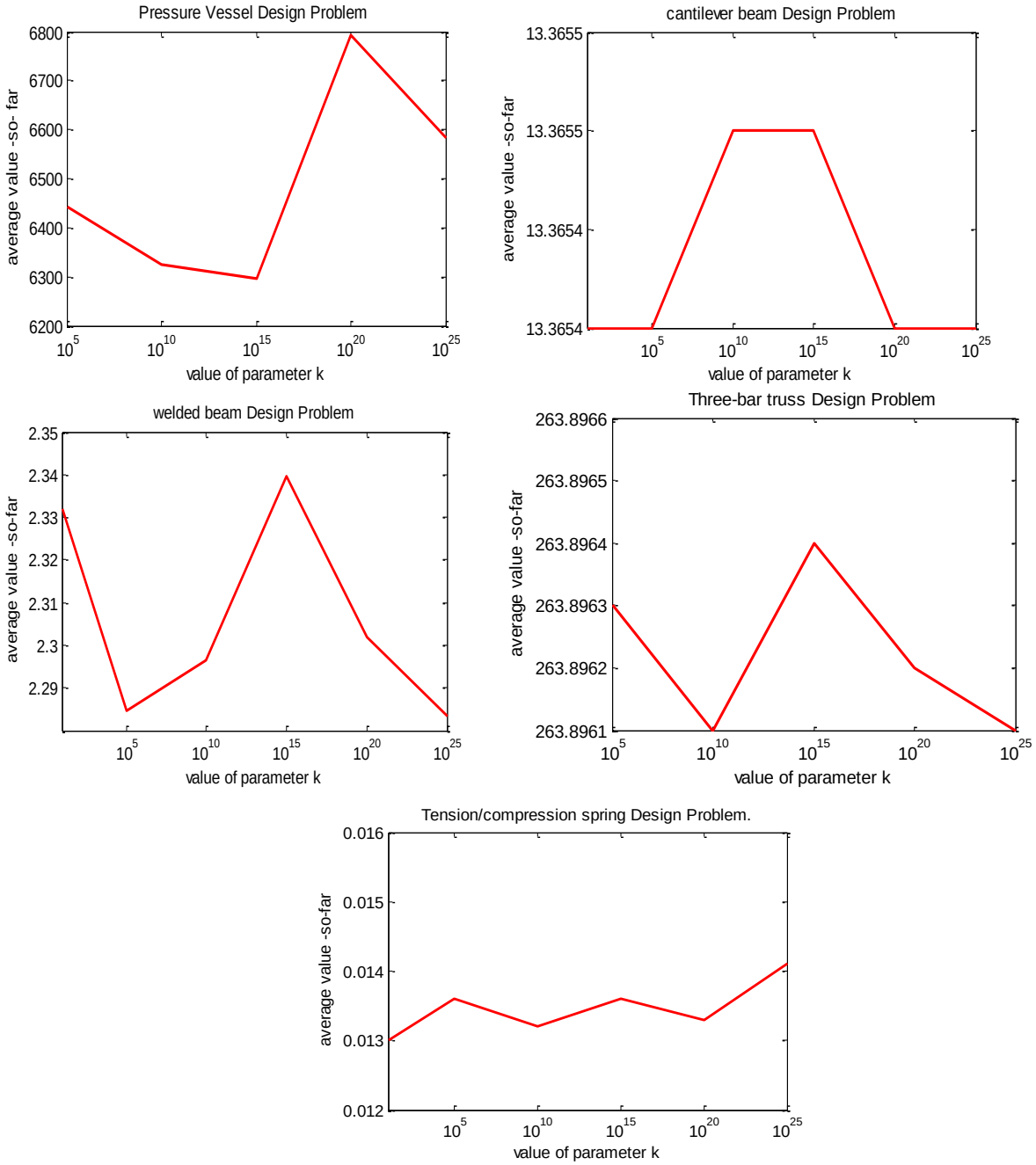


Fig 9: effects of change value of parameter k (1) on the average best value for test problems

Table 14. Results of test problems with different values of parameter k(1) in adaptive penalty

k(1)	Pressure Vessel Design Problem	cantilever beam Design Problem	welded beam Design Problem	Three-bar truss Design Problem	Tension/compression spring Design Problem.
10	infeasible	13.3654	2.3320	infeasible	0.0130
10^5	6.4414e+003	13.3654	2.2845	263.8963	0.0136
10^{10}	6.3254e+003	13.3655	2.2965	263.8961	0.0132
10^{15}	6.2965e+003	13.3655	2.3397	263.8964	0.0136
10^{20}	6.7917e+003	13.3654	2.3018	263.8962	0.0133
10^{25}	6.5837e+003	13.3654	2.2832	263.8961	0.0141

6. CONCLUSION

In order to solve the constrained optimization problems, the constraints should be handled. There are most of constraints handling techniques have been suggested. The most popular constraint handling technique among user is penalty function. In this paper, results are discussed in two ways. Firstly, we have studied the performance of the Ant Lion Optimizer (ALO) with a number of penalty-based methods (static, dynamic and adaptive penalty). A small comparative study is conducted using six real engineering problems as benchmark. Experimental results show that the methods of adaptive penalty is outperform on the static and dynamic methods in most of engineering problems, this due to that the penalty parameter doesn't remain constant but updates itself at every generation (iteration) during evolutionary process, but in general it is impossible to say one of the methods is the best for every problem.

