Evolving Systems (2020) 11:531–539
https://doi.org/10.1007/s12530-018-9250-z

ORIGINAL PAPER

Sonar inspired optimization (SIO) in engineering applications

Alexandros Tzanetos[1] · Georgios Dounias[1]

Received: 5 January 2018 / Accepted: 13 August 2018 / Published online: 28 August 2018

Abstract
Recently, a new Nature Inspired Intelligent scheme has been proposed and presented, named Sonar Inspired Optimization (SIO). This algorithm is inspired by the SONAR mechanism, which is used by Warships to detect targets and avoid mines. In this paper, improvements have been done to the SIO approach in an attempt to increase the performance of the algorithm. Also, results from experiments in known constrained Engineering applications are presented and discussed. SIO tackles with these problems, managing to overcome the performance of other Nature Inspired metaheuristics, heuristics and mathematical approaches in most of the cases.

Keywords Sonar inspired optimization · Nature-Inspired Intelligent (NII) algorithm · Engineering optimization · Tension/compression spring design · Welded beam design · Pressure vessel

1 Introduction

In the last twenty (20) years, a growth on Nature Inspired Intelligent (NII) methods (Yang 2010; Chiong 2009; Liu and Tsui 2006) is observed. Applications (Marrow 2000) and new challenges (Yang 2012) are presented, underlying the major contribution of these algorithms on the field of optimization. Except for swarm based techniques (Kennedy et al. 2001), there are many others that are inspired by physical phenomena (Shah-Hosseini 2009) and laws of science (Nasir et al. 2012). Recently the authors have extensively searched and collected all the algorithms that are based in the above-mentioned categories and extracted some useful conclusions (Tzanetos and Dounias 2017). The overwhelming majority is population based schemes. A detail that highlights the need of multiple agents to achieve high exploration, while many of these algorithms are based also on attraction between their agents through equations that model the main idea inspired from nature.

Most of the schemes used, are based on the gravitational law [Gravitational Search Algorithm (Rashedi et al. 2009)] or in attraction-based laws, e.g. charged system search (Kaveh and Talatahari 2010), Electromagnetism-like optimization (Birbil and Fang 2003). Based on these phenomena, the best solution attracts all the others towards it. On the proposed scheme, introduced in (Tzanetos and Dounias 2017a, b), each agent doesn't interact with the others and thus, performs its independent search. The only information shared between all agents is the best-so-far fitness achieved. That's a very useful feature, because all best-so-far solutions are contributing to find the best one and the algorithm cannot be trapped in local optima. So, a good balance between exploration and exploitation is achieved.

What is more, in recent works, a major point of interest is the need for parameter tuning of the metaheuristic for different kind of problems (Yang et al. 2013; Crawford et al. 2013; Fallahi et al. 2014). The initial goal was to provide a new self-tuning algorithm, which overcomes the problem of setting the exact number of agents to solve a problem. This goal is fulfilled in this work due to improvements that have been done on the initial scheme. Also, a self-tuning mechanism which is based on the fitness of the solution, controls the size of the step for the current agent.

Furthermore, a significant detail is that this algorithm needs less parameterization. One of the open issues described in the previous work (Tzanetos and Dounias 2017a, b) is solved here. The parameter of maximum

✉ Alexandros Tzanetos
atzanetos@aegean.gr

Georgios Dounias
g.dounias@aegean.gr

[1] Management and Decision Engineering Laboratory, Department of Financial and Management Engineering, University of the Aegean, 41 Kountouriotou Str., Chios 82121, Greece

rotation angle is auto-tuned based on the fitness of the solution. The concept proposed here is based on the auto-tuning of the intensity parameter that determines how big search steps in the solution space the algorithm performs. At last, the relocation mechanism has been altered to maintain the balance between exploration and exploitation.

Finally, recent reviews of nature inspired algorithms (Tzanetos and Dounias 2017; Vassiliadis and Dounias 2009; Fister et al. 2013) show that even more schemes are presented every year. The importance of a new algorithm can be shown by its effectiveness in a specific application or the usage as a hybrid component. Thus, in this paper two constrained engineering optimization problems were chosen to measure the performance of SIO. This novel method is proven to be useful as much in terms of finding optimum fitness as in consistency of providing nearly good solutions.

The rest of the paper is organized as follows; in Sect. 2 the algorithm is explained analytically, in Sect. 3 selected engineering applications are described, in Sect. 4 the experimental results are presented and explained and in Sect. 5 there are further research recommendations and conclusion.

