Applied Soft Computing 50 (2017) 94–108

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

An evolutionary gray gradient algorithm for multilevel thresholding of brain MR images using soft computing techniques

Rutuparna Panda [a,*], Sanjay Agrawal [a], Leena Samantaray [b], Ajith Abraham [c]

[a] Department of Electronics & Telecommunication Engineering VSS University of Technology, Burla, Odisha, 768018, India
[b] Depatment of Electronics & Communication Engineering, ABIT, Cuttack, Odisha, India
[c] Machine Intelligence Research Labs (MIR Labs) Scientific Network for Innovation and Research Excellence, WA 98071-2259, USA

ARTICLE INFO

Article history:
Received 25 July 2016
Received in revised form 21 October 2016
Accepted 2 November 2016
Available online 11 November 2016

Keywords:
Multilevel thresholding
MRI brain
Soft computing techniques
Otsu's method

ABSTRACT

The conventional two dimensional (2-D) histogram based Otsu's method gives unreliable results while considering multilevel thresholding of brain magnetic resonance (MR) images, because the edges of the brain regions are not preserved due to the local averaging process involved. Moreover, some of the useful pixels present inside the off-diagonal regions are ignored in the calculation. This article presents an evolutionary gray gradient algorithm (EGGA) for optimal multilevel thresholding of brain MR images. In this paper, more edge information is preserved by computing 2-D histogram based gray gradient. The key to our success is the use of the gray gradient information between the pixel values and the pixel average values to minimize the information loss. In addition, the speed improvement is achieved. Theoretical formulations are derived for computing the maximum between class variance from the 2-D histogram of the brain image. A first-hand fitness function is suggested for the EGGA. A novel adaptive swallow swarm optimization (ASSO) algorithm is introduced to optimize the fitness function. The performance of ASSO is validated using twenty three standard Benchmark test functions. The performance of ASSO is better than swallow swarm optimization (SSO). The optimum threshold value is obtained by maximizing the between class variance using ASSO. Our method is tested using the standard axial T_2 – weighted brain MRI database of Harvard medical education using 100 slices. Performance of our method is compared to the Otsu's method based on the one dimensional (1-D) and the 2-D histogram. The results are also compared among four different soft computing techniques. It is observed that results obtained using our method is better than the other methods, both qualitatively and quantitatively. Benefits of our method are – (i) the EGGA exhibits better objective function values; (ii) the EGGA provides us significantly improved results; and (iii) more computational speed is achieved.

1. Introduction

In recent years, application of image segmentation techniques in medical applications has attracted significant interest in research and clinical diagnosis. In fact, medical image segmentation has emerged as a challenging and promising area of research [1]. At present, magnetic resonance imaging (MRI) is one of the most popular clinical diagnostic procedures. It provides more detail data related to the human soft tissue framework. It gives high contrast between the various soft tissues of the body. This makes it more suitable as compared to other medical imaging techniques such as CT or X-rays. The segmentation of MR brain images into various regions, particularly gray matter (GM), white matter (WM) and cerebrospinal fluid (CSF), is one of the important problems. In connection with the problem of MR brain image segmentation, the authors emphasize on segmenting brain images without perceptible diseases into three regions.

Thresholding is one of the simplest techniques of achieving image segmentation. Effective thresholding leads to success of diagnosing disorder or lesions in the brain. There are several techniques in the literature reported for effective thresholding [2–4]. The problem of thresholding is classified based on number of threshold values. Single level thresholding partitions an image into two classes by selecting one proper threshold value. Multilevel thresholding partitions an image into multiple classes by selecting multiple threshold values. It is more suitable for images with complex boundaries and multimodal histogram. For this valid reason, multilevel thresholding is still considered as an important area of

* Corresponding author.
E-mail address: r_ppanda@yahoo.co.in (R. Panda).

http://dx.doi.org/10.1016/j.asoc.2016.11.011
1568-4946/

research. A wide variety of algorithms is proposed for thresholding based on the 1-D image histogram (first order statistics) [5–13]. But, one can find that spatial correlation among the neighboring pixels in an image is not found in the 1-D image histogram. Consequently, the thresholding performance using the above methods may degrade.

In order to address this problem, the authors propose an algorithm based on gray gradient information derived from the 2-D histogram (second order statistics) of the brain MR image for multilevel thresholding. The problem of multilevel thresholding becomes computationally expensive with increase in number of threshold level. That is why; researchers have suggested optimal multilevel thresholding. The optimal threshold values are found by optimizing (maximizing or minimizing) a fitness function. Researchers have suggested a number of criteria in the literature for determining the fitness function [14–23]. Moreover, some of the above-mentioned techniques are used for multilevel thresholding of brain MR images [24,25].

In 2014, Manikandan et al. [10] presented a multilevel thresholding scheme for segmenting medical (human) brain images using real coded genetic algorithm. The authors used the simulated binary crossover method for application. The optimal threshold value was obtained by maximizing the Kapur's entropy. The results were compared with algorithms like Nelder–Mead simplex, particle swarm optimization (PSO), bacteria foraging (BF) and adaptive bacteria foraging (ABF). In 2014, Hadjidimitriou et al. [11] presented morphological processing based T1-weighted brain MRI segmentation. They used a fast histogram technique. According to the authors, there is a strong need for a fast and accurate method to distinguish a healthy person from a patient.

Wang et al. (2015) [12] proposed a new technique for automatic segmentation of brain MRI to separate out cerebral vessels. They used Otsu's algorithm to split the brain images into 2-regions. Mesejo et al. (2015) [13] presented a medical image segmentation scheme using the geometric deformable models and metaheuristics. Their paper described a hybrid level set approach for medical image segmentation with an increased accuracy. Their scheme is a combination of edge and region information with the prior knowledge of the shape. They used Genetic Algorithm and Scatter Search to derive the shape. Sathya et al. (2011) [24] presented a multilevel thresholding approach based on adaptive bacteria foraging for brain MR image segmentation. The authors used an adaptive step size to improve the conventional BF algorithm performance. This might have helped in reducing the execution time only. They used Kapur's entropy and between-class variance (1-D Otsu based) as the criteria for determining the fitness function. The authors compared their results with BF, PSO and genetic algorithm (GA). Maitra et al. (2008) [25] suggested an optimal multilevel thresholding algorithm using BF algorithm for brain MR image segmentation. They used 1-D histogram based Otsu method for thresholding. The authors compared their results with PSO and concluded that BF outperforms PSO.

From the literature, it is found that the most of the earlier methods are based on the first order statistics (1-D histogram). Hence, spatial correlation information between the neighboring pixels is not taken into account. To address this problem, Otsu's 1-D threshold method is extended to the 2-D versions forming a 2-D histogram of the image. The 2-D histogram is constructed by considering gray level and local average gray level of the neighboring pixels. However, there is a loss of edge information due to the local averaging. In addition, 2-D Otsu's method proposed in the literature is computationally expensive.

A single threshold value partitions the 2-D histogram of an image into four quadrants. The diagonal quadrants correspond to local transitions within the background and the foreground only. The off-diagonal quadrants correspond to joint transitions across the boundaries between the background and the foreground. While considering the 2-D Otsu method, the variance of the diagonal areas of the 2-D histogram is computed to obtain the threshold. This concept is extended here for multilevel thresholding of brain MR image. However, the local averaging involved in the process of forming the histogram incurs loss of information. More importantly, the 2-D Otsu method cannot preserve accurate edges of an image having complex boundaries. Therefore, it may not be appropriate for brain MR image segmentation. This has motivated us to propose an interesting method called EGGA, which uses the gray gradient information from the 2-D histogram. Although various fitness (objective) functions are suggested based on the Otsu's method for thresholding, the authors have put an effort to introduce a new objective function based on gray gradient information. The gray gradient feature is extracted from the 2-D histogram as a feature of the image. The idea is explained in Section 3. For both the single level and multilevel thresholding, the row wise areas of the 2-D histogram, instead of the diagonal areas, are used for computing the variance. As a result, the loss of information is minimized here. This method is quite different from the earlier methods suggested for multilevel thresholding applications. This may draw clear attention of the researchers for a further insight into our proposal.

