www.czasopisma.pan.pl PAN POLSKA AKADEMIA NAUK www.journals.pan.pl

PAN ARCHIVES OF ACOUSTICS
Vol. **44**, No. 1, pp. 137–151 (2019)
DOI: 10.24425/aoa.2019.126360

Multi-Layer Perceptron Neural Network Utilizing Adaptive Best-Mass Gravitational Search Algorithm to Classify Sonar Dataset

Mohammad Reza MOSAVI[(1)*], Mohammad KHISHE[(1)], Mohammad Jafar NASERI[(2)]
Gholam Reza PARVIZI[(3)], Mehdi AYAT[(1)]

[(1)] *Department of Electrical Engineering*
Iran University of Science and Technology
Narmak, Tehran 16846-13114, Iran
*Corresponding Author e-mail: m_mosavi@iust.ac.ir

[(2)] *Department of Electronic and Communication Engineering*
University of Marine Sciences
Nowshahr, Iran

[(3)] *Department of English*
Alborz Institute for Higher Education
Qazvin, Iran

(*received August 28, 2016; accepted October 22, 2018*)

In this paper, a new Multi-Layer Perceptron Neural Network (MLP NN) classifier is proposed for classifying sonar targets and non-targets from the acoustic backscattered signals. Besides the capabilities of MLP NNs, it uses Back Propagation (BP) and Gradient Descent (GD) for training; therefore, MLP NNs face with not only impertinent classification accuracy but also getting stuck in local minima as well as low-convergence speed. To lift defections, this study uses Adaptive Best Mass Gravitational Search Algorithm (ABGSA) to train MLP NN. This algorithm develops marginal disadvantage of the GSA using the best-collected masses within iterations and expediting exploitation phase. To test the proposed classifier, this algorithm along with the GSA, GD, GA, PSO and compound method (PSOGSA) via three datasets in various dimensions will be assessed. Assessed metrics include convergence speed, fail probability in local minimum and classification accuracy. Finally, as a practical application assumed network classifies sonar dataset. This dataset consists of the backscattered echoes from six different objects: four targets and two non-targets. Results indicate that the new classifier proposes better output in terms of aforementioned criteria than whole proposed benchmarks.

Keywords: Multi-Layer Perceptron Neural Network; Adaptive Best Mass Gravitational Search Algorithm; sonar; classification.

1. Introduction

Classification of underwater targets from the acoustic backscattered signals includes discrimination between target and non-target objects as well as the description of background clutter (Mosavi *et al.*, 2015). There are a lot of components that complicate this procedure such as: non-repeatability and alteration of the target signature with aspect angles, ranging and grazing angle (Mosavi *et al.*, 2015), challenging natural and man-made clutter (Pailhas *et al.*, 2012), effects of latitude and longitude (Williams, Fakiris, 2014), highly variable and reverberant working environment (Pan *et al.*, 2014; Zhou, Zhang, 2005), dependency on the water's temperature, the salinity, the depth (Aubry *et al.*, 2012) and the lack of any pre-knowledge about the form and the geometry of the non-target (Chu, Stanto, 2010).

Having considered mentioned complexities, many efforts have been made to propose effective classifier in this field (Fei *et al.*, 2015; Kumar *et al.*, 2015; Blumrosen *et al.*, 2014; Das *et al.*, 2013; Pearce, Bird, 2013). Recently, using of Multi-Layer Perceptron (MLP) Neural Networks (NNs) is taken

into consideration for their significant outcomes (Cui *et al.*, 2015; Han, Wang, 2014; Souza *et al.*, 2016; Yegireddi, 2015). High accuracy, versatility, inherently parallel structure, which is very useful in hardware implementation and real-time processing, are some of the distinguished feature of MLP NNs in the sonar dataset classification, all of which encourage researchers to use assumed classifier.

The most challenging part of the MLP NNs is training (Mirjalili *et al.*, 2014). In most applications, MLP NNs optimized by Back Propagation (BP) (Auer *et al.*, 2008) or standard BP algorithms (Moody, Darken, 1989) are used as training methods. BP algorithm is based on the gradient that has few drawbacks such as slow convergence rate (Karayiannis, 1999) and the use of a small searching area (Liu *et al.*, 2014); therefore, it is not reliable to be used as practical applications. As a result, the uses of innovative and new meta-heuristic algorithms have become more typical to resolve above mentioned problem through the real dataset in recent years.

Regardless of differences between various meta-heuristic methods, dividing the search process into two discrete phases, exploration and exploitation, is the joint feature of them. Finding a proper equivalent between these two phases is a real challenge when considering random identity of the meta-heuristic methods.

Gravitational Search Algorithm (GSA) is one of the methods that has fine performance within review. This algorithm is inspired by natural gravitational forces between masses. GSA has indicated a more superior performance in the exploration phase than the other well-known heuristic algorithms such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Genetic Algorithm (GA) (Sabri *et al.*, 2013; Pei *et al.*, 2014; Doraghinejad *et al.*, 2014; Rahmani *et al.*, 2013; Li *et al.*, 2014; Parsazad *et al.*, 2013; Li *et al.*, 2012; Han, Chang, 2012; Zhang *et al.*, 2012; Li, Duan, 2012; Gonzalez *et al.*, 2011; Sarafrazi *et al.*, 2011). However, as the number of iterations increases, the search process in GSA slows down. Considering the increasing effect of the fitness function on mass, masses get heavier over the iterations. This phenomenon prevents masses from quickly exploiting the best solutions in subsequent iterations. Therefore, it appears to be the main drawback of this algorithm facing height dimension real-world engineering problems.

When review the literature, combining meta-heuristic algorithms with GSA is a usual method to reach a better exploitation ability, the same as in the hybrids PSO-GSA (Mirjalili, Hashim, 2010; Gu, Pan, 2013), Combined the Quantum inspired Binary GSA (QBGSA) with K-Nearest Neighbor (K-NN) (Han *et al.*, 2013), Hybrid of GSA and Free Search Differential Evolution (FSDE) (Liu, Ma, 2013), Hybrid GA and GSA (HGA-GSA) (Sun, Zhang, 2013), hybrid of GSA and Artificial Bee Colony (ABC) (Guo, 2012), K-Means (KM) (Hatamlou *et al.*, 2012), Local Search (LS) (Noman, Iba, 2008; Chen *et al.*, 2005) and Gradient Descent (GD) (Chen *et al.*, 2007; Meuleau, Dorigo, 2002) have been utilized. All of these hybrid methods are able to improve performance, but they cause further computational cost. Normally, merging two algorithms, the time complexity is worse than any single because both algorithms should be run either in sequential or parallel way. Therefore, these unavoidable problems should be considered, particularly for real-world engineering problems such as sonar dataset classification.

This paper attempts to enhance the exploitation phase of GSA with a sorely low-cost technique in favour to sonar dataset classification. This problem has been discussed indirectly or directly in previous researches. Sinaie (2010) dissolved the Traveling Salesman Problem (TSP) with GSA and NN sequentially. In this method, GSA and NN accomplished the exploration and exploitation phases, respectively. For improving the convergence speed of GSA Shaw *et al.* (2012) used an opposition-based training. Chen *et al.* (2011) used GSA accompanied by a multi-type local improvement scheme as a local search operator.

In 2012, Zhang *et al.* (2012) proposed a new Immune Gravitation Optimization Algorithm (IGOA) to solve the slow convergence rate in exploitation phase. Hatamlou *et al.* (2011) proposed a successive algorithm in such a way that it used GSA for finding a near-optimal solution and another meta-heuristic algorithm to improve the solutions taken by GSA. Li and Zhou (2011) and Mirjalili *et al.* (2012) merged individual and social thinking of PSO to GSA to improve exploitation of GSA.

All these studies indicate that GSA suffers from poor exploitation phase. Nevertheless, none of them discuss this problem state in detail. In this paper, the major reason for the GSA's poor exploitation is being further investigated in detail; a comprehensive low-cost solution is proposed exploiting adaptive coefficients, so that these coefficients are adaptively updated based on the best mass obtained as far while considering classification rate of MLP NN. In other words, by using the classification rate of MLP NN, the best mass of GSA is elected. Then, the elected best mass directs other masses to the global optimum. From this point forward, this method is named "ABGSA". To administer a comprehensive test, in addition to sonar dataset, the researchers investigate the performance of designed adaptive classifier on the two other benchmark datasets, each with a different dimension which any of them has a different dimension. Utilizing MLP NN and ABGSA is based upon following reasons:

- MLP NN is fully capable of working with data, disable to discriminate in linear form. In the

meantime, sonar datasets are indiscriminate in linear form and meet the needs of high dimension classifier.