2 Physical analogue and proposed algorithm

2.1 The actual sonar mechanism

The mechanism that provides inspiration in the proposed algorithm is the sonar that the Navy uses for war ships' exploration for submarines. The basic idea behind the sonar application was to send an ultrasound and based on the sound level that the radio receives the size of an object or an obstacle can be estimated. So, the ship can identify the position of possible targets (Fig. 1).

A characteristic feature of SONAR is the cyclic scan of the area around the ship. To model this phenomenon, the concept of intensity of sound is implemented (Lurton 2002). Initially, the acoustic power output or sound power (P) has to be calculated:

$$P = \eta \cdot Pe, \tag{1}$$

where Pe is the power input and η is the transducer efficiency, which is defined as the percentage of power output to power input. Then, the Intensity is calculated as the ratio between sound power (P) and the area scanned, as it is shown in Fig. 2.

$$I = \frac{P}{area}, \tag{2}$$

where the area is calculated as:

$$Area = 4 \cdot \boldsymbol{\pi} \cdot r^2. \tag{3}$$

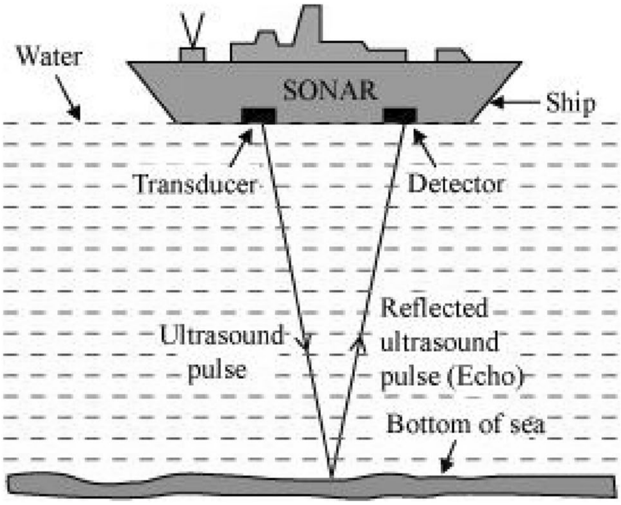

Fig. 1 Ship's SONAR. (source: http://brightmags.com/how-does-sonar-work/)

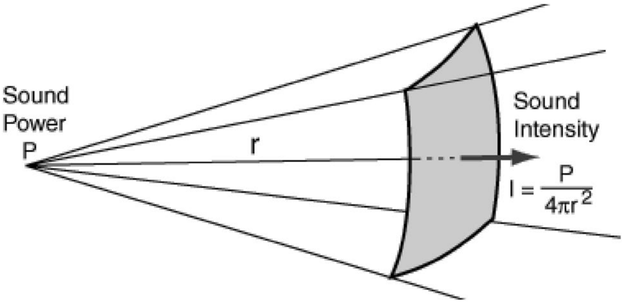

Fig. 2 Sound Intensity depended upon sound power and area. (source: http://hyperphysics.phy-astr.gsu.edu/hbase/Forces/isq.html)

And r is the radius of the imaginary sphere around the ship that is scanned.

As a result, someone can observe that the decrease of intensity I causes an increase of the effective radius r and thus, the area that is scanned. This relation is used also in the proposed scheme.

2.2 The proposed SIO algorithm

We consider each agent $X_i = \{x_1, x_2, x_3, \ldots, x_n\}$ to be a ship, where $i \in 1, 2, \ldots, N$ and N being the maximum number of agents, while n is the maximum number of dimensions of the problem. The number of ships (agents) is predefined at the start of the algorithm, as in every nature-inspired algorithm, for saving computational power. Although generally the more agents there are, the higher is the probability of finding the optimal solution, in the proposed algorithm this is not the case. As it can be seen in the next subsections, the multitude of generated points around each agent's work provides a wider search of the solution space, while the number of agents can remain the same. The strongest feature of the proposed scheme is the wider range of the solution area that is being searched, keeping the number of agents constant.

At start, the position of the agents is initialized somewhere in the solution space; the easiest way to do that is with random way via the normal distribution function, but this can be altered based on the values that every decision variable can take. In this paper, all problems use variables with boundaries, so the function used is:

$$x_i^d = lower\_bound^d + (upper\_bound^d - lower\_bound^d) \cdot rand, \tag{4}$$

where *rand* is a random number between (0,1), $lower\_bound^d$ and $upper\_bound^d$ are the lower and upper bound of each decision variable d, respectively.