Fig. 1 shows the block diagram of our proposed method. A brain MR image is read and the proposed technique is used after selecting the number of thresholds. At the end, the thresholded image is obtained.

The motivation behind our proposal is to preserve more accurate edge information. In fact, our approach is based on the second order statistics. A useful fitness function is derived and proposed for the application. The control parameters in SSO [26] are made adaptive in accordance with the fitness values. As a result, the convergence rate becomes faster than SSO. The contribution of the paper is twofold. Firstly, development of new theoretical foundations for 2-D histogram based multilevel thresholding of brain MR images. Sec-

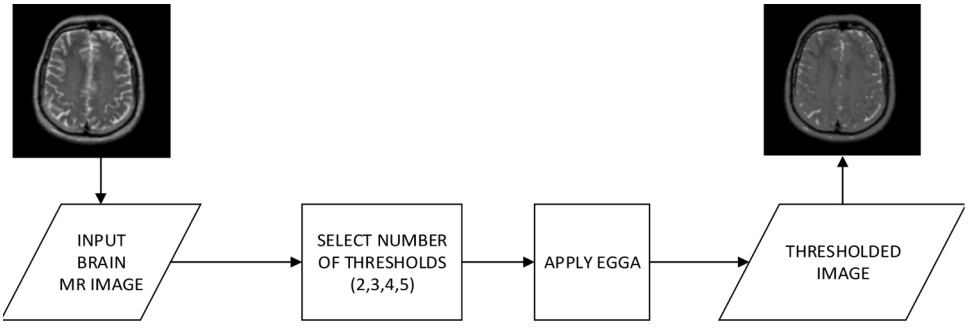


Fig. 1. Block diagram of the proposed method.

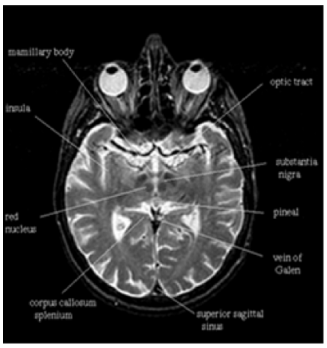


Fig. 2. T_2-weighted structural MR slice 24 of a normal brain.

ondly, a new adaptive SSO is introduced for optimization of the fitness function.

Results are compared with soft computing techniques namely SSO, PSO and coral reef optimization (CRO). SSO has faster convergence speed of getting the optimal result at lower number of iterations. The authors have compared SSO with PSO. Note that CRO being a new technique is considered for comparison. That's why our results are compared with PSO and CRO. We consider 100 slices of T_2 – weighted brain MR images from Harvard medical education to experiment. Some T_2 sequences reveal additional contrast between GM (lighter gray) and WM (darker gray). These images are found more suitable for brain study [27]. Fig. 2 shows a T_2-weighted MR slice of a normal brain.

The rest of the paper is organized as follows. Section 2 presents multilevel thresholding using Otsu method. Section 3 introduces our idea about the multilevel image thresholding using the gray gradient information derived from the 2-D histogram of an image. Section 4 presents the proposed optimization technique. Results and discussions are presented in Section 5. The conclusions are given in Section 6.

2. Otsu's method

Normally, the Otsu's method calculates the between-class and within-class variance using the 1-D image histogram. Then minimum within-class variance or the maximum between-class variance criterion is used to determine the threshold value. This concept is also extended to multilevel thresholding as well. It is noted that 1-D Otsu's method is considered to be the fastest and simplest of all methods. The threshold selection is mostly determined by the 1-D histogram of the image. However, the 1-D histogram does not provide us the spatial correlation between the pixels in the image. For this reason, images having complex boundaries may yield a poor thresholding performance. Accordingly, Otsu's 1-D method is extended to the 2-D version, where the 2-D histogram of an image is used [28]. The 2-D Otsu's method is explained in this section. The authors have implemented the 2-D Otsu's criteria extended to multilevel thresholding, for a comparison. The between class variance concept extended to 2-D is presented below.

Let I represent a gray scale image of size $M \times N$ with L gray levels $g = [1, 2, \ldots, L]$. Let $f(x, y)$ be the gray value of the pixel located at coordinate $(x, y)$where$x \in \{1, 2, \ldots, M\}$,$y \in \{1, 2, \ldots, N\}$. Let the number of pixels with gray value i be denoted asq_i. The total number of pixels is $M \times N$. Let $h(x, y)$ represent the local average gray value in a $w \times w$ neighborhood window, given by

$$h(x,y) = \left\lfloor \frac{1}{w \times w} \sum_{m=-k}^{k} \sum_{n=-k}^{k} f(x+m, y+n) \right\rfloor \tag{1}$$

where $k = \lfloor w/2 \rfloor$, $\lfloor . \rfloor$signifies the integer part of the number " $\cdot$ ". Note that $w < \min \ (M, N)$. Normally, w is taken an odd number. Let q_{ij} be the occurrence time of the pair (i, j), where $f(x, y) = i$ and $h(x, y) = j$. Then frequency of occurrence of pair (i, j) is given as:

$$p_{ij} = \frac{q_{ij}}{M \times N}, \qquad 1 \le i \le L, \qquad 1 \le j \le L \tag{2}$$

Now using (i, j) and p_{ij}, the 2-D histogram of the image I is constructed. The matrix representation of the 2-D histogram (of size $L \times L$) is displayed in Fig. 3.

As shown in Fig. 3(a), the coordinate pair (S, T) divides the 2-D histogram into four quadrants. The threshold point $(S, T)$is obtained by intersection of the two lines. The regions 1 and 4 are the diagonal quadrants and the regions 2 and 3 are the off-diagonal quadrants. Normally, the diagonal quadrants are used for threshold selection and the off-diagonal quadrants are ignored. Their exclusion may lead to inaccurate thresholding, because the quadrants 2 and 3 also contain some foreground and background information. In Fig. 3(b), the coordinate pairs (S_1, T_1) and (S_2, T_2) divides the 2-D histogram into nine regions. As per the convention, for 2-level thresholding, the diagonal regions '1, 5, 9′ are considered for threshold selection.

From Fig. 3(a), in the threshold point (S, T), T is the gray level threshold and S is the local average threshold. For single level thresholding, the image is divided into two classes c_f and c_b. Let ω_f be the probability distribution of the foreground pixels and ω_b

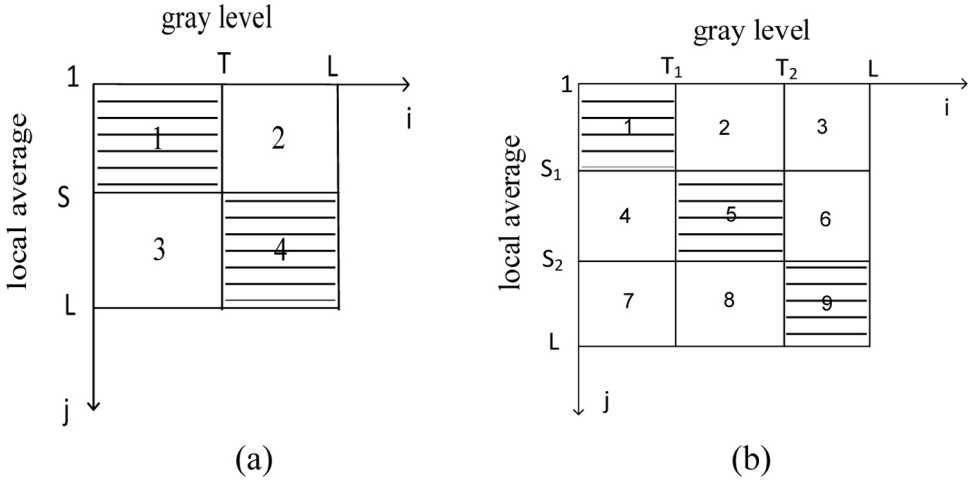


Fig. 3. Matrix representation of 2-D histogram based on local averaging. (a) 1-level thresholding (b) 2-level thresholding.