- Targets, non-targets and clutter have the same features; therefore, a pattern capable of discovering searching space completely must be chosen.
- Extra-ability in discovering is the strong point of the GSA, stronger than the other meta-heuristic algorithms in such a way that searching space gets searched by masses of various weights completely.
- Low speed in exploration phase is the weak point of the GSA. This defection has been resolved by adaptive best mass within ABGSA.
- Finally, the designed classifier should be able to classify objects in real-time. MLP NN and ABGSA have inherently parallel structures; therefore by hardware implementation of these two structures on FPGA platform, the researchers can reach the goals in real-time classification.

MLP NN is explained in the Sec. 2. MLP NN training algorithms defined as GSA and ABGSA is analysed in the Sec. 3. In the Sec. 4, MLP NN is taught by ABGSA and the results of the simulations are elaborated and finally in the Sec. 5 conclusions are presented.

2. Multi-Layer Perceptron Neural Network

Figure 1 shows a MLP NN with two layers. R is the number of input nodes, S_1 is the number of hidden neurons and S_2 is the number of output neurons. As it can be seen, there exists one-way connection between nodes in a MLP NN categorized under the feed-forward NNs family. The MLP NN outputs are calculated using the Eq. (1):

$$\mathbf{n}_1 = \mathbf{IW} \times \mathbf{P} + \mathbf{b}_1, \tag{1}$$

where **IW** is the connection weight matrix from the input nodes to the neurons of hidden layer, $\mathbf{b}_1$ is the neuron's bias matrix (in hidden layer), and **P** is the node's input matrix. Each hidden layer neuron's output is calculated using a sigmoid function as given in Eq. (2):

$$a_1 = \text{Sigmoid}(n_1) = \frac{1}{1+\exp(-n_1)}. \tag{2}$$

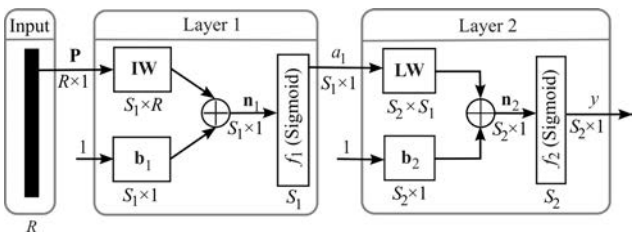

Fig. 1. A MLP NN with one hidden layer.

Final outputs after calculating the hidden nodes output can be defined as:

$$\mathbf{n}_1 = \mathbf{IW} \times \mathbf{P} + \mathbf{b}_2, \tag{3}$$

$$y = \text{Sigmoid}(n_2) = \frac{1}{1+\exp(-n_2)}, \tag{4}$$

where **LW** is the connection weight matrix from the hidden layer to the output layer and b_2 is the neuron's bias matrix in the output layer. The most important parts of the MLP NNs are the connection weights and neuron's biases. As it can be seen from above equations, the final output of the network is defined by connection weights and neuron's biases. Training the MLP NN includes finding the best values for connection weights and neuron's biases, so that the specific inputs will be obtained from desired outputs.

3. MLP NN training algorithms

In this part, ABGSA which exerted for MLP NN training will be explained. In doing so, first GSA will be explained in brief.

3.1. Gravitational Search Algorithm

GSA is inspired by the Newton gravity law. GSA does searching by the use of factor set (chosen solution) which have the masses appropriate with their fitness function. Along iteration, masses absorb each other by gravitational forces in between's. The heavier weigh has more gravitational force. Therefore, the heaviest mass probably closes to generic optimal. It absorbs the other masses in proportion to their distances. In this algorithm, according to Eq. (5) each mass takes the location in search space as follows (Rashedi *et al.*, 2009):

$$X_i = \left(\chi_i^1, ..., \chi_i^d, ..., \chi_i^n\right), \qquad i = 1, 2, ..., N, \tag{5}$$

where N equals to the number of masses, n indicates problem dimension, and χ_i^d posits i-th factor in d-th dimension of the algorithm. The algorithm starts with putting all factors in the searching space randomly. Along all ranges gravitational force from the j-th factor to i-th factor is defined within definite time t at the Eq. (6):

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t)+\varepsilon}(\chi_j^d(t) - \chi_i^d(t)), \tag{6}$$

where $M_{aj}(t)$ indicates active gravitational mass relating to j-th factor and $M_{pi}(t)$ indicates passive gravitational mass relating to i-th factor, $G(t)$ is gravity constant at t time, ε indicates a diminutive constant, and $R_{ij}(t)$ indicates Euclidean distance between i-th factor and j-th factor.

To develop discovering in starting iterations and exploiting in ending iterations G is designed with adaptive compliance. Therefore, it increases along iterations. On the other hand, G stimulates searching factors to the movement with large pitches, but they are limited to slack movement with final iteration. The gravity factor (G) and the Euclidean distance between i-th factor and j-th factor is calculated through the following formula:

$$G(t) = G_0 \times \exp\left(-a \times \frac{iter}{maxiter}\right), \tag{7}$$

$$R_{ij}(t) = \|X_i(t), X_j(t)\|_2, \tag{8}$$

where α indicates decreasing coefficient, G_0 indicates initial gravity constant, and *iter* indicates the number of present iterations. Also, *maxiter* indicates maximum iteration numbers. Within a problem space with monotonous dimension for d, total force which exerts to i-th factor is calculated by the Eq. (9):

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j F_{ij}^d(t), \tag{9}$$

where $rand_j$ indicates a random figure within definite range [0, 1]. To pick random pitch movement alongside gravity force from finally gained force and factor, random part is put in the formula. This case gets more miscellaneous treatments and it helps while moving toward search factor. The Newton's rule has been used within algorithm where acceleration of the mass equals to exerted force divided by correspondent mass. Therefore, the acceleration of all factors is calculated by Eq. (10):

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}, \tag{10}$$

where d indicates problem dimension, t indicates a certain time, M_{ii} indicates inertia mass of i-th factor, and the speed and situation of the factors are calculated by the Eqs (11) and (12), respectively:

$$\nu_i^d(t+1) = rand_i \times \nu_i^d(t) + a_i^d(t), \tag{11}$$

$$\chi_i^d(t+1) = \chi_i^d(t) + \nu_i^d(t+1). \tag{12}$$

As can be resulted from the Eqs (11) and (12), current speed of the factor is defined as the part of the last speed to which acceleration is added ($0 \leq rand_i \leq 1$).

Since the masses factors are defined by the fitness function, a factor with the heaviest mass is the most appropriate factor. According to the above equations, the heaviest factor gets the highest gravity force and the lowest kinesis. There is, nonetheless, a direct relation between mass and fitted function, a normalized method for scaling mass has been taken into consideration, according to following equations:

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)}, \tag{13}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}, \tag{14}$$

where $fit_i(t)$ indicates proportional value of i-th factor at the time of t-th. Also, $best(t)$ is the fittest agent at time t, and $worst(t)$ is the weakest agent at time t as follows:

$$best(t) = \min_{j\in\{1,\ldots,N\}} fit_j(t), \tag{15}$$

$$worst(t) = \max_{j\in\{1,\ldots,N\}} fit_j(t). \tag{16}$$

Within GSA, first all factors get installed by random values. Along iterations, speeds and positions are defined by Eqs (11) and (12). Yet, the other parameters like gravity constant and masses are defined by Eqs (7) and (14). Finally, GSA ends up with assessing the last factor.

3.2. Adaptive Best Mass Gravitational Search Algorithm

Within GSA gravity constant (G) sets masses speedup in such a way that with that solutions vary their locations in solution space. According to (Rashedi *et al.*, 2009), great G leads into high acceleration caused by fast kinesis of the mass within the primitive iterations. Yet, G repeatedly increases and it helps GSA during exploration period (Mirjalili, Lewis, 2014). Therefore, exploration phases coincidence with weaker gravity force and heavier masses. Unfortunately, the heavier masses with slow kinesis and weaker gravity force decreases convergence speed remarkably. Therefore, it seems that GSA suffers from slow searching speed led by these factors in exploration phase.

As it can been seen in Fig. 2a, M_1 and M_3 masses get absorbed from M_2 mass in $t+1$ and $t+2$ iterations. Yet, these two masses absorb M_2 and move it away from minimal value slowly. The particles approach to minimal values any way, but they compulsorily are not able to accelerate moving toward optimum values. It is remarkable to say that GSA has no memory to save the best ever obtained solution. Therefore, the best solution might miss so that the best mass gets absorbed with the other less fitted masses.