Using Eqs. (2) and (3), the initial radius and intensity for every agent is calculated. We set the power input as the fitness of each agent, and so we get:

$$Pe = fit_i\ ,\ i \in \{1, 2, \ldots, N\}. \tag{5}$$

On the other hand, Eq. (1) is reformed as:

$$P = e^{Pe}, \tag{6}$$

in order to transform the fitness' value in positive numbers. This has to be done, because of the usage of logarithm for the rescale of intensity values. Logarithmic equations cannot take negative values, while fitness could be negative in some problems. Thus, we solve this difficulty with a transformation inspired by the physical analogue (i.e. the corresponding idea inspired from nature for the algorithm).

The next steps are repeated until the stopping criteria are met. In the experiments conducted, the stopping criterion is the maximum number of iterations, named "number of scans". For each ship we calculate the fitness function in order to find out the best solution. The best solution is saved and all agents change their intensity based on the solution they have found; if the solution is better from the previous best of the current agent, then the intensity increases and if the opposite exists, then the intensity decreases. That affects also the alteration of the effective radius.

Finally, one more useful mechanism is applied in our scheme. In reality, when a war ship doesn't detect anything in an area, it changes place. An easy way to relocate an agent is to take into consideration the position of the best solution found so far, as described in (Tzanetos and Dounias 2017a, b):

$$x_i^d = best^d + r_0^i \cdot rand, \tag{7}$$

where x_i^d is the position of ith agent in the dth dimension, $best^d$ is the best position found in the current iteration, r_0^i is the effective radius of the ith agent and *rand* a random uniformly distributed number. However, this step is done only when the agent fitness was below the average fitness of all agents. Otherwise, the agent is randomly relocated using the Eq. (4). This method retains the balance between exploration and exploitation. Inspired by the similar concept of mutation rate (Nilsson and Snoad 2002):

$$\mu_{opt} = \frac{1}{\tau}, \tag{8}$$

where τ is the number of generations between environmental changes, we set the limit of time without improvement (or without environmental change) as the 1% of the number of iterations (scans). So, we get (Fig. 3):

$$checkpoint = scans \cdot 0.01. \tag{9}$$

```
Sonar Inspired Optimization
Initialization of ships' position
Initialize effective radius and intensity for every agent
While stopping criteria not met
    If counter = checkpoint
        Relocate the agent
        Recalculate intensity and radius
    End
    Update radius for every agent
    Calculate Intensity for every agent
    While full_scan = false
        Update the rotation angle in every dimension
        Calculate fitness of possible position
        Save the best so far for each agent in the current scan
    End
    Update best position and fitness
    Update intensity and acoustic power output for every agent
End
```

Fig. 3 Pseudocode of the proposed Sonar Inspired Algorithm (SIO)

2.2.1 Intensity parameter

The most important parameter in our algorithm is the intensity parameter. Intensity affects the change of effective radius and thus, the maximum size of area that each agent searches. Intensity is redefined at the end of each iteration based on the solution found by the corresponding agent. Using the exponential function's attribute:

$$I_i^d = I_i^d \cdot e^{magnitude^d}. \tag{10}$$

Magnitude is a way to define the importance of the target found by an agent/ship and is calculated as[1]:

$$magnitude^d = \left[\left(scan\_best_i - best \right) \times 10^{-b} + s \right] \cdot weighted^d, \tag{11}$$

where $scan\_best_i$ is the fitness of the best solution found by the ith agent in the current scan and *best* is the globally best solution at the time. In previous experimentations, *magnitude* had such a value, that the updated Intensity was leading to Infinite values. This problem is solved using the

[1] Equation (11) is formed for minimization problems. In maximization problems, $scan\_best_i$ and best reverse signs.

multiplication of 10^{-b}, where b is the highest power met in fitness by the current population. This parameter is decreased every time that all fitness values are below its current value.

Another issue was that the *magnitude* didn't alter all dimensions respectively. Thus, in this paper a vector that splits the magnitude properly is presented:

$$weighted^d = \frac{accept\_range^d}{\sum_1^d accept\_range^d}, \tag{12}$$

where $accept\_range^d = upper\_bound^d - lower\_bound^d$ for each dimension d of the problem.

To avoid the zero value of magnitude that the agent with the global best solution will return, we add a very small value s.

The Eq. (10) is formed based on the graph of e^x. As it can be shown in Fig. 4, when the value of x for the e^x is below zero, the value of y is lower than one. Thus, if the magnitude is negative (meaning that the agent found better solution), then the intensity will be decreased because it is multiplied with a number lower than the unity. On the other hand, if magnitude is positive, the intensity will increase because is multiplied with a number bigger than unity. And so, as further of the optimum an agent is so much bigger the increase of the intensity will be, resulting into bigger steps to find a better solution.