be the probability distribution of the background pixels, then one can have

$$\omega_f = P(c_f) = \sum_{i=1}^{S}\sum_{j=1}^{T} p_{ij} \tag{3}$$

$$\omega_b = P(c_b) = \sum_{i=S+1}^{L}\sum_{j=T+1}^{L} p_{ij} \tag{4}$$

If μ_f and $\mu_b$represent the mean vectors of the foreground and the background region, respectively; then they are defined as

$$\mu_f = \left(\mu_{f1}, \mu_{f2}\right)^T = \left[\sum_{i=1}^{S}\sum_{j=1}^{T} ip_{ij}/\omega_f, \sum_{i=1}^{S}\sum_{j=1}^{T} jp_{ij}/\omega_f\right]^T \tag{5}$$

$$\mu_b = (\mu_{b1}, \mu_{b2})^T = \left[\sum_{i=S+1}^{L}\sum_{j=T+1}^{L} ip_{ij}/\omega_b, \sum_{i=S+1}^{L}\sum_{j=T+1}^{L} jp_{ij}/\omega_b\right]^T \tag{6}$$

And μ be the overall mean vector, which is expressed as

$$\mu = (\mu_1, \mu_2)^T = \left[\sum_{i=1}^{L}\sum_{j=1}^{L} ip_{ij}, \sum_{i=1}^{L}\sum_{j=1}^{L} jp_{ij}\right]^T \tag{7}$$

The between class variance is computed as

$$\sigma_b{}^2 = \omega_f[(\mu_{f1} - \mu_1)^2 + (\mu_{f2} - \mu_2)^2] + \omega_b[(\mu_{b1} - \mu_1)^2 + (\mu_{b2} - \mu_2)^2] \tag{8}$$

Maximization of the between class variance value gives the image's optimal threshold as

$$(S_{opt}, T_{opt}) = arg\max_{S,T}\{\sigma_b{}^2(S,T)\} \tag{9}$$

where (S_{opt}, T_{opt}) represents optimum threshold point.

But for multilevel thresholding, no such formulations exist based on 2-D Otsu's method. So, the authors have put an effort to define the theoretical formulations for 2-D histogram based Otsu's method for multilevel thresholding. The new formulations are derived as follows.

As we know, for multilevel thresholding, $(K-1)$ thresholds $[S_1T_1, S_2T_2, \ldots, S_{k-1}T_{k-1}]$ divides the image into K classes: c_1 with threshold values$[1, \ldots, S_1T_1]$, c_2 with values $[S_1 + 1\ \ T_1 + 1, \ldots, S_2T_2]$ and similarly c_k with values$[S_{K-1} + 1\ \ T_{K-1} + 1, \ldots, LL]$. The objective function for maximizing the between class variance using 2-D Otsu's method for multilevel thresholding is

$$(S_{opt_1}T_{opt_1}, S_{opt_2}T_{opt_2}, \ldots, S_{opt_{K-1}}T_{opt_{K-1}}) = arg\max_{1<S_iT_i<L}\{\sigma_b{}^2(S_1T_1, S_2T_2, \ldots, S_{K-1}T_{K-1})\} \tag{10}$$

As defined above the probability distributions of the pixels are,

$$\begin{aligned}
\omega_1 &= P(c_1) = \sum_{i=1}^{S_1}\sum_{j=1}^{T_1} p_{ij},\\
\omega_2 &= P(c_2) = \sum_{i=S_1+1}^{S_2}\sum_{j=T_1+1}^{T_2} p_{ij},\\
&\vdots\\
\omega_K &= P(c_K) = \sum_{i=S_{K-1}+1}^{L}\sum_{j=T_{K-1}+1}^{L} p_{ij}
\end{aligned} \tag{11}$$

The mean vectors for diagonal regions for multilevel thresholding are $[\mu_1, \mu_2, \ldots, \mu_K]$ where

$$\begin{aligned}
\mu_1 &= (\mu_{11}, \mu_{12})^T = \left[\sum_{i=1}^{S_1}\sum_{j=1}^{T_1} ip_{ij}/\omega_1, \sum_{i=1}^{S_1}\sum_{j=1}^{T_1} jp_{ij}/\omega_1\right]^T,\\
\mu_2 &= (\mu_{21}, \mu_{22})^T = \left[\sum_{i=S_1+1}^{S_2}\sum_{j=T_1+1}^{T_2} ip_{ij}/\omega_2, \sum_{i=S_1+1}^{S_2}\sum_{j=T_1+1}^{T_2} jp_{ij}/\omega_2\right]^T\\
&\vdots\\
\mu_K &= (\mu_{K1}, \mu_{K2})^T = \left[\sum_{i=S_{K-1}+1}^{L}\sum_{j=T_{K-1}+1}^{L} ip_{ij}/\omega_K, \sum_{i=S_{K-1}+1}^{L}\sum_{j=T_{K-1}+1}^{L} jp_{ij}/\omega_K\right]^T
\end{aligned} \tag{12}$$

The overall mean μ_T is defined as

$$\mu_T = (\mu_{T1}, \mu_{T2})^T = \left[\sum_{i=1}^{L}\sum_{j=1}^{L} ip_{ij}, \sum_{i=1}^{L}\sum_{j=1}^{L} jp_{ij}\right]^T \tag{13}$$

The between class variance is defined as

$$\sigma_b{}^2 = \omega_1[(\mu_{11} - \mu_{T1})^2 + (\mu_{12} - \mu_{T2})^2] + \omega_2[(\mu_{21} - \mu_{T1})^2 + (\mu_{22} - \mu_{T2})^2] + \cdots + \omega_K[(\mu_{K1} - \mu_{T1})^2 + (\mu_{K2} - \mu_{T2})^2] \tag{14}$$

It is wise to reiterate here that the 2-D histogram consists of the pair of gray levels(i, j). The occurrence time of each pair is q_{ij}. The frequency of occurrence of each pair is denoted byp_{ij}. Here both i and j are independent variables whereas p_{ij} is the dependent variable. In case of the 2-D histogram shown in Fig. 3 (b), one can find little variation between the gray value and the local average gray value which is mainly distributed along the diagonal. However, for images like brain MR image, threshold selection by using the 2-D Otsu's method has the following limitations – (i) the regions along the diagonal is not smooth; (ii) the transition information in the off-diagonal region is not considered; and (iii) the shape of the image edge is not preserved. For that reason, the 2-D Otsu's method is not useful for brain MR image segmentation.

3. The proposed method

3.1. Gray Gradient

In this section, the authors suggest a new multilevel thresholding technique using the gray gradient information of an image. Now the 2-D histogram is formed using

$$f(x, y) = i$$

and

$$g(x, y) = abs(f(x, y) - h(x, y))$$

where $f(x, y)$ and $h(x, y)$ are explained in Eq. (1).

Let q_{ij} be the occurrence time of the pair(i, j), where $f(x, y) = i$ and$g(x, y) = j$. Then the frequency of occurrence of the pair (i, j) is given as:

$$p_{ij} = \frac{q_{ij}}{M \times N}, \qquad 1 \le i \le L, \qquad 1 \le j \le L \tag{15}$$

Now using (i, j) and p_{ij}, the 2-D histogram based on gray gradient information of the image I is formed. The matrix representation of the 2-D histogram (of size $L \times L$) is displayed in Fig. 4.