The main idea of the ABGSA is to save and use the best mass location for accelerating in exploitation phase. Figure 2b indicates using the best solution for accelerating masses movement toward optimal values. As it can be seen therein, the *Gbest* element exerts additional speed element toward the last known location for the best mass. By using this, '*Gbest* external force' prevents masses from falling to the local

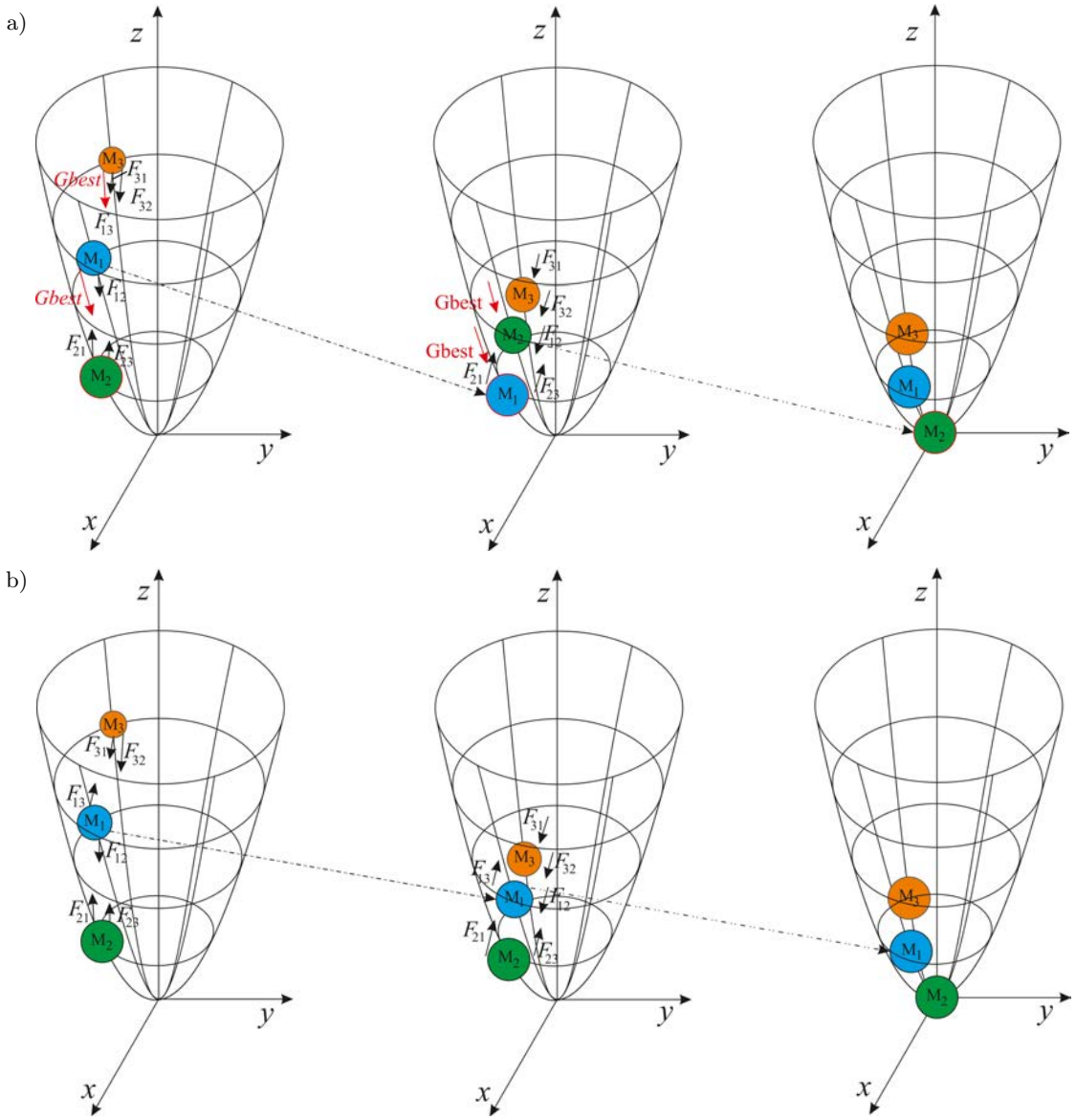

Fig. 2. Masses movement: a) GSA method and b) ABGSA method.

optimums. There are two advantages of this method: first, accelerating particle movement toward the location of the best mass- it helps them surpassing and reaching for the best mass in the next iteration second, the best recent obtained solution gets saved for using the next iterations. The presented method is simulated via Eq. (17):

$$V_i(t+1) = rand \times V_i(t) + c'_1 \times ac_i(t) + c'_2 \times (gbest - X_i(t)). \quad (17)$$

In this equation, $V_i(t)$ indicates the speed of i-th factor in t-th iteration, c'_1 and c'_2 indicate the coefficient of acceleration, *rand* is a random figure between zero and one, $ac_i(t)$ indicates the acceleration of the i-th factor in t-th iteration, and *Gbest* is the position of the best updated and obtained solution. In Eq. (17), the second element $(c'_2 \times (gbest - X_i(t))$ takes the responsibility of masses toward the best mass that obtains till t time. The distance of each mass from the best mass can be calculated by $gbest - X_i(t)$. The ultimate force toward the best mass in random part is the distance by which is defined c'_2 that is ultimate external force exerted over each mass. In any iteration, the position of the masses are updating via Eq. (18):

$$X_i(t+1) = X_i(t) + V_i(t+1). \quad (18)$$

The constant element added in this method, have negative effect at exploring phase. As it can be seen in Fig. 3, adaptive values for c'_1 and c'_2 are used to avoid reducing exploration capability in new speed updating. When algorithm reaches for exploitation phase, c'_1 decreases and c'_2 increases by adaptive way in such a manner that masses intend to speed up toward the best so-

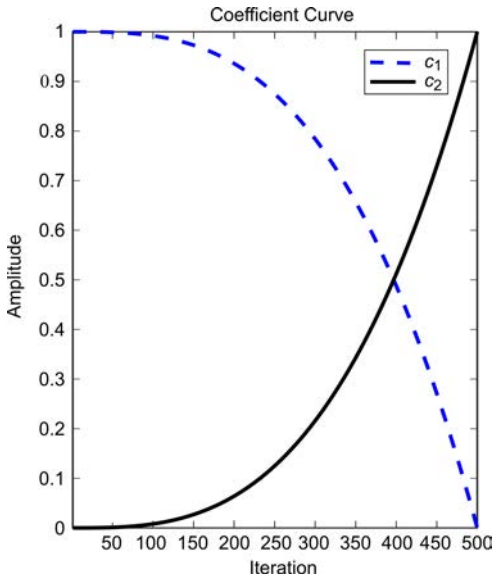

Fig. 3. c_2' and c_1' coefficient curve.

lution. There is no border between exploration phase and exploitation phase in evolutionary algorithms.

The best choice to gradually transition between two phases is adaptive method. Additionally, this adaptive approach focuses on exploring in the first iterations and exploiting in the last iterations.

4. Training MLP NN using ABGSA

Generally, there are three methods to present combination of passive elements: a) vector, b) matrix, and c) binary state (Mirjalili, 2015). As an example of encoding method, the final vector of MLP NN is shown in Fig. 4 given by Eq. (19)

$$Mass = [w_{15}\ w_{16}\ ...\ w_{810}\ w_{910}\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6]. \quad (19)$$

The general trend in the training MLP NN by ABGSA is shown in Fig. 5.

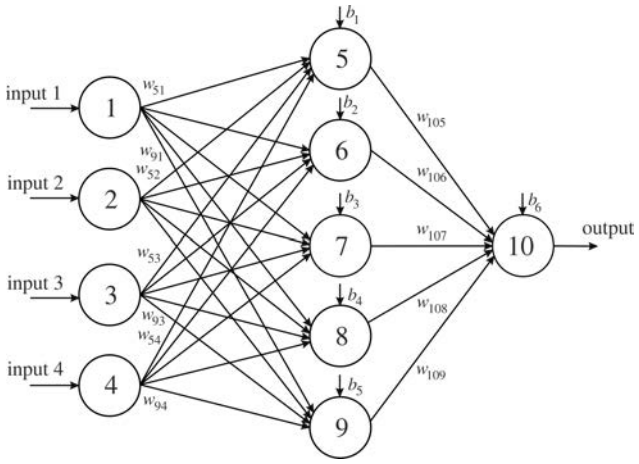

Fig. 4. MLP NN (4, 5, 1) structure.

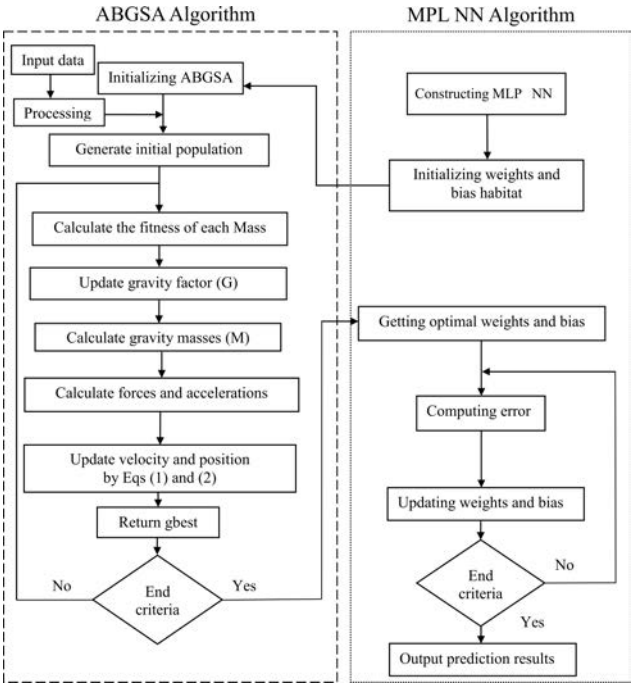

Fig. 5. Training an MLP NN using ABGSA.