Although, to transform the high value of I_i to a more useful one, implementing the physical analogue:

$$I_i = 10 \cdot \log \frac{I_i}{I_0}, \tag{13}$$

where I_i is the Intensity of the ith agent and I_0 is the Threshold of Hearing (Lurton 2002), which is defined as:

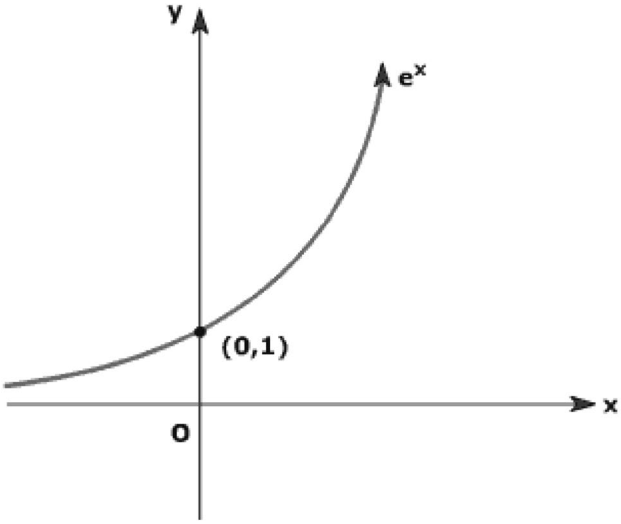

Fig. 4 Graph of y = e^x function

$$I_0 = 10^{-12}\ \text{watts}/_{\text{m}^2} = 10^{-16}\ \text{watts}/_{\text{cm}^2}. \tag{14}$$

In this algorithm the value of Threshold of Hearing I_0 is set to 10^{-12}. This value is fixed no matter what problem is being solved. Previous experimentation has shown that this value is proper for any kind of problem and does not affect the performance of the algorithm.

2.2.2 Effective radius r_0

The initial value of r_0 should be considered, based on the solution space. A small value of the radius will drive the algorithm to perform smaller steps. On the other hand, the choice of a bigger radius will lead to longer jumps of the algorithm towards better optima, but with the risk of overlooking (bypassing) other solutions of desired quality.

By reversing Eq. (3), the effective radius r_0 is calculated as:

$$r_{0_i}^{\ d} = \sqrt{\frac{area_i^{dk}}{4 \cdot \pi}}, \tag{15}$$

where $area_i^k$ is the area scanned by the ith agent for the dth dimension in kth iteration. As we see, this model represents the real relation between these measures; in higher intensity the area scanned is bigger than in lower intensity. Thus, the effective radius r_0 is smaller too. The aim here is to increase the radius, if no better solution is found, so that each agent searches further than its current position.

2.2.3 Full scan loop

In order to search wider areas of the solution space, in each iteration every agent checks the space around it that is limited by its effective radius r_0. This process is called full scan loop, because three steps are repeated until a full cyclic search has been done. Beginning from the angle of 0°, random rotations in each dimension are executed. Each rotation covers a maximum of a° and is calculated as follows:

$$angle^d = angle^d + rand \times a°, \tag{16}$$

where $rand$ is a random number produced from the uniform distribution function and $angle^d$ is the rotation angle in dimension d. When any of $angle^d$ exceeds 360°, the loop is stopped. The vector of angles is converted in vector of movements in every dimension as follows:

$$movement^d = r^d \cdot \cos\left(angle^d\right), \tag{17}$$

where r^d is the random radius inside the cycle that is defined by the effective radius $r_0{}^d$ for the dth dimension of the problem. In Fig. 5 below, an example of the way that the $movement^d$ is calculated in every dimension d is presented. Let the current solution of the dimension be the center of the circle, shown in Fig. 5. This circle is defined by the effective

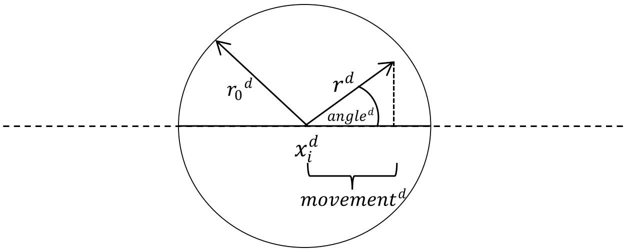

Fig. 5 Example of how the *movement*^{d} is calculated in every dimension d

radius $r_0{}^d$. The possible solutions checked in every loop of the full scan loop are calculated via Eq. (17) and one example is illustrated as the projection of the small arrow on the horizontal line, as shown in Fig. 5.