As shown in Fig. 4(a), the pair (S, T) partitions the 2-D histogram into four quadrants for one level thresholding. However, for 2-level thresholding, the pair (S, T_1) and (S, T_2) divides the 2-D histogram

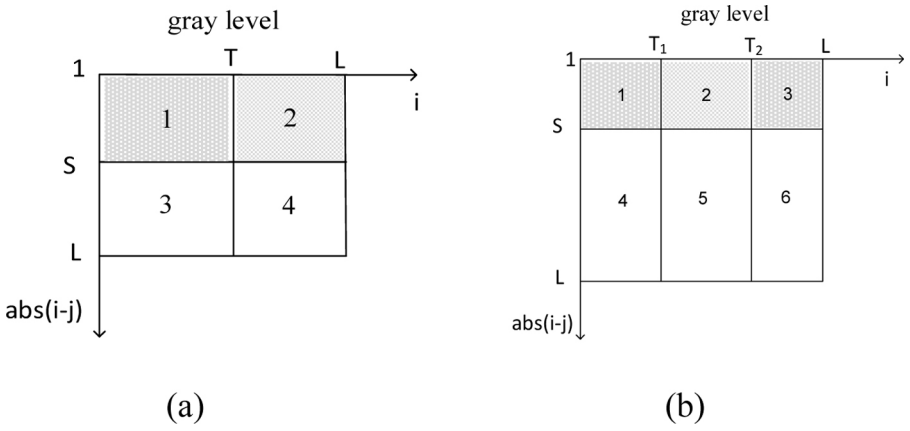


Fig. 4. Matrix representation of 2-D histogram based on Gray gradient. (a) 1-level thresholding (b) 2-level thresholding.

into six regions. In Fig. 4(a), the intersection of the two lines gives the overall threshold point(S, T). Here the quadrants 1 and 2 are called the first row wise quadrants. The quadrants 3 and 4 are coined as the second row wise quadrants. It is observed that the gray gradient information is contained in quadrants 1 and 2, so they are considered in the threshold calculation. The quadrants 3 and 4 are ignored in the calculation, because the area under quadrants 3 and 4 do not contain any edge information. This is an important issue for the threshold selection of brain MR images. In fact, the first row wise quadrants are used for threshold selection, i.e. the quadrant 1 pixels as the background and the quadrant 2 pixels as the foreground or vice versa.

Likewise, in Fig. 4(b), the intersection of the three lines give the overall threshold points $(S, T_1)$and $(S, T_2)$for two-level thresholding. Here, the regions 1, 2, 3 belong to the first row wise regions. These regions are only considered for the threshold calculation for two level thresholding. This idea is extended to multilevel thresholding, where the regions corresponding to local transitions are derived from a gray gradient based 2-D histogram. Depending on the number of levels of thresholding, the 2-D histogram is divided into many regions. Here, 2-level thresholding divides the 2-D histogram into 6 regions, 3-level thresholding divides it into 8 regions, 4-level thresholding divides it into 10 regions and 5-level thresholding divides the 2-D histogram into 12 regions. Hence, the authors propose to select the first row wise regions of the 2-D histogram corresponding to local transitions, the idea of which is influenced by the difference between the pixel value and the mean pixel value.

Two dimensional histogram plots for the test image (2018) from Berkeley segmentation dataset (BSD300) [29] are shown in Fig. 5. The original test image is displayed in Fig. 5 (a). Based on the discussions made in Section 2, the 2-D histogram for the Otsu method is displayed in Fig. 5 (b). As explained in Section 3, the 2-D histogram for the proposed EGGA method is displayed in Fig. 5 (c). Interestingly, in our case, Fig. 5 (c), the regions are distributed along the first row. In fact, the information regarding the gray gradient is found along the rows only. Other regions do not contain any edge information. Hence, the proposed technique seems to be very useful for brain MR image segmentation to preserve more accurate edges. The theoretical formulations for EGGA is explained below.

Let $(K-1)$ thresholds $[ST_1, ST_2, \ldots, ST_{k-1}]$ divide the image into K classes: c_1 with threshold values$[1, \ldots, ST_1]$, c_2 with values $[ST_1 + 1, \ldots, ST_2]$ and similarly c_k with values$[ST_{K-1} + 1, \ldots, SL]$. The new objective function proposed for multilevel thresholding is given by

$$(S_{opt}T_{opt_1}, S_{opt}T_{opt_2}, \ldots, S_{opt}T_{opt_{K-1}}) = arg \max_{1<ST_i<L} \{\sigma^2_{EGGA}(ST_1, ST_2, \ldots, ST_{K-1})\} \quad (16)$$

where the probability distribution of the regions are expressed as,

$$\begin{aligned}
\omega_1 &= P(c_1) = \sum_{i=1}^{S}\sum_{j=1}^{T_1} p_{ij},\\
\omega_2 &= P(c_2) = \sum_{i=1}^{S}\sum_{j=T_1+1}^{T_2} p_{ij},\\
&\vdots\\
\omega_K &= P(c_K) = \sum_{i=1}^{S}\sum_{j=T_{K-1}+1}^{L} p_{ij}
\end{aligned} \quad (17)$$

The mean vectors for multilevel thresholding are $[\mu_1, \mu_2, \ldots, \mu_K]$ where

$$\begin{aligned}
\mu_1 &= (\mu_{11}, \mu_{12})^T = \left[\sum_{i=1}^{S}\sum_{j=1}^{T_1} ip_{ij}/\omega_1, \sum_{i=1}^{S}\sum_{j=1}^{T_1} jp_{ij}/\omega_1\right]^T\\
\mu_2 &= (\mu_{21}, \mu_{22})^T = \left[\sum_{i=1}^{S}\sum_{j=T_1+1}^{T_2} ip_{ij}/\omega_2, \sum_{i=1}^{S}\sum_{j=T_1+1}^{T_2} jp_{ij}/\omega_2\right]^T\\
&\vdots\\
\mu_K &= (\mu_{K1}, \mu_{K2})^T = \left[\sum_{i=1}^{S}\sum_{j=T_{K-1}+1}^{L} ip_{ij}/\omega_K, \sum_{i=1}^{S}\sum_{j=T_{K-1}+1}^{L} jp_{ij}/\omega_K\right]^T
\end{aligned} \quad (18)$$

The overall mean μ_T is expressed as $\mu_T = (\mu_{T1}, \mu_{T2})^T = \left[\sum_{i=1}^{L}\sum_{j=1}^{L} ip_{ij}, \sum_{i=1}^{L}\sum_{j=1}^{L} jp_{ij}\right]^T$

The between class variance is already defined in Eq. (14).

3.2. Algorithm

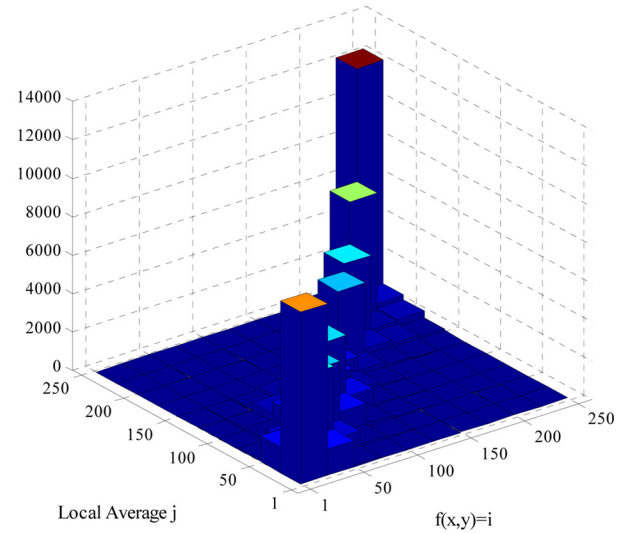
In this section, the authors propose a new algorithm for multilevel thresholding of brain MR images based on gray gradient information. The proposed algorithm is described below and the associated flow chart is displayed in Fig. 6:

begin

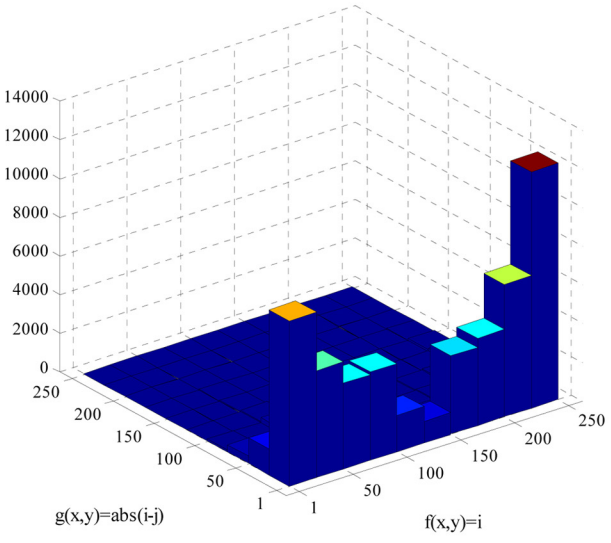
1: A grayscale brain MR image is taken as the input.