4.1. Setting and Testing Parameters

To test ABGSA algorithm usefulness in training MLP NN, this network was not only trained by ABGSA algorithm, but also by GD, GSA, PSO, GA and PSOGSA. Designed classifiers get exerted on data of sonar, Iris, and Lenses (as described in Table 1). Initial values and parameters of this algorithm are shown in Table 2. The obtained identity of classifiers in terms of classification rate, convergence speed, and stuck avoiding in local optimum were tested.

Each network was tested 10 times. To do a fair comparison, all algorithms are stopped when the maximum number of iterations reaches 250. Since there is no established standard for choosing the number of hidden nodes in classification of the datasets, based on the structure of MLP NNs, the proposed method in (Mirjalili, 2015) and Eq. (20) is used

$$H = 2 \times N + 1, \quad (20)$$

where N represents the number of inputs and H indicates the number of hidden nodes. The average (AVE) and Standard Deviation (STD) of Mean Square Error (MSE) in table of results are calculated. To reach a greater capability of the algorithm and to avoid local minimum, the value of AVE $\pm$ STD must be lower. AVE shows the average of MSE over 10 runs and by reaching a lower value for AVE the greater capability of the algorithm, to avoid reaching local optimum and finding solutions near the global optimum, is indicated.

Average value of the two algorithms can be equal although their performances in finding the global op-

Table 1. Datasets used in the paper.

Name	Data type	Default task	Attribute characteristics	# Attributes	# Instances	Year
Iris	Classification	Multivariate	Real	4	150	1988
Lenses	Classification	Multivariate	Categorical	4	24	1990
Sonar	Classification	Multivariate	Real	23	200	2015

Table 2. Parameters and initial values of the applied algorithms.

Algorithm	Parameter	Value
PSO	Topology	Fully connected
	Cognitive constant (C1)	1
	Social constant (C2)	1
	Inertia constant (w)	0.3
	Maximum number of iterations	250
GD	Population size	200
	Learning factor (η)	0.01
ABGSA	c_1'	$(-2t^3/T^3)+2$
	c_2'	$2t^3/T^3$
	G_0	1
	Maximum number of iterations	250
GSA	Number of masses	70
	α	20
	Number of masses	70
	Gravitational constant	1
	Maximum number of iterations	250

timum in each run are different. Because of this reason, AVE is not a good parameter alone and another parameter like STD will help specify the dispersion of results. To have a lower dispersion of results, the STD must be lower. Therefore, it is normal to show the ability of an algorithm in avoiding local minimum by adding the two mentioned parameters together (AVE ± STD).

According to Derrac *et al.* (2011), statistical tests are needed to have an adequate evaluation of performance of meta-heuristic algorithms. Comparing algorithms according to their means and standard deviations values is not enough and a statistical test is needed to demonstrate a remarkable improvement of a new algorithm in comparison to the other existing algorithms to solve a particular problem. In order to see whether the results of ABGSA differ from PSO, GSA, GD, GA and PSOGSA in a statistically significant way, a non-parametric statistical test, Wilcoxon's rank-sum test (Wilcoxon, 1945) was administered at 5% significance level. The calculated p-values in the Wilcoxon's rank-sum are given in the results as well. In Tables 3–5, N/A demonstrates "Not Applicable" which means that the corresponding algorithm cannot be compared with itself in the rank-sum test. Conventionally, p-values less than 0.05 are considered as strong evidence against the null hypothesis. Note that p-values greater than 0.05 are underlined in Tables 3–5. Another comparative measure shown in the results is classification rates.

4.2. Iris dataset

The dataset includes four features of three types of flowers which are named Setosa, Versicolor, and Virginica. These four features are the latitude and longitude of the leaves and bowl petals. For each type of flower, there are fifty samples. So, there are wholly 150 samples and each of which has four features (Gorman, Sejnowski, 1998). MLP NN solved this dataset with the structure (4, 9, 3).

Table 3 shows the results of training algorithms to solve this dataset. On account of this dataset, the results show that ABGSA based-classifier has a better performance to avoid reaching local minimum in comparison to the other algorithms; according to the results of AVE, STD, and p-values. In addition, percentage of classification for the samples was approximately 96.2667% that was better than the other algorithms. The convergence curves for this dataset are shown in Fig. 6a. It demonstrates that the convergence curve of ABGSA classifier is better than the other algorithms.

Table 3. Experimental results for Iris dataset.

Algorithm	MSE (AVE ± STD)	p-values	Classification rate [%]
MLPABGSA	0.0291 ± 0.0667	N/A	96.2667
MLPPSO	0.0640 ± 0.2666	0.0079	93.6667
MLPGSA	0.0776 ± 0.0072	0.0079	92.4667
MLPGD	0.1025 ± 0.0323	6.39e–05	88.9333
MLPGA	0.0899 ± 0.1236	6.39e–05	89.3333

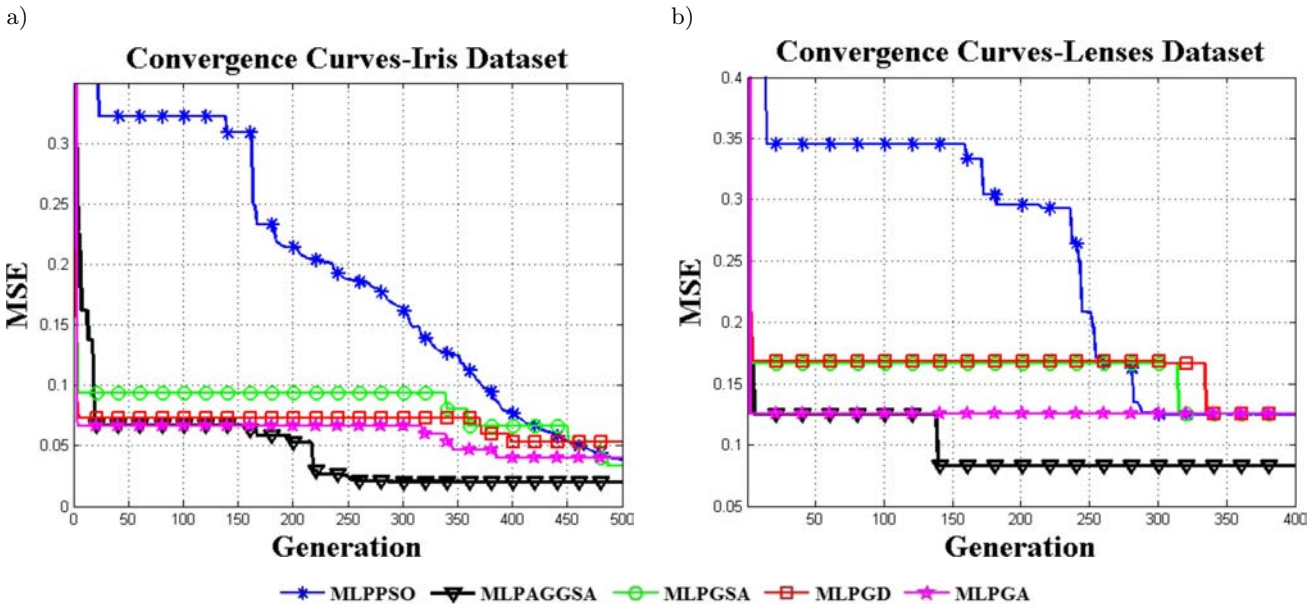

Fig. 6. Convergence curves of the algorithms: a) Iris dataset and b) Lenses dataset.

4.3. Lenses dataset

Having determined the attributes of the clients, the lenses dataset attempted to predict whether a person would need a soft-contact lenses or hard-contact one or no contact. The dataset has four attributes as patient age, spectacle prescription, astigmatism notion, and tear production rate data. There is also an extra attribute which is a three-value class to give appropriate lenses prescription to a patient (hard-contact lenses, soft contact lenses, no contact). Now, the dataset consists of 24 samples and class distributions of it are as 4 samples for hard-contact lenses, 5 samples for soft-contact lenses, and 15 samples for non-contact lenses (Gorman, Sejnowski, 1998). Table 4 and Fig. 6b show the results of training algorithms to solve this dataset.