A decrease of the maximum rotation angle °, leads into smaller rotations and thus, more generated points in every dimension. To decrease computational time, a new addition is presented. Instead of keeping the maximum rotation angle $a°$ same for all agents, they are sorted based on their fitness and according to the sub-group into which they belong, the maximum rotation angle is altered. In this paper, six sub-groups have been used, given the values of maximum rotation angle as follows:

$$\overrightarrow{a°} = [50 \quad 40 \quad 30 \quad 20 \quad 10 \quad 5].$$

With this mechanism, each agent searches more points around its position, while other algorithms' agents check one point per iteration. But, at the same time, each agent searches the number of points that corresponds to its fitness: worst fitness leads to lower sub-group and thus, with smaller maximum rotation angle more points are searched. This provides bigger probability for the agent to jump to a better solution.

The new position is calculated as:

$$x_i^d = movement^d + x_i^d \tag{18}$$

where, x_i^d is the position of the ith agent in the dth dimension and *movement*^{d} is the dth element of the Eq. (17). In each one of the rotation phases, the fitness of the new position is calculated and if it is better than the best found by the current agent, the best position and its fitness are updated.

2.2.4 Correction mechanisms

In order to avoid exceeding the bounds of the solution space, a correction mechanism has also been implemented. If an x_i^d is violating the bound constraints, it is relocated as:

$$x_i^d = lower\_bound^d + \left(upper\_bound^d - lower\_bound^d\right) \cdot \cos\left(x_i^d\right), \tag{18}$$

in order to fulfil the relation: $lower\_bound^d < x_i^d < upper\_bound^d$.

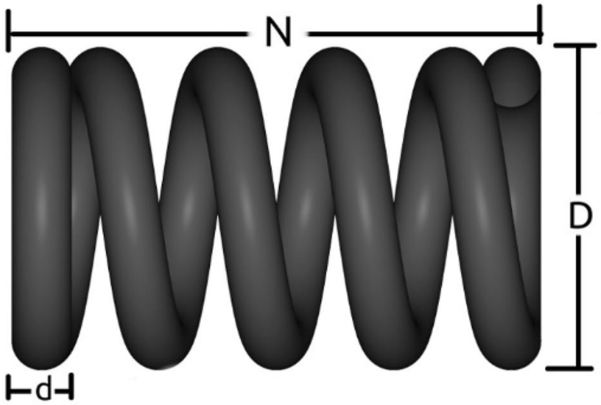

Fig. 6 Tension/compression spring design

The same correction mechanism is used also for the effective radius r_0. If the effective radius r_0 of any agent in any dimension exceeds the *accept_range*^{d} mentioned before, then a new effective radius is generated using the same concept.

3 Engineering applications

In previous work (Tzanetos and Dounias 2017a, b), SIO has been proven to be a good optimization tool. However, as it was stated in further research, the real challenge for a Nature Inspired Intelligent scheme is its application in Real World Problems. Two known constrained Engineering Optimization problems have been chosen: tension/compression spring and welded beam designs.

3.1 Tension/compression spring design

The objective of this problem is to minimize the weight of a tension/compression spring as illustrated in Fig. 6. The minimization process is subject to some constraints such as shear stress, surge frequency, and minimum deflection. There are three variables in this problem: wire diameter (d), mean coil diameter (D), and the number of active coils (N).

Considering the solution as a vector $\vec{x} = \left[x_1, x_2, x_3\right] = [d,\ D,\ N]$, the mathematical formulation of this problem is as follows (Mirjalili et al. 2014):

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

$$\begin{aligned}
\text{Subject to}: \quad & g_1(\vec{x}) = 1 - \frac{x_2^3x_3}{71785x_1^4} \le 0 \\
& g_2(\vec{x}) = \frac{4x_2^2 - x_1x_2}{12566\left(x_2x_1^3 - x_1^4\right)} + \frac{1}{5108x_1^2} \le 0 \\
& g_3(\vec{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0 \\
& g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \le 0,
\end{aligned}$$

given that P = 6000 lb, L = 14 in., E = 30 × 10^6 psi, G = 12 × 10^6 psi.

δ_{max}= 0.25 in., τ_{max}= 13,600 psi, σ_{max}= 30,000 psi.

and variable range:

$$0.05 \leq x_1 \leq 2$$
$$0.25 \leq x_2 \leq 1.3$$
$$2 \leq x_3 \leq 15.$$

Table 1 SIO parameters used in experimentation

Parameter	Value
Number of iterations	1000
Number of population	25
Number of independent runs	40
s	0.0008

3.2 Welded beam design

The objective of this problem is to minimize the fabrication cost of a welded beam as shown in Fig. 7. The constraints of this problem are: the shear stress (τ), the bending stress in the beam (θ), the buckling load on the bar (P_c), the end deflection of the beam (δ) and side constraints, as described below in problem formulation.