2: The 2-D histogram of the image using gray gradient is constructed.

(a)



(b)



(c)

Fig. 5. Two dimensional histogram plot. (a) 2018 image (b) For 2-D Otsu (c) For EGGA.

3: The positions, velocities of all particles in the population are initialized randomly.

4: The Objective function using Eq. (16) is calculated for all the particles.

5: Head leader (HL) and local leader (LL) are assigned from the initial population.

6: The position and velocity of each exploring particle in each group is updated using Eq. (22) defined in Section 4.

7: Then the Local leader in each group is found by comparing the fitness values.

8: From the local leader, the head leader is selected.

9: Again, the position and velocity of the head leader are updated using Eq. (22) as given in Section 4.

10: Using the updated parameter of the head leader, the velocity of the exploring particle is updated in the next iteration using Eq. (22).

11: Then, using the updated velocity of the exploring particle, the updated position using Eq. (22) is found.

12: The new position of aimless particle is computed.

13: The new position of aimless particle is compared with the head leader and local leader.

14: The termination condition is checked.

14: If the condition is not satisfied, then steps 4 to 13 are repeated.

15: After the condition is satisfied, the algorithm produces the optimized result.

end

The 2-D Otsu's method is also implemented for optimum multiple threshold values using Eq. (10).

After the optimum thresholds are obtained, the image is thresholded using the following reconstruction rule for both our proposed method as well as the 2-D Otsu's method. For K optimum thresholds $[T_1, T_2, \ldots, T_K]$, pixels having gray levels less than T_1 retain their values, pixels having gray levels between T_1 and T_2, are assigned T_1, pixels having gray levels between T_2 and T_3, are assigned T_2, similarly, pixels having gray levels between T_K and L, are assigned T_K. The images from the Harvard dataset are considered to experiment. The flow chart of our proposed method is shown in Fig. 6.

The next section explains the proposed optimization technique.

4. Optimization techniques

4.1. Particle swarm optimization (PSO)

Since PSO is a well-established technique, the detail discussion on PSO is not presented here. Readers can refer [30] for more details.

4.2. Coral reef optimization (CRO)

The coral reef optimization (CRO) is inspired by the reproduction and the reef formation behavior of corals. The allocation of space in a reef represents a solution to the optimization problem. In the initial phase some space on the reef is occupied and some space is left free. The free space will be occupied by corals having a better health function. The health function represents the problem fitness function. It is this fight for space, which makes CRO a suitable optimization technique to be considered in this problem. A detail discussion on CRO is found in [31].

4.3. Swallow swarm optimization (SSO)

Swallow swarm optimization (SSO) is a practically new swarm intelligence based algorithm developed by Neshat et al. (2013) [26]. It imitates the behavior of swallow swarms. Some of the peculiar features of swallows such as, social life and migration of large groups, high speed flying, effective communication for food and

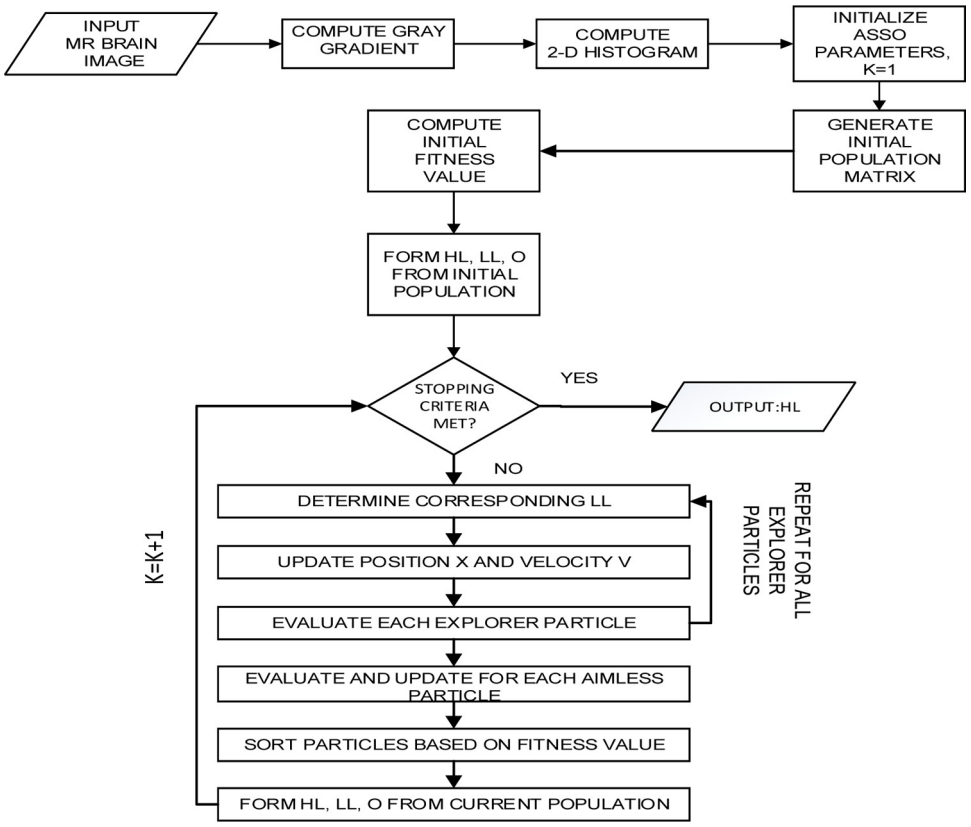


Fig. 6. Flow chart of the proposed method.

security and organization in small groups, form the basis of SSO. Even though the algorithm looks similar to PSO, it has a unique feature. In SSO, three types of particles are identified: leader (L_i), explorer (X_i) and aimless (A_i) particles. Leader particles are again classified into two types: local leaders (LL_i) who take care of sub-colonies, which indicate local optimum. The best particle in each sub-colony is selected as the local leader. Head leaders (HL_i) take care of entire colony, which indicate the global optimum. The head leaders are chosen from the best local leaders. The explorer particles take care of exploration of the search space influenced by many parameters such as position of the local and head leaders and the personal best position.

In each iteration k, particles play different roles (leader or explorer) according to their type. The mathematical equations representing the operations, in each iteration, are given below. All the particles follow the head leaders. They update their direction and converge to the position determined by the head leaders.

The explorer particles update their position as per the following equation.

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{19}$$

where $V_i^{k+1} = V_{HL_i}^{k+1} + V_{LL_i}^{k+1}$

Here, X_i^{k+1} is the new position of the explorer particle obtained by adding a change velocity V_i^{k+1} to its current position X_i^k. The change velocity is obtained by considering $V_{HL_i}^{k+1}$ (change velocity towards the Head Leader) or $V_{LL_i}^{k+1}$ (change velocity towards the Local Leader).