According to the Table 4, MLPABGSA has the best performance based upon three statistical characteristics (STD, AVE and p-value). Therefore, it also shows a good ability to solve these kinds of datasets (dataset with few samples) in favour of not reaching local minimum more than the other algorithms.

4.4. Sonar dataset

As it can be seen in Fig. 7, the new dataset used in the study has been obtained from a special sonobuoy (Naseri, 2015). In this study six objects, including four targets and two non-targets, are laid on the sandy

Table 4. Experimental results for Lenses dataset.

Algorithm	MSE (AVE ± STD)	p-values	Classification rate [%]
MLPABGSA	8.09e–157 ± 1.51e–42	N/A	100
MLPPSO	0.0949 ± 0.0787	0.0159	94.1547
MLPGSA	0.0068 ± 0.0310	8.6974e–14	99.4112
MLPGD	0.1658e–27 ± 0.0745	0.0079	97.5333
MLPGA	0.0677 ± 0.0489	0.0453	98.4328

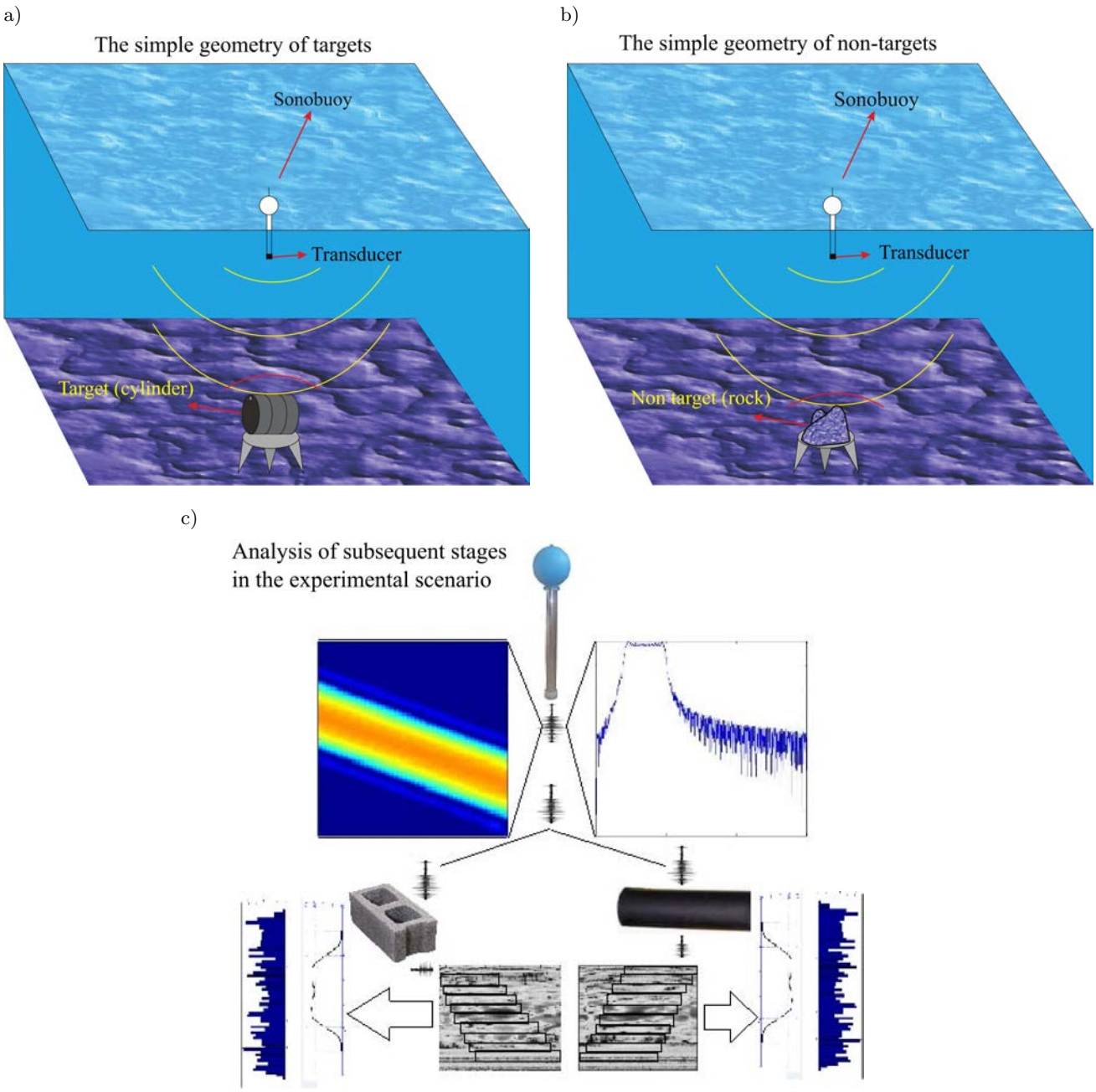

Fig. 7. The simple geometry of the experiment: a) target position, b) non-target position, and c) a simple shape of the sample transmitted and received signal and in subsequent stages of its analysis.

sea bottom. A simple geometry of the sonobuoy, targets, and non-targets is shown in Fig. 7a and Fig. 7b. In this experiment, the transmitted signal is Wide-Band Linear Frequency Modulated Pulse (WLFM) that covers frequency from five to a hundred and ten. Targets laid on the sea bottom rotate as much as 180 degrees as they have one-degree interval accuracy by an electromotor. The backscattered echoes are accumulated from 10 meters away from the targets.

A set of 200 echoes out of 1000 echoes is chosen on the basis of specular return strength (8.0 to 15.0 dB signal-to-noise ratio). Each temporal return having a target-like echo (with 6.0 to 8.0 signal-to-noise ratio) considered as clutter. Of the 200 chosen samples, a number of 120 samples belong to targets (30 samples for each target) and the rest belong to non-targets (40 samples for each non-target). A mean of 10 echoes is chosen from each aspect angle.

Regarding massive raw data obtained from previous stage, the above mentioned massive calculation will be expected. To ease calculation complexity, relating to classifying and extracting feature, it is essential to detect targets out of total received data. To implement this, the intensity of the received signal is used. It is in-

evitable to consider multi-path propagation, secondary reflections, and reverberation due to shoal of the region. The researcher attempts to eliminate trajectory after detecting stage and before extracting feature by applying a matched filter.

Afterwards, the researcher proceeds to regain main backscattered signal by means of inverse filter. The reason behind this stage is that separating trajectories in matched filter domain is much easier than time domain. The event of pre-processing takes place in four stages as follow:

1) **Scaling**: It converts raw signal into rapid signal due to eliminating filter and boosting gain- effect at accumulating stage.
2) **Down Sampling**: Main sampling rate is 2 MHz which highly surpasses main signal band width. To reduce sampling rate in such a way that useful data does not miss, reference (Preston, 2004) has been applied. Based upon this reference, by means of environmental data such as water depth, functional frequency, area under exploitation etc., few fixed points are selected at sampling stage. Here 2048 points are chosen not to waste useful data from which extract feature.
3) **Multi-path and Artificial Elimination Process**: In this method, by means of randomly cross-correlating the backscattered signal the position of maximum matched filter output, named x, is determined at each angle. Afterwards, a window covering $[x-\text{left}: x+\text{right}]$ exerts over signal. This area includes right = 300 and left = 211 that ultimately form a window of 512 points. To maintain quantity of the main signal, this signal is segmented and zero padded and to eliminate the effect of transmitted signal, inversed filtering is done by Eq. (21):