The variables of the problem are the thickness of the weld (h), the length of the attached part (l), the height of the bar (t) and the thickness of the bar (b). Considering the solution as a vector $\vec{x} = \left[x_1, x_2, x_3, x_4\right] = [h,\ l,\ t,\ b]$, the optimization problem can be described as (Mirjalili et al. 2014):

$$\begin{aligned}
\text{Minimize}\quad & \mathit{Minimize}\, f\left(\vec{x}\right) = 1.10471x_1^2x_2 + 0.04811x_3x_4\left(14 + x_2\right)\\
\text{Subject to:}\quad & g_1\left(\vec{x}\right) = \tau\left(\vec{x}\right) - \tau_{max} \leq 0\\
& g_2\left(\vec{x}\right) = \sigma\left(\vec{x}\right) - \sigma_{max} \leq 0\\
& g_3\left(\vec{x}\right) = \delta\left(\vec{x}\right) - \delta_{max} \leq 0\\
& g_4\left(\vec{x}\right) = x_1 - x_4 \leq 0\\
& g_5\left(\vec{x}\right) = P - P_c(\vec{x}) \leq 0\\
& g_6\left(\vec{x}\right) = 0.125 - x_1 \leq 0\\
& g_7\left(\vec{x}\right) = 1.10471x_1^2 + 0.04811x_3x_4\left(14 + x_2\right) - 5 \leq 0,
\end{aligned}$$

$$\text{where}\quad \begin{aligned}
\tau\left(\vec{x}\right) &= \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}\\
\tau' &= \frac{P}{\sqrt{2}x_1x_2},\ \tau'' = \frac{MR}{J},\ M = P\left(L + \frac{x_2}{2}\right)\\
R &= \sqrt{\frac{x_2^2}{2} + \left(\frac{x_1 + x_3}{2}\right)^2}\\
J &= 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{2} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}\\
\sigma\left(\vec{x}\right) &= \frac{6PL}{x_4x_3^2},\ \delta\left(\vec{x}\right) = \frac{6PL^3}{x_4x_3^2}\\
P_c\left(\vec{x}\right) &= \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right).
\end{aligned}$$

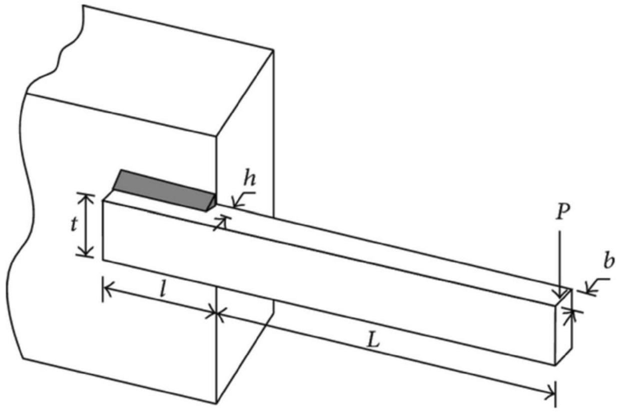

Fig. 7 Structure of welded beam design

given that P = 6000 lb, L = 14 in., E = 30 × 10^6 psi, G = 12 × 10^6 psi.

δ_{max} = 0.25 in., τ_{max} = 13,600 psi, σ_{max} = 30,000 psi.

The variable range is given as follows:

$$0.1 \leq x_1 \leq 2 \overset{g_6}{\Rightarrow} 0.125 \leq x_1 \leq 2$$
$$0.1 \leq x_2 \leq 10$$
$$0.1 \leq x_3 \leq 10$$
$$0.1 \leq x_4 \leq 2.$$

4 Experimental results

All experiments were conducted using Matlab on a 4 GB, 3.6 GHz Intel Core i7 Windows 10 Pro. For every problem, 40 independent runs were done to measure the statistical performance of the algorithm. The results are compared with the corresponding results obtained by various algorithms in literature. In Table 1 below, the parameters used in all experiments are shown.

The performance of the algorithm on the Spring Design Problem can be seen below, in Table 2. Results of other known Nature-inspired metaheuristics, such as Grey Wolf Optimizer (GWO) (Mirjalili et al. 2014) and co-evolutionary Particle Swarm Optimization (CPSO) (He and Wang 2007), are used as benchmarks. Also, results from heuristic methods [Evolutionary Strategy (ES) (Mezura-Montes and Coello 2008), Genetic Algorithm (GA) (Coello 2000), Harmony Search (HS) (Mahdavi et al. 2007), co-evolutionary Differential Evolution (CDE) (Huang et al. 2007)] and mathematical approaches [numerical optimization technique (Arora 2004) and mathematical optimization technique (Belegundu and Arora 1985)] are used as benchmarks. SIO managed to outperform the other algorithms and provided design points that do not violate any of the constraints.