Now the change velocity towards the head leader is determined using the following equation.

$$V_{HL_i}^{k+1} = V_{HL_i}^k + \alpha_{HL} rand() \left(X_{best}^k - X_i^k\right) + \beta_{HL} rand() \left(HL_i^k - X_i^k\right)$$

and the change velocity towards the local leader is determined using the following equation

$$V_{LL_i}^{k+1} = V_{LL_i}^k + \alpha_{LL} rand() \left(X_{best}^k - X_i^k\right) + \beta_{LL} rand() \left(LL_i^k - X_i^k\right)$$

Here, X_{best}^k represents the personal best position of the explorer particles. $V_{HL_i}^k$ and $V_{LL_i}^k$ represents the current velocity vector towards the head leader and the local leader respectively. HL_i^k and LL_i^k represent the position of the head and the local leader respectively. The value of the control parameters (α_{HL}, α_{LL}) which affect the individual particles is determined following [26]. The value of the control parameters $\left(\beta_{HL}, \beta_{LL}\right)$ which affect the collective particles is also determined following [26]. The aimless particles play an important role in SSO. They help in enhancing the exploration of the search space, which might not have been explored by the explorer particles. Further, they may also prevent a particle from being trapped in local optima. The reason behind this is the random movement of these particles and sharing their information with other particles. The aimless particles move in the space using the following equation.

$$A_{i+1} = A_i + \left[rand\left(\{-1, 1\}\right) * \frac{rand(min_s, max_s)}{1 + rand()}\right]$$

where A_{i+1} is the next position of the aimless particle, A_i is the current position of the aimless particle, rand() is a random number uniformly distributed in [0,1].

4.4. The proposed adaptive swallow swarm optimization (ASSO)

In this section, the authors propose a novel adaptive swallow swarm optimization technique. Here, in the adaptive SSO algo-

Table 1
Unimodal benchmark test functions for $d = 30$.

function	Description	Range	Theoretical optima
Sphere Model	$f_1(X) = \sum_{i=1}^{d} x_i^2$	$(-100, 100)^d$	$f_1\left(\vec{0}\right) = 0$
Schwefel's P 2.22	$f_2(X) = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	$(-10, 10)^d$	$f_2\left(\vec{0}\right) = 0$
Schwefel's P 1.2	$f_3(X) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j\right)^2$	$(-100, 100)^d$	$f_3\left(\vec{0}\right) = 0$
Schwefel's P 2.21	$f_4(X) = \max_i \left\{ \lvert x_i \rvert, 1 \le i \le d \right\}$	$(-100, 100)^d$	$f_4\left(\vec{0}\right) = 0$
Generalized Rosenbrock's Function	$f_5(X) = \sum_{i=1}^{d-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2 \right]$	$(-30, 30)^d$	$f_5\left(\vec{0}\right) = 0$
Step Function	$f_6(X) = \sum_{i=1}^{d} (\lfloor x_i + 0.5 \rfloor)^2$	$(-100, 100)^d$	$f_6\left(\vec{0}\right) = 0$
Quartic Function i.e. Noise	$f_7(X) = \sum_{i=1}^{d} i x_i^4 + random[0, 1)$	$(-1.28, 1.28)^d$	$f_7\left(\vec{0}\right) = 0$

Table 2
Multimodal benchmark test functions for $d = 30$.

function	Description	Range	Theoretical optima
Generalized Schwefel's P 2.26	$f_8(X) = \sum_{i=1}^{d} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	$(-500, 500)^d$	$f_8\left(\vec{420.9687}\right) = -12569.5$
Generalized Rastrigin's Function	$f_9(X) = \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	$(-5.12, 5.12)^d$	$f_9\left(\vec{0}\right) = 0$
Ackley's Function	$f_{10}(X) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{d} \cos(2\pi x_i)\right) + 20 + e$	$(-32, 32)^d$	$f_{10}\left(\vec{0}\right) = 0$
Generalized Griewank Function	$f_{11}(X) = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$(-600, 600)^d$	$f_{11}\left(\vec{0}\right) = 0$
Generalized Penalized Function 1	$f_{12}(X) = \frac{\pi}{d}\left\{ 10\sin(\pi y_i) + \sum_{i=1}^{d} (y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_d - 1)^2 \right\}$ where, $+\sum_{i=1}^{d} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$, and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, \ -a < x_i < a \\ k(-x_i - a)^m, x_i < -a \end{cases}$	$(-50, 50)^d$	$f_{12}\left(\vec{1}\right) = 0$
Generalized Penalized Function 2	$f_{13}(X) = 0.1\left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{d} (x_i - 1)^2 \cdot \left[1 + \sin^2(3\pi x_i + 1)\right] + (x_d - 1)^2 \cdot \left[1 + \sin^2(2\pi x_d)\right] \right\} + \sum_{i=1}^{d} u(x_i, 5, 100, 4)$ Where, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, \ -a < x_i < a \\ k(-x_i - a)^m, x_i < -a \end{cases}$	$(-50, 50)^d$	$f_{13}\left(\vec{1}\right) = 0$

rithm, the control parameters of convergence speed α, β are made adaptive and are given as:

$$\alpha_i^{k+1} = \left(\frac{1}{k}\right)^{\left|\frac{bestf(k) - f_i(k)}{bestf(k) - worstf(k)}\right|} \tag{20}$$

$$\beta_i^{k+1} = \left(\frac{1}{k}\right)^{\left|\frac{bestf(k) - f_i(k)}{bestf(k) - worstf(k)}\right|} \tag{21}$$

where k = iteration, $fi\ (k)$ = Fitness value of i^{th} position in the iteration k, $bestf\ (k)$ = Best fitness value in the iteration k, $worstf\ (k)$ = Worst fitness value in the iteration k.

The control parameters decrease with the increase in the iteration k. From the Eqs. (20), (21), it is clear that the control parameters become adaptive in nature. Note that, the control parameter values are made adaptive according to the fitness value. In addition, the ASSO does not require any initialization of the control parameters α and β. Finally, the following new equations for ASSO are proposed.

$$X_i^{k+1} = X_i^k + V_i^{k+1}$$

$$V_i^{k+1} = V_{HL_i}^{k+1} + V_{LL_i}^{k+1}$$

$$V_{HL_i}^{k+1} = V_{HL_i}^k + \alpha_i^{k+1} rand()\left(X_{best}^k - X_i^k\right) + \beta_i^{k+1} rand()\left(HL_i^k - X_i^k\right)$$

$$V_{LL_i}^{k+1} = V_{LL_i}^k + \alpha_i^{k+1} rand()\left(X_{best}^k - X_i^k\right) + \beta_i^{k+1} rand()\left(LL_i^k - X_i^k\right)$$

$$A_{i+1} = A_i + \left[rand\left(\{-1, 1\}\right) * \frac{rand(min_s, max_s)}{1 + rand()} \right] \tag{22}$$

where, α_i^{k+1} *and* β_i^{k+1} are defined in Eqs. (20), (21). The pseudo code for the proposed ASSO algorithm is given in the following section and it is tested with standard benchmark functions.

4.5. Pseudo code for ASSO

The position of N number of particles $X_i(k+1) = \left(x_i^1, x_i^2, \cdots, x_i^d\right)$ for $i = 1, 2, \cdots, N$; d-dimension problem is initialized randomly. Then the objective function $f(X)$ is chosen. Initially, the iteration $k = 1$ is considered and the objective function of the particles $f(X_i)$ for $i = 1, 2, \ldots, N$ is computed.

do *{*

1: Find the *bestf* and *worstf* of the current iteration among the particles.

2: Calculate the *control parameters* using the Eqs. (20), (21).

3: Then calculate the *new position* of *particles* using the Eq. (22).

4: Evaluate the *objective function*.

5: Then chose a particle among N (*say* j) randomly.

if $(f_i > f_j)$

then replace X_i with X_j for a *minimization* problem, otherwise swap X_j with X_i for a *maximization* problem.

end

6: $k = k + 1$.

} ***while*** $(k <= k_{MAX})$ or *end* criterion not satisfied.

Then generate the best solution after ranking.