$$H(k) = \frac{X(k)}{|X(k)|^2 + c}, \tag{21}$$

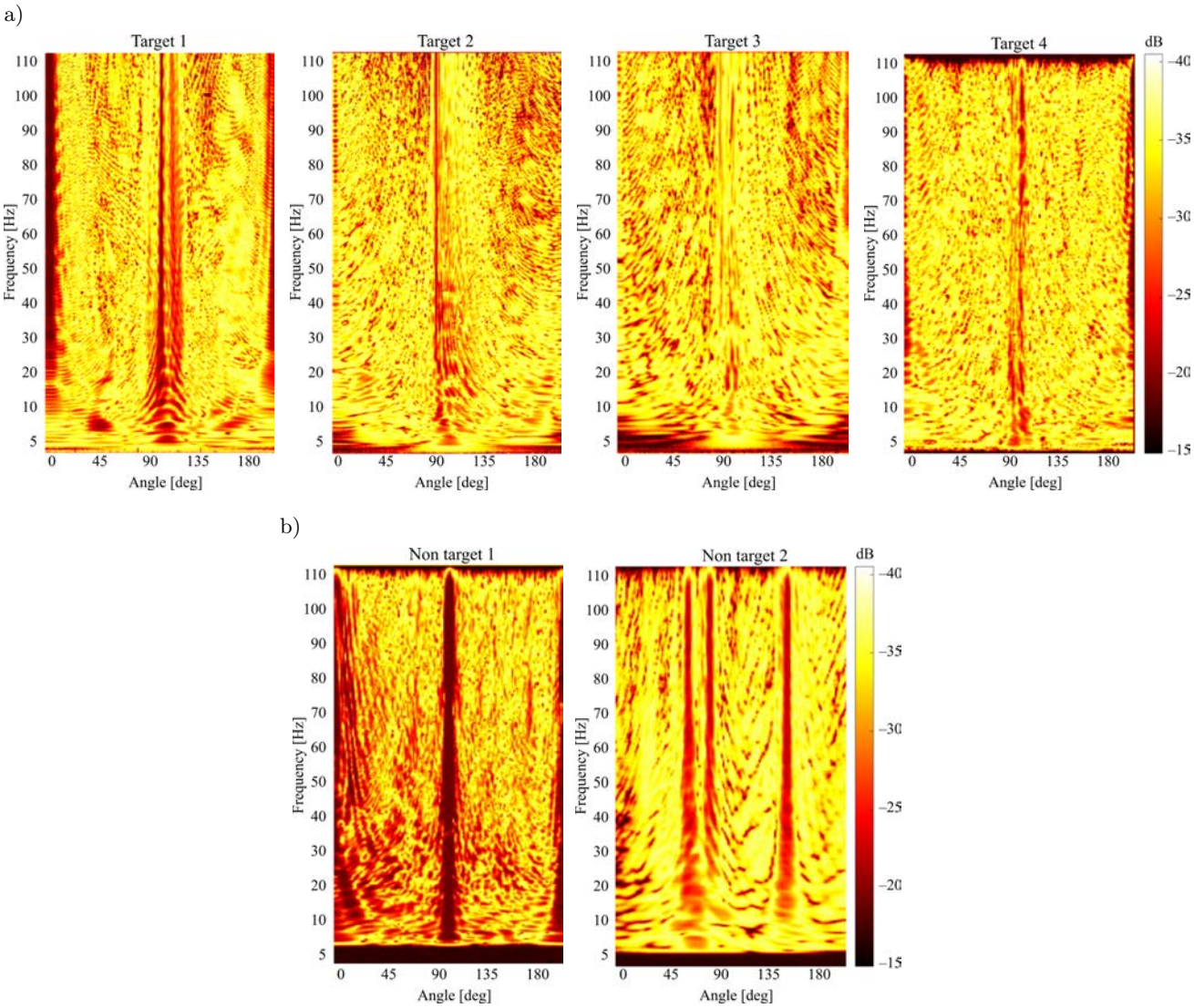

Fig. 8. Samples of backscattered signal: a) targets 1–4, and b) non-targets 1, 2.

wherein $X(k)$ is Fourier transform of the transmitted signal and $c = 0.0025 \cdot \max(|X(k)|^2)$ is added to equation to eliminate singularity problem. The output of recent signal is pure subtraction without trajectory effect.

4) **Normalization**: Any target scales in such a way that each takes the same target strength finally. To do so, each backscattered signal is dividend through Signal Reference Amplitude (SRA) which is the largest amplitude and less than 90% of the maximum amplitude of whole aspects for interesting target. Samples of backscattered signal from various targets and non-targets are indicated in Fig. 8. They are frequency's function and target bearing.

4.4.1. Feature extraction

After pre-processing and receiving detected frames which comprise of the sound of backscattered signal, detected sounds are delivered to the feature extraction stage, and trajectory effect of these sounds is eliminated and transformed into frequency domain under the name $S(k)$. At this stage, energy spectrum is determined by Eq. (22):

$$|S(k)|^2 = S_r^2(k) + S_i^2(k), \tag{22}$$

where $S_r(k)$ and $S_i(k)$ are real and imagery Fourier transform of the detected signal. Then energy spectrum $|S(k)|^2$ is filtered by Mel-scaled triangular filter. The output energy of l-th filter is determined by Eq. (23):

$$E(l) = \sum_{k=0}^{N-1} |S(k)|^2 H_l(k), \tag{23}$$

where N indicates the number of discrete frequencies used in pre-processing at FFT conversion and $H_l(k)$ is the filter transfer function in such a way that $l = 0, 1, ..., M$.

The dynamic range of the filtered energy spectrum at Mel-filtered energy spectrum is compacted by logarithm function through Eq. (24):

$$e(l) = \log(E(l)). \tag{24}$$

Eventually, Mel-Frequency Cepstral Coefficients (MFCCs) are calculated by moving back to time domain and using Discrete Cosine Transform (DCT) through Eq. (25):

$$c(n) = \sum_{l=1}^{M} e(l) \cos\left(n(l-0.5)\frac{\pi}{M}\right). \tag{25}$$

Upon this, for each detected target, feature vector takes the form as Eq. (26):

$$X_m = [c(0) \quad c(1) \quad ... \quad c(P-1)]^{\mathrm{T}}. \tag{26}$$

All above stages within pre-processing and feature extracting are shown in Fig. 9 schematically. Finally, after software verification, FPGA implementation of all stages was done. Xilinx Virtex 7 was used for this goal. An experimental sample setup for all procedures is indicated in Fig. 10.

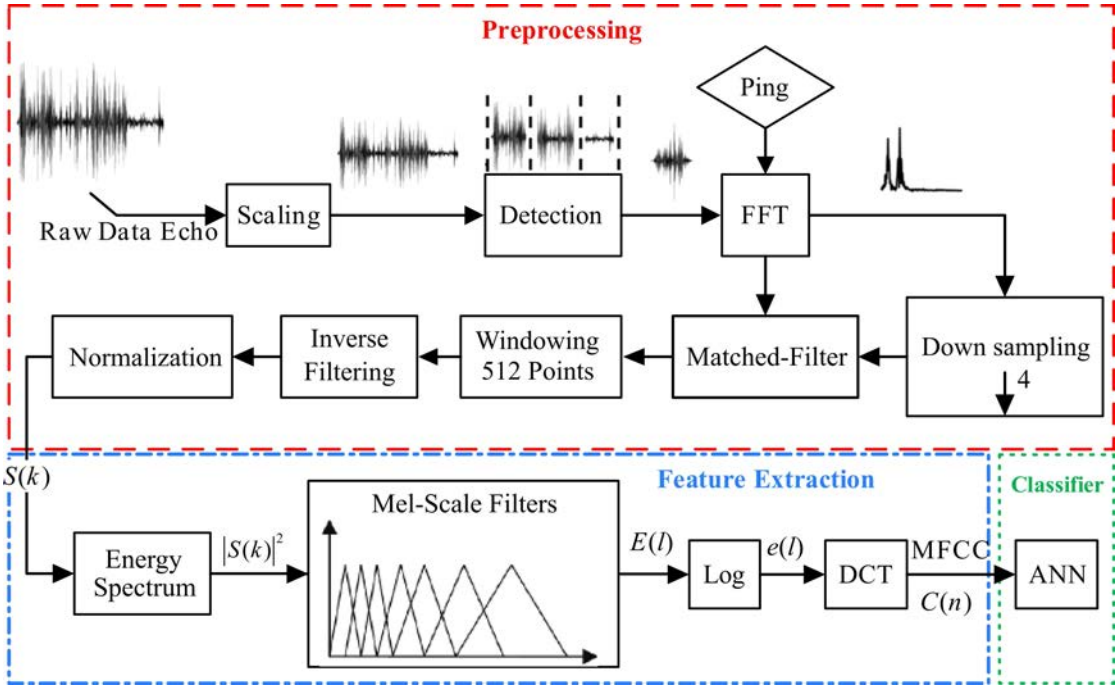

Fig. 9. Block diagram for all stages.

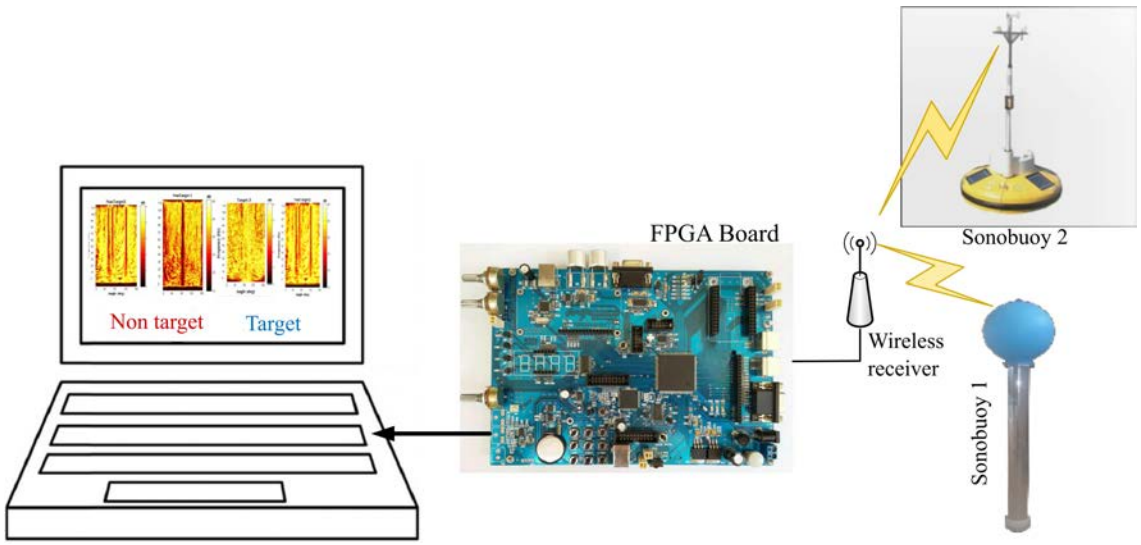

Fig. 10. Experimental test setup.