Table 3 contains the comparison of results for the Welded Beam Design problem. Previous results from Grey Wolf Optimizer (GWO) (Mirjalili et al. 2014), Genetic Algorithm (GA) (Coello Coello 2000; Deb 1991, 2000), Harmony Search (HS) (Lee and Geem 2005) and mathematical approaches (Ragsdell and Phillips 1976) can be seen. SIO overcomes all other schemes except of GWO. Although, it is slightly worse, there are no statistical results to compare the performance between them.

In both problems, SIO successfully find high quality near optimal solutions without violating any constraint. In Tension/Compression Spring Design problem overcame the other algorithms. In Welded Beam Design problem was a lot better than most of the other scheme and slightly worse than GWO, but the lack of statistical performance of GWO does not give the chance to export safe conclusions. Also, the statistical results show that SIO consists a powerful optimization tool, which manages to provide optimal or near optimal solutions.

Table 2 Comparison of results for tension/compression spring design problem

Algorithm	Optimum variables			Optimum weight	Mean	Std
	d	D	L			
SIO	0.051354	0.405225	7.554277	**0.0102104**	**0.0116751**	7.8400e−3
GWO	0.05169	0.356737	11.28885	0.012666	N/A	N/A
CPSO	0.051728	0.357644	11.244543	0.0126747	0.012730	5.1985e−5
ES	0.051989	0.363965	10.890522	0.0126810	0.012923	5.9200e−4
GA	0.051480	0.351661	11.632201	0.0127048	0.012769	5.9000e−5
HS	0.051154	0.349871	12.076432	0.0126706	N/A	N/A
CDE	0.051609	0.354714	11.410831	0.0126702	0.012703	2.7000e−5
Mathematical optimization	0.053396	0.399180	9.1854000	0.0127303	N/A	N/A
Constrained correction	0.050000	0.315900	14.250000	0.0128334	N/A	N/A

Bold values show the best value found within all schemes, which coped with the corresponding problem

Table 3 Comparison of results for welded beam design problem

Algorithm	Optimum variables				Optimum cost	Mean	Std
	h	l	t	b			
SIO	0.331482	2.017442	9.045926	0.208865	1.7621466	2.075521	0.449052
GWO	0.205676	3.478377	9.03681	0.205777	**1.72624**	N/A	N/A
GA	N/A	N/A	N/A	N/A	1.8245	N/A	N/A
GA	N/A	N/A	N/A	N/A	2.3810	2.383	N/A
GA	0.2489	6.1730	8.1789	0.2533	2.4331	N/A	N/A
HS	0.2442	6.2231	8.2915	0.2443	2.3807	N/A	N/A
Richardson's random	0.4575	4.7313	5.0853	0.6600	4.1185	N/A	N/A
Simplex	0.2792	5.6256	7.7512	0.2796	2.5307	N/A	N/A
Davidon-Fletcher-Powell	0.2434	6.2552	8.2915	0.2444	2.3841	N/A	N/A
Linear approximation	0.2444	6.2189	8.2915	0.2444	2.3815	N/A	N/A

Bold values show the best value found within all schemes, which coped with the corresponding problem

5 Conclusions and future research

In this paper, a novel meta-heuristic algorithm named SIO (Sonar Inspired Optimization) was presented and tested in real world engineering optimization problems. Three new modifications were implemented to improve the performance of the algorithm; the maximum rotation angle is auto-tuned based on the fitness of the solution, the magnitude respectively alters the Intensity in every dimension and the relocation of the agents is done in a smarter way, so that exploration and exploitation balance remains until the end of the algorithm. The very limited parameterization that SIO needs, makes this algorithm useful for a wide range of problems. The most important feature of SIO is the balance between exploration and exploitation, which is achieved via the relocation rule and the full scan loop, respectively. As the results from this work show, SIO is proven to handle efficiently engineering optimization problems. What is more, the first attempt to solve problems with constraints gave promising results.

SIO was tested in known constrained engineering optimization problems, namely the Tension/Compression Spring Design problem and the Welded Beam Design problem. Also, compared with other nature-inspired metaheuristics, heuristics and mathematical approaches was found statistically comparable or superior in most of the cases. The lack of statistical analysis of the performance of competitive algorithms make it difficult to extract further conclusions. Nevertheless, the corresponding performance of SIO showed that this algorithm is consistent and provides optimal or near optimal solutions.