4.6. Performance evaluation of the proposed ASSO algorithm

Here 23 benchmark test functions [32], defined in Table 1-3 for evaluating the performance of our proposed algorithm, are considered. Table 1 displays 7 numbers of unimodal benchmark test functions, where the rate of convergence is to be focused upon. Table 2 shows 6 numbers of multimodal benchmark test functions with many local minima. Further, Table 3 displays 10 numbers of multimodal benchmark test functions with fixed dimension.

Table 4 displays the parameters used for ASSO and SSO algorithms. Note that 50 independent runs are considered for each algorithm. It is noteworthy to mention here that, the SSO algorithm [26] needs to set the values of control parameters α and β, which has been made adaptive in the proposed technique.

The authors have presented here the best, mean, standard deviation, and average time as performance measures obtained from 50 independent runs (2000 iterations). These values are presented in Tables 5–7 for a comparison. Here, the term '*best*' indicates the minimum value of the objective function, '*mean*' denotes the average best value of the objective function, '*stddev*' implies the standard deviation, and '*avgtime*' represents the average time required to evaluate the test functions (2000 iterations for functions f_1-f_{13}, whereas, 500 iterations for functions f_{14}-f_{23}).

Table 5 displays the comparison results of performance measures of the unimodal benchmark test functions (described in

Table 4
Parameter setting for the benchmark test functions.

Parameters	SSO	ASSO
Number of iterations	2000	2000
Number of particles	30	30
Number of local leaders	5	5
Number of aimless particles	3	3

Table 3
Multimodal benchmark test functions with fixed dimension.

function	Description	Range	Theoretical optima
Shekel's Foxholes Function	$f_{14}(X) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}\right)$	$(-65.5, 65.5)^2$	$f_{14}(-32, -32) \approx 1$
Kowalik's Function	$f_{15}(X) = \sum_{i=1}^{11} \left[a_i - \frac{x_1\left(b_i^2 + b_i x_2\right)}{b_i^2 + b_i x_3 + x_4}\right]^2$	$(-5, 5)^4$	$f_{15}(0.1928, 0.1908, 0.1231, 0.1358) \approx 0.0003075$
Six-Hump Camel-Back Function	$f_{16}(X) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$(-5, 5)^2$	$f_{16}(0.08983, -0.7126)$ or $f_{16}(-0.08983, 0.7126) = -1.0316285$
Branin Function	$f_{17}(X) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right) + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	$-5 \le x_1 \le 10$, $0 \le x_2 \le 15$	$f_{17}(-3.142, 12.275)$ or$f_{17}(3.142, 2.275)$ or$f_{17}(9.425, 2.425) = 0.398$
Goldstein-Price Function	$f_{18}(X) = [1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2\right)] \times [(2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) + 30]$	$(-2, 2)^2$	$f_{18}(0, -1) = 3$
Hartman's Family	$f_{19}(X) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	$(0, 1)^3$	$f_{19}(0.114, 0.556, 0.852) = -3.86$
	$F_{20}(X) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	$(0, 1)^6$	$f_{20}(0.201, 0.15, 0.477, 0.275, 0.311, 0.657) = -3.32$
Shekel's Family	$F_{21}(X) = -\sum_{i=1}^{5}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	$(0, 10)^4$	$f_{21}\left(\vec{X}\right) = -10.1532$
	$F_{22}(X) = -\sum_{i=1}^{7}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	$(0, 10)^4$	$f_{22}\left(\vec{X}\right) = -10.4028$
	$F_{23}(X) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	$(0, 10)^4$	$f_{23}\left(\vec{X}\right) = -10.5363$

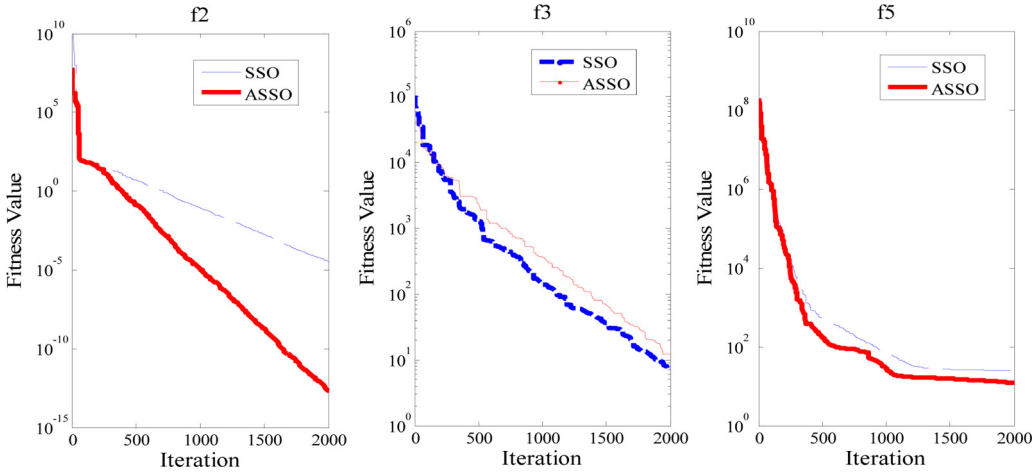


Fig. 7. Fitness value vs Iteration of the unimodal benchmark test functions (f_2, f_3, f_5) using SSO and ASSO algorithm.

Table 5
Comparison of Performance measures.

function	algorithm	best	mean	stddev	avgtime
f_1	ASSO	3.52E-23	4.37E-21	9.28E-21	5.3826
	SSO	2.70E-11	5.90E-10	8.73E-10	6.8102
f_2	ASSO	2.21E-13	1.20E-11	1.77E-11	5.6423
	SSO	1.70E-05	5.57E-05	2.56E-05	7.1158
f_3	ASSO	2.5065	8.4329	4.1564	23.8049
	SSO	5.8577	16.6119	7.9143	25.5340
f_4	ASSO	0.2894	1.8739	1.4729	9.2142
	SSO	1.8867	5.3148	2.4025	11.5310
f_5	ASSO	9.7783	26.3625	17.8455	6.5695
	SSO	15.4965	31.5992	18.2566	9.3166
f_6	ASSO	0.0000	0.0000	0.0000	7.1768
	SSO	0.0000	0.0000	0.0000	6.9559
f_7	ASSO	0.0061	0.0184	0.0058	6.5352
	SSO	0.0175	0.0389	0.0231	7.8790

Table 6
Comparison of Performance measures.

function	algorithm	best	mean	stddev	avgtime
f_8	ASSO	−1.1143E+04	−9.4197E+03	603.1573	6.2570
	SSO	−1.0885E+04	−1.0243E+04	299.6812	7.5741
f_9	ASSO	43.7891	69.4523	19.6658	6.1063
	SSO	36.7787	57.6203	10.6023	7.4417
f_{10}	ASSO	2.9725E-11	9.9978E-08	2.6254E-07	5.9324
	SSO	1.2137E-04	0.6103	0.7364	7.3792
f_{11}	ASSO	0.0000	5.3745E-04	0.0019	6.6301
	SSO	5.3732E-10	2.9804E-04	0.0013	8.0241
f_{12}	ASSO	3.4762E-20	0.0075	0.0383	19.6419
	SSO	9.9101E-08	0.3056	0.3288	22.0268
f_{13}	ASSO	1.2981E-21	3.9921E-04	0.0020	18.6264
	SSO	3.1684E-09	0.5729	2.1619	21.9863

Table 1). It is observed that the best value is minimum for ASSO as compared to SSO for all the functions. This results in smaller mean values, which is also seen in the case of ASSO. The standard deviation should be as small as possible to demonstrate the stability of an algorithm. The average time illustrates the convergence speed of an algorithm. Smaller is the average time, faster is the convergence rate. It is observed that ASSO takes smaller average time in most of the functions. Fig. 7 shows the rate of convergence of the proposed algorithm. It is observed that our proposed algorithm (ASSO) exhibits better convergence than SSO for most of the functions. However, the ASSO lags behind SSO in case of function f_3. The reason may be the number of iterations considered here, i.e. 2000 which may not be adequate for ASSO to converge.