4.4.2. Sonar targets classifications

After pre-processing sonar backed-echoes and obtaining normalized dataset between 0 and 1, having been trained by various algorithms, dataset of $200 \cdot 23$ (200 samples have 23 features) are exerted in MLP NN. Outputs are illustrated in Fig. 11 and Table 5.

As it can be seen in Table 5, ABGSA algorithm with 95.2015 takes the best result and GD algorithm with 81.9333 takes the weakest operation. Regarding periodical identity, extra local maximum and minimum possibility of falling in local maximum is too high for algorithm such as GD. The weak result of GD algorithm confirms this statement; whereas algorithms such as ABGSA, GSA, GA and PSO with random identities and no use of differentiations have better functions than any other algorithms. From another side, it can be seen that in experiment, GSA

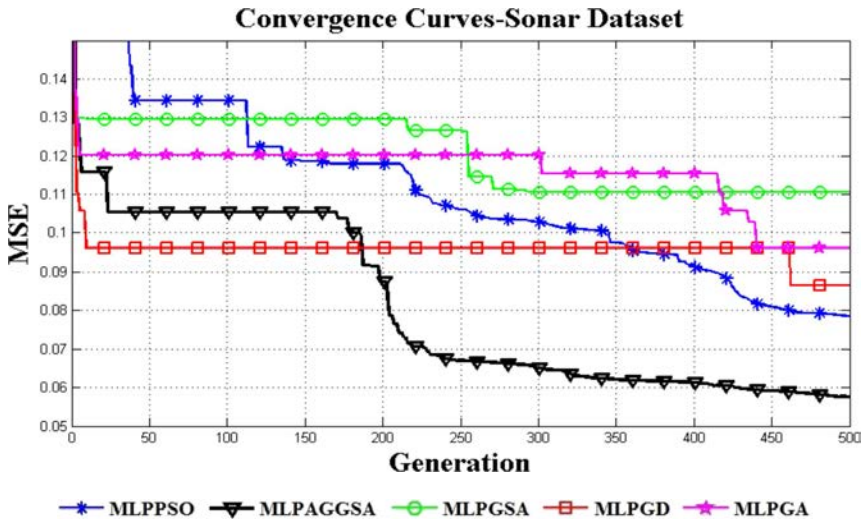

Fig. 11. Classification rate and convergence speed of the various algorithms exerted on sonar dataset.

Table 5. Experimental results for sonar dataset.

Algorithm	MSE (AVE ± STD)	p-values	Classification rate [%]
MLPABGSA	0.083e–2 ± 0.0082	N/A	95.2015
MLPPSO	0.1311 ± 0.1076	7.2239e–04	92.9512
MLPGSA	0.1049 ± 0.0965	9.2798e–20	94.0969
MLPGD	0.2427 ± 0.1064	0.0039	81.9333
MLPGA	0.0794 ± 0.0112	0.0079	91.8067

and ABGSA algorithms due to high capability in detecting targets reach for better results in this kind of dataset. As mentioned before sonar dataset due to covering whole searching space for classifying, requires algorithm strong in detection phase. Algorithms of GSA family are better than the other meta-heuristic ones. Regarding Fig. 11, it can be seen that ABGSA algorithm used the best mass and developed function in exploitation phase has better results than the other algorithms even GSA.

5. Conclusions

In this study, a new proposed meta-heuristic algorithm entitled ABGSA for training MLP NN was applied. For measuring designed classifier, dataset from Iris, Lenses, and sonar have been used and obtained results were compared with features of GD, PSO, GA, and GSA algorithms. As mentioned before GSA algorithm has high capability in exploration phase that is why this algorithm is used for sonar datasets which needs high dimensions requirement. Developed algorithm known as ABGSA for training MLP NN was applied due to defection of the GSA in exploration phase. Results indicated that ABGSA algorithm presents, in comparison with featured algorithms, much better outputs in terms of convergence speed, capability of avoiding from falling into the local minimums, and classification accuracy. Obtained results also indicated that classification rate from dataset of Iris, Lenses, and sonar due to the usage of designed classifiers with ABGSA (MLPABGSA) (in comparison with classic GSA algorithm) have been increased to 3.8, 0.6, and 1.2, respectively.

References

1. Allahyar M.R., Nemati M.H., Naseri A.S., Golshani A.A. (2012), *Monitoring and modelling studies of Iranian coasts phase 5: northern coasts*, Ports and Maritime Organization, http://irancoasts.pmo.ir/en/first.
2. Aubry A., De Maio A., Piezzo M., Farina A., Wicks M. (2012), *Cognitive design of the receive filter and transmitted phase code in reverberating environment*, IET Radar, Sonar & Navigation, **6**, 9, 822–833, doi: 10.1049/iet-rsn.2012.0029.
3. Auer P., Burgsteiner H., Maass W. (2008), *A learning rule for very simple universal approximators consisting of a single layer of perceptrons*, Neural Networks, **21**, 5, 786–795.
4. Blumrosen G., Fishman B., Yovel Y. (2014), *Non-contact wideband sonar for human activity detection and classification*, IEEE Sensors Journal, **14**, 11, 4043–4054, doi: 10.1109/JSEN.2014.2328340.
5. Chen H., Li S., Tang Z. (2011), *Hybrid gravitational search algorithm with random-key encoding scheme combined with simulated annealing*, International Journal of Computer Science and Network Security, **11**, 6, 208–217.
6. Chen J., Qin Z., Liu Y., Lu J. (2005), *Particle swarm optimization with local search*, IEEE Conference on Neural Networks and Brain, pp. 481–484, Beijing.
7. Chen S., Mei T., Luo M., Yang X. (2007), *Identification of nonlinear system based on a new hybrid gradient-based PSO algorithm*, International Conference on Information Acquisition, pp. 265–268, Seogwipo-si.
8. Chu D., Stanto T.K. (2010), *Statistics of echoes from a directional sonar beam insonifying finite numbers of single scatterers and patches of scatterers*, IEEE Journal on Oceanic Engineering, **35**, 2, 267–277.
9. Cui X., Geol V., Kingsbury B. (2015), *Data augmentation for deep neural network acoustic modelling*, IEEE/ACM Transaction on Audio, Speech, and Language Processing, **23**, 9, 1496–1477, doi: 10.1109/TASLP.2015.2438544.
10. Das A., Kumar A., Bahl R. (2013), *Marine vessel classification based on passive sonar data: the spectrum-based approach*, IET Radar, Sonar & Navigation, **7**, 1, 87–93.
11. Derrac J., García S., Molina D., Herrera F. (2011), *A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms*, Swarm and Evolutionary Computation, **1**, 1, 3–18.
12. Doraghinejad M., Nezamabadi-Pour H., Mahani A. (2014), *Channel assignment in multi-radio wireless mesh networks using an improved gravitational search algorithm*, Journal of Network and Computer Applications, **38**, 163–171.
13. Fei T., Kraus D., Zoubir A.M. (2015), *Contributions to automatic target recognition systems for underwater mine classification*, IEEE Transaction on Geosciences and Remote Sensing, **53**, 1, 505–518.
14. Gonzalez-Álvarez D.L., Vega-Rodríguez M.A., Gómez-Pulido J.A., Sánchez-Pérez J.M. (2011), *Applying a multiobjective gravitational search algorithm (MO-GSA) to discover motifs*, [in:] *Advances in Computational Intelligence*, pp. 372–379, Springer, Berlin Heidelberg.
15. Gorman R.P., Sejnowski T.J. (1998), *Datasets*, from http://archive.ics.uci.edu/ml/datasets.
16. Gu B., Pan F. (2013), *Modified gravitational search algorithm with particle memory ability and its application*, International Journal of Innovative Computing, Information and Control, **9**, 4531–4544.
17. Guo Z. (2012), *A hybrid optimization algorithm based on artificial bee colony and gravitational search algorithm*, International Journal of Digital Content Technology and Its Applications, **6**, 620–626.
18. Han K., Wang D. (2014), *Neural network based pitch tracking in very noisy speech*, IEEE/ACM Transaction on Audio, Speech, and Language Processing, **22**, 2158–2168.