Secondly, sensitivity analysis of choosing parameters of static and adaptive penalty methods is performed to check the performance of ALO. The obtained results show that the best value obtained with 30 runs sometimes improve and sometimes worsen with fine-tuning of the parameters and the main problem is to set appropriate values of the penalty parameters so the users have to experiment with different values of penalty parameters.

There is several research directions can be recommended for future work such as investigated the performance of ALO in other benchmark and real-life problems. Another research direction is to use other different constraints handling techniques.

7. ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to Allah - the Most Gracious, the Most Merciful for wonderful opportunity who has given me to pursue and complete my work.

I wish to thank all the people who made this work possible. I would like to thank my supervisors, *Prof. Dr. Roushdi Mohamed Farouk* and *Dr. Rasha Mohamed abobakr* for their time and cooperation in reviewing this work.

Finally, I would like to thank my father, mother, brothers and wife for their love, trust, understanding, and every kind of support not only throughout my work but also throughout my life.

8. REFERENCES

[1] Thomas Philip Runarsson, and Xin Yao. Stochastic Ranking for for Constrained Evolutionary Optimization. IEEE Transactions on Evolutionary Computation, 2000.

[2] Michalewicz, Z. and Schouenauer, M. Evolutionary algorithms for constrained parameter optimization problem, Evolutionary Computation, 4, 1-32, 1996.

[3] Mirjalili S. A., The Ant lion optimizer, Advance in Engineering software vol. 83, pp.80-90, (2015).

[4] R.Courant, variational methods for the solution of problems of equilibrium and vibration, Bull. Am. Math. Soc. 49(1943) 1-23.

[5] Carlos A. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput. Methods Appl. Mech. Engrg. 191(2002) 1245-1287.

[6] ¨Ozg¨ur Yeniay. PENALTY FUNCTION METHODS FOR CONSTRAINED OPTIMIZATION WITH GENETIC ALGORITHMS. Mathematical and Computational Applications, Vol. 10, NO. 1, PP. 45-56, 2005.

[7] R.G. Le Riche, R.T. Haftka, Optimization of laminate stacking sequence for buckling load maximization by genetic algorithm, AIAA J. 31 (5) (1993) 951-970.

[8] R.G Le Riche, C. Knopf-lenoir, R.T. Haftka, A segregated genetic algorithm for constrained structural optimization, in: L.J. Eshelman (Ed.) proceedings of the Sixth International Conference on Genetic Algorithms, University of Pittsburgh, Morgan Kaufmann, San Mateo, CA, July 1995, pp. 558-565.

[9] A.E. Smith, D.W. Coit, Constraint handling techniques – penalty function, in: T. B¨ack, D.B. Fogel, z. Michalewicz (Eds.), Handbook of Evolutionary Computation, Oxford University Press and Institute of Phsics Publishing, 1997 (Chapter C 5.2).

[10] W. Siedecki, J. Sklanski, Constrained genetic optimization via dynamic reward-penalty balancing and its use in pattern recognition, in: J.D. Schaffer (Ed.), proceedings of the Third International Conference on Genetic Algorithms, George Mason University, Morgan Kaufmann, San Mateo, CA, June 1989, pp. 141-150.

[11] A.E. Smith, D.M. Tate, Genetic optimization using a penalty function, in: S.Forrest(Ed.), proceedings of the Fifth International Conference on Genetic Algorithms, University of Illinois at Urbana-Champaign, Morgan Kaufmann, San Mateo, CA, July 1993, pp. 499-503.

[12] L. Davis, Genetic Algorithms and Simulated Annealing,

Pitman, London, 1987.

[13] Homaifar, A., Lai, S.H.Y. and Qi, X. Constrained optimization Via genetic algorithms, Simulation, 62, 242-254, 1994.

[14] Joines, J. and Houk, C. On the use of non-stationary penalty functions to solve non-linear constrained otimization problems with Gas, Proceedings of the First IEEE International Conference on Evolutionary Computation, IEEE Press, 579-584, 1994.

[15] Z. Michalewicz, G. Nazhiyath, Genocop III: A co-evolutionary algorithm for numerical optimization with nonlinear constraints, in : D.B. Fogel (Ed.) proceedings of the Second IEEE International Conference on Evolutionary Computation, IEEE press, piscataway, NJ, 1995, pp. 647-651.

[16] Hadj-Alouane, A.B. and Bean, J.C. A Genetic algorithm for the multiple-choice inter program, Operations Research, 45, 92-101, 1997.

[17] Scharf I, Subach A, Ovadia O.Foraging behviour and habitat selection in Pit-building antlion larvae in constant light or dark conditions. Anim Behav 2008;76:2049-57.

[18] Scharf I, Ovadia O. Factors influencing site abandonment and site selection in a sit-and-wait predator: a review of pit-building antlion larvae. J Insect Behav 2006;19:197-218.

[19] Grzimek B, Schlager N, Olendorf D, McDade MC. Grzimek's animal life encyclopedia. Michigan: Gale Farmington Hills; 2004.

[20] B.k. kannan, S.N. Kramer, An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design, J. Mech. Des. Trans. ASME 116 (1994) 318-320.

[21] Chickermane H, Gea H. Structural optimization using a New local approximation method. Int J Numer Meth Eng 1996;39:829-46.

[22] S.S. Rao, Engineering Optimization, third ed., Wiley, New York, 1996.

[23] Leticia C. Cagnina and Susana C. Esquivel, Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer, Informatica 32 (2008)319–326.