Table 7
Comparison of Performance measures.

function	algorithm	best	mean	stddev	avgtime
f_{14}	ASSO	1.0978	1.0878	1.2948E-16	1.9330
	SSO	1.0978E+00	1.0878E+00	8.2354E-08	2.0527
f_{15}	ASSO	4.9254E-04	7.4491E-04	1.1173E-04	0.5908
	SSO	6.6904E-04	7.8456E-04	7.6422E-05	0.6806
f_{16}	ASSO	−1.1348	−1.1244	6.5184E-16	0.4940
	SSO	−1.1348	−1.1244	1.4990E-10	0.5775
f_{17}	ASSO	0.4377	0.4337	8.2201E-11	0.4848
	SSO	0.4377	0.4337	1.1147E-08	0.5654
f_{18}	ASSO	3.3000	3.2700	1.1631E-15	0.4848
	SSO	3.3000	3.2700	1.2051E-08	0.5708
f_{19}	ASSO	−4.2491	−4.2105	2.6808E-15	0.6666
	SSO	−4.2491	−4.2105	2.4153E-08	0.7349
f_{20}	ASSO	−3.6542	−3.6209	3.1823E-04	0.6580
	SSO	−3.6542	−3.6197	0.0011	0.7484
f_{21}	ASSO	−11.1685	−11.0670	2.1918E-08	0.6961
	SSO	−11.1682	−11.0493	0.0303	0.7857
f_{22}	ASSO	−11.4432	−11.3392	1.5489E-09	0.7645
	SSO	−11.4430	−11.1912	0.4343	0.8525
f_{23}	ASSO	−11.5900	−11.4847	3.2036E-07	0.8672
	SSO	−11.5884	−11.4203	0.1023	0.9545

Similarly, Table 6 shows the comparison results of performance measures of the multimodal benchmark test functions (described in Table 2) and Table 7 shows the results of the multimodal benchmark test functions with fixed dimension (described in Table 3). Fig. 8 and Fig. 9 presents the convergence rate of the proposed ASSO algorithm for rest of the function groups. It is evident from the graph that our proposed ASSO algorithm shows better convergence than the SSO algorithm for most of the benchmark functions. So, it can be claimed that the proposed algorithm can be implemented for solving various engineering optimization problems efficiently with a faster convergence rate. With the performance of ASSO algorithm, the authors are motivated to investigate its application in multilevel thresholding of brain MR images.

5. Results and discussions

The proposed work is tested with 100 slices (from slice 15 to slice 114) of T_2-weighted structural brain MR images taken from Harvard medical education [33]. Different slices of brain images in the axial plane at same time are considered for the experiment. The experiments are carried out on Intel core i-5 processor with 4 GB RAM running under Windows 8.1 operating system. Results are provided after 50 runs of each of the algorithms. The dimension of each of the images is resized to 256 × 256. The soft

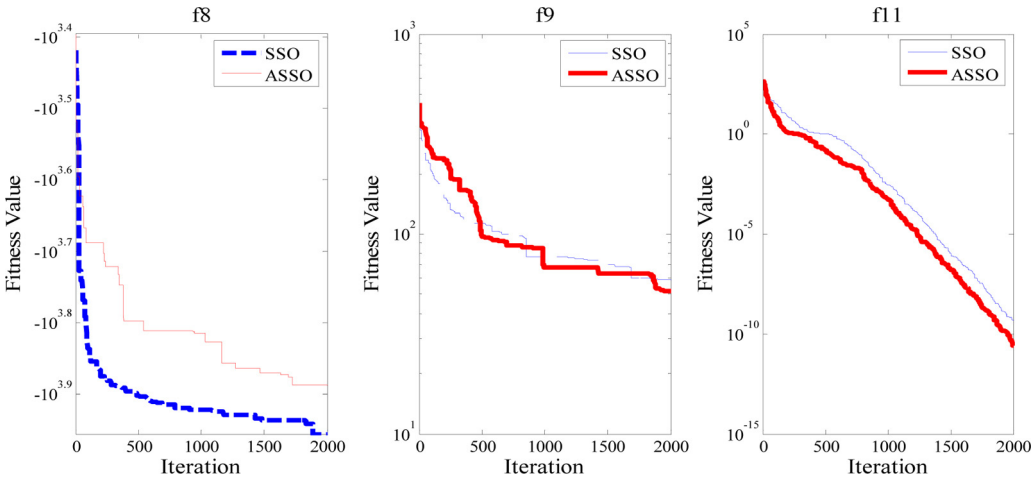


Fig. 8. Fitness value vs Iteration of the unimodal benchmark test functions (f_8, f_9, f_{11}) using SSO and ASSO algorithm.

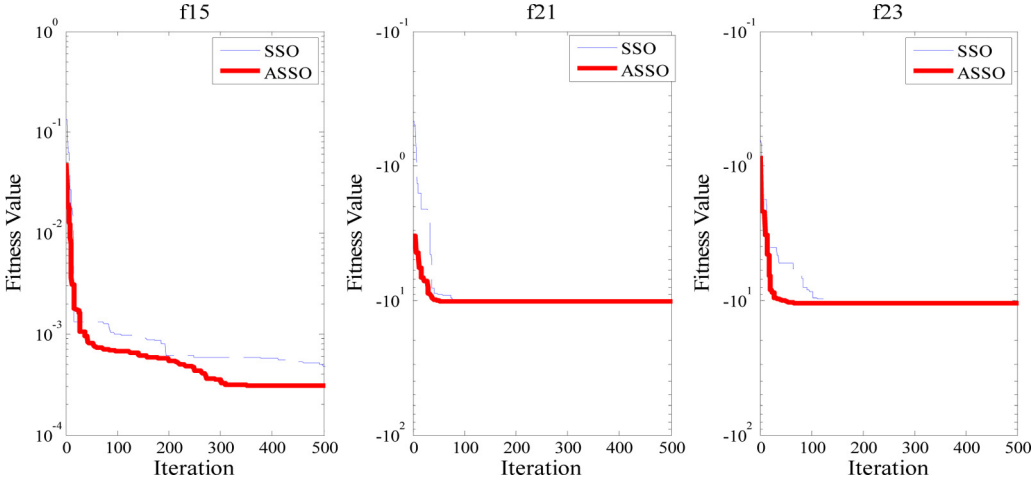


Fig. 9. Fitness value vs Iteration of the unimodal benchmark test functions (f_{15}, f_{21}, f_{23}) using SSO and ASSO algorithm.

Table 8
Parameter setting for the soft computing techniques.

Algorithm	Parameters
ASSO	$N = 30$, Iter = 100, Number of local leaders = 5, Number of aimless particles = 3
SSO	$N = 30$, Iter = 100, Number of local leaders = 5, Number of aimless particles = 3
CRO	$NP = 30$, $Iter = 100$, Broadcast probability = 0.9, Asexual reproduction probability = 0.1 Free/total initial proportion (occupancy parameter) = 0.6, Depredation probability = 0.1
PSO	$NP = 30$, $Iter = 100$, $weight = 0.9$, $C_1 = C_2 = 2.0$

computing algorithms ASSO, SSO, CRO, and PSO are implemented using MATLAB. Here five different well known performance metrics SSIM, FSIM, IDM, Contrast and PSNR for evaluating the thresholding performance are used. The parameter setting for soft computing algorithms is shown in Table 8. The number of iterations define the termination condition in each of the optimization algorithms.

Optimal Multilevel thresholding of 2D histogram based brain MR images using ASSO, SSO, and CRO have never been investigated. It can be seen that the histogram of brain image is multimodal. The initial solution is generated randomly in each of the optimization algorithms. To get the best threshold value, each of the algorithms is repeated for 50 independent runs.

Table 9 shows the comparison of best average objective function values at different threshold levels $m = 2, 3, 4, 5$. Higher is the average objective function value, better is the thresholding performance. It is observed that values obtained using EGGA optimized with ASSO are higher as compared to other algorithms