19. Han X.H., Chang X.M. (2012), *A chaotic digital secure communication based on a modified gravitational search algorithm filter*, Information Sciences, **208**, 14–27.

20. Han XH., Quan L., Xiong X.Y., Wu B. (2013), *Facing the classification of binary problems with a hybrid system based on quantum-inspired binary gravitational search algorithm and K-NN method*, Engineering Applications of Artificial Intelligence, **26**, 2424–2430.

21. Hatamlou A., Abdullah S., Nezamabadi-Pour H. (2012), *A combined approach for clustering based on K-means and gravitational search algorithms*, Swarm and Evolutionary Computation, **6**, 47–52.

22. Hatamlou A., Abdullah S., Othman Z. (2011), *Gravitational search algorithm with heuristic search for clustering problems*, 3rd Conference on Data Mining and Optimization (DMO), pp. 190–193, Putrajaya.

23. Karayiannis N.B. (1999), *Reformulated radial basis neural networks trained by gradient descent*, IEEE Transaction on Neural Networks, **10**, 657–671.

24. Kumar N., Mitra U., Narayanan S.S. (2015), *Robust object classification in underwater side scan sonar images by using reliability-aware fusion of shadow features*, IEEE Journal on Oceanic Engineering, **40**, 592–606.

25. Li C., Li H., Kou P. (2014), *Piecewise function based gravitational search algorithm and its application on parameter identification of AVR system*, Neurocomputing, **124**, 139–148.

26. Li C., Zhou J. (2011), *Parameters identi?cation of hydraulic turbine governing system using improved gravitational search algorithm*, Energy Conversion and Management, **52**, 374381.

27. Li C., Zhou J., Xiao J., Xiao H. (2012), *Parameters identification of chaotic system by chaotic gravitational search algorithm*, Chaos, Solitons & Fractals, **45**, 539–547.

28. Li P., Duan H.B. (2012), *Path planning of unmanned aerial vehicle based on improved gravitational search algorithm*, Science China Technological Sciences, **55**, 2712–2719.

29. Liu C., Wang H., Yao P. (2014), *On terrain-aided navigation for unmanned aerial vehicle using b-spline neural network and extended Kalman filter*, IEEE Conference on Guidance, Navigation and Control, pp. 2258– 2263, Chinese.

30. Liu Y., Ma L. (2013), *Improved gravitational search algorithm based on free search differential evolution*, Journal of Systems Engineering and Electronics, **24**, 690–698.

31. Meuleau N., Dorigo M. (2002), *Ant colony optimization and stochastic gradient descent*, Artificial Life, **8**, 103–121.

32. Mirjalili S.A. (2015), *How effective is the grey wolf optimizer in training multi-layer perceptrons*, Applied Intelligence, **43**, 150–161.

33. Mirjalili S.A., Hashim S.Z.M. (2010), *A new hybrid PSOGSA algorithm for function optimization*, IEEE Conference on Computer and Information Application, pp. 374–377, Tianjin.

34. Mirjalili S.A., Lewis A. (2014), *Adaptive gbest-guided gravitational search algorithm*, Neural Computing and Applications, **25**, pp. 1569–1584.

35. Mirjalili S.A., Mirjalili S.M., Lewis A. (2014), *Let a biogeography-based optimizer train your multi-layer perceptron*, Journal of Information Sciences, **269**, 188–209.

36. Mirjalili S.A., Mohd Hashim S.Z., Moradian Sardroudi H. (2012), *Training feed forward neural networks using hybrid particle swarm optimization and gravitational search algorithm*, Applied Mathematics and Computation, **218**, 11125–11137.

37. Moody J., Darken C.J. (1989), *Fast learning in networks of locally-tuned processing units*, Neural Computer, **1**, 281–294.

38. Mosavi M.R., Khishe M., Aghababaee M., Mohammadzadeh F. (2015), *Approximation of active sonar clutter's statistical parameters using array's effective beam-width*, Iranian Journal of Marine Science and Technology [in Persian], **73**, 11–22.

39. Mosavi M.R., Khishe M., Ebrahimi E. (2015), *Classification of sonar targets using OMKC genetic algorithms and statistical moments*, Journal of Advance in Computer Research, **7**, 50–59.

40. Naseri M.J. (2015), *Floating buoy controller design and implementation by using special sonobuoys*, M.Sc. Thesis, University of Nowshahr Marine Sciences, Iran.

41. Noman N., Iba H. (2008), *Accelerating differential evolution using an adaptive local search*, IEEE Transaction on Evolutionary Computing, **12**,107–125.

42. Pailhas Y., Capus C., Brown K. (2012), *Dolphin-inspired sonar system and its performance*, IET Radar, Sonar & Navigation, **6**, 753–763.

43. Pan X., Li C., Xu Y., Xu W., Gong X. (2014), *Combination of time-reversal focusing and nulling for detection of small targets in strong reverberation environments*, IET Radar, Sonar & Navigation, **8**, 9–16.

44. Parsazad S., Sadoghi Yazdei H., Effati S. (2013), *Gravitation based classification*, Information Sciences, **220**, 319–330.

45. Pearce S.K., Bird J.S. (2013), *Sharpening side scan sonar images for shallow-water target and habitat classification with a vertically stacked array*, IEEE Journal on Oceanic Engineering, **38**, 455–469.

46. Pei J., Liu X., Pardalos P.M., Fan W., Yang S., Wang L. (2014), *Application of an effective modified gravitational search algorithm for the coordinated scheduling problem in a two-stage supply chain*, International Journal of Advanced Manufacturing Technology, **70**, 335–348.

47. Preston J.M. (2004), *Resampling sonar echo time series primarily for seabed sediment*, United State Patent, US.

48. Rahmani Hosseinabadi A.A., Ramzannezhad Ghaleh M.R., Hashemi S.E. (2013), *Application of modified gravitational search algorithm to solve the problem of teaching hidden Markov model*, International Journal of Computer Science, **10**, 3, 1–8.

49. Rashedi E., Nezamabadi-Pour H., Saryazdi S. (2009), *GSA: A gravitational search algorithm*, Information Sciences, **179**, 2232–2248.

50. Sabri N.M., Puteh M., Mahmood M.R. (2013), *A review of gravitational search algorithm*, International Journal of Advance in Soft Computing and Application, **5**, 1–39.

51. Sarafrazi S., Nezamabadi-Pour H., Saryazdi S. (2011), *Disruption: A new operator in gravitational search algorithm*, Scientia Iranica, **18**, 539–548.

52. Shaw B., Mukherjee V., Ghoshal S.P. (2012), *A novel opposition based gravitational search algorithm for combined economic and emission dispatch problems of power systems*, International Journal of Electrical Power & Energy Systems, **35**, 21–33.

53. Sinaie S. (2010), *Solving shortest path problem using gravitational search algorithm and neural networks*, M.Sc. Thesis, Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia (UTM).

54. Souza Filho J.B.O., De Seixas J.M. (2016), *Class-modular multi-layer perceptron networks for supporting passive sonar signal classification*, IET Radar, Sonar & Navigation, **10**, 311–317.

55. Sun G., Zhang A. (2013), *A hybrid genetic algorithm and gravitational using multilevel thresholding*, [in:] *Pattern Recognition and Image Analysis*, Sanches J.M., Mico L., Cardoso J.S. [Ed.], pp. 707–714, Springer Berlin Heidelberg, China University.

56. Wilcoxon F. (1945), *Individual comparisons by ranking methods*, Biometrics Bulletin Society, **1**, pp. 80–83.

57. Williams D.P., Fakiris E. (2014), *Exploring environmental information for improved underwater target classification in sonar imagery*, IEEE Transaction on Geosciences and Remote Sensing, **52**, 6284–6297.

58. Yegireddi S. (2015), *A combined approach of generic algorithm and neural networks with an application to geoacoustic inversion studies*, Indian Journal of Geo-Marine Sciences, **44**, 195–201.

59. Zhang Y., Li Y., Xia F., Luo Z. (2012), *Immunity-based gravitational search algorithm*, 3rd International Conference on Information Computing and Applications, pp. 754–761, Chengde, China.

60. Zhang Y., Wu L., Zhang Y., Wang J. (2012), *Immune gravitation inspired optimization algorithm*, [in:] *Advanced Intelligent Computing*, Huang D.S., Gan Y., Bevilacqua V., Figueroa J.C. [Ed.], pp. 178–185, Springer Berlin Heidelberg, Haikou, China.

61. Zhou J.X., Zhang X.Z. (2005), *Shallow-water reverberation level: measurement technique and initial reference values*, IEEE Journal on Oceanic Engineering, **30**, 832–842.