Furthermore, the main SIO advantages should be highlighted; the minimal parameterization and the higher exploration of the solution space. Especially the second feature, SIO's agents search many possible positions around their current location in each iteration, while in other algorithms agents check only one new point. Additions and modifications of the mechanisms of the algorithm are presented here, resulting in improved performance of the algorithm.

Currently, work is underway on the application of Sonar Inspired Optimization in Decision Engineering problems. Experiments have already taken place in this direction, in financial and industrial engineering problems. A new hybrid scheme which contains SIO as a component is underway, too. Application of SIO in other Engineering and Structural Problems will take place in future.

References

Arora J (2004) Introduction to optimum design. Academic Press, Cambridge

Belegundu AD, Arora JS (1985) A study of mathematical programming methods for structural optimization. Part I: theory. Int J Numer Meth Eng 21(9):1583–1599

Birbil Şİ, Fang SC (2003) An electromagnetism-like mechanism for global optimization. J Gl Opt 25(3):263–282

Chiong R (ed) (2009) Nature-inspired algorithms for optimisation (vol 193). Springer, Berlin

Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41(2):113–127

Coello Coello CA (2000) Constraint-handling using an evolutionary multiobjective optimization technique. Civ Eng Syst 17(4):319–346

Crawford B, Valenzuela C, Soto R, Monfroy E, Paredes F (2013) Parameter tuning of metaheuristics using metaheuristics. Adv Sci Lett 19(12):3556–3559

Deb K (1991) Optimal design of a welded beam via genetic algorithms. AIAA J 29(11):2013–2015

Deb K (2000) An efficient constraint handling method for genetic algorithms. Comput Methods Appl Mech Eng 186(2):311–338

Fallahi M, Amiri S, Yaghini M (2014) A parameter tuning methodology for metaheuristics based on design of experiments. Int J Eng Tech Sci 2(6):497–521

Fister I Jr, Yang XS, Fister I, Brest J, Fister D (2013) A brief review of nature-inspired algorithms for optimization. Elektrotehniški vestnik 80(3):116–122

He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20(1):89–99

Huang FZ, Wang L, He Q (2007) An effective co-evolutionary differential evolution for constrained optimization. Appl Math Comput 186(1):340–356

Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213(3):267–289

Kennedy JF, Kennedy J, Eberhart RC, Shi Y (2001) Swarm intelligence. Morgan Kaufmann, Burlington

Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput Methods Appl Mech Eng 194(36):3902–3933

Liu J, Tsui KC (2006) Toward nature-inspired computing. Commun ACM 49(10):59–64

Lurton X (2002) An introduction to underwater acoustics: principles and applications. Springer Science & Business Media, Berlin

Mahdavi M, Fesanghary M, Damangir E (2007) An improved harmony search algorithm for solving optimization problems. Appl Math Comput 188(2):1567–1579

Marrow P (2000) Nature-inspired computing technology and applications. BT Technol J 18(4):13–23

Mezura-Montes E, Coello CAC (2008) An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. Int J Gen Syst 37(4):443–473

Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61

Nasir ANK, Tokhi MO, Abd Ghani NM, Raja Ismail RMT (2012) Novel adaptive spiral dynamics algorithms for global optimization. In: 11th IEEE international conference on cybernetic intelligent systems (CIS), pp 99–104. IEEE Press, Ireland

Nilsson M, Snoad N (2002) Optimal mutation rates in dynamic environments. Bull Math Biol 64(6):1033–1043

Ragsdell KM, Phillips DT (1976) Optimal design of a class of welded structures using geometric programming. J Eng Ind 98(3):1021–1025

Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248

Shah-Hosseini H (2009) The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm. Int J Bio-Insp Comp 1(1–2):71–79

Tzanetos A, Dounias G (2017a) A new metaheuristic method for optimization: sonar inspired optimization. In: International conference on engineering applications of neural networks (pp 417–428). Springer, Cham

Tzanetos A, Dounias G (2017b) Nature inspired optimization algorithms related to physical phenomena and laws of science: a survey. Int J Artif Intell Tools 26(06):1750022

Vassiliadis V, Dounias G (2009) Nature–inspired intelligence: a review of selected methods and applications. Int J Artif Int Tools 18(04):487–516

Yang XS (2010) Nature-inspired metaheuristic algorithms. Luniver press, Frome

Yang XS (2012) Nature-inspired metaheuristic algorithms: success and new challenges. J Comput Eng Inf Technol 1:1–2

Yang XS, Deb S, Loomes M, Karamanoglu M (2013) A framework for self-tuning optimization algorithm. Neural Comp App 23(7–8):2051–2057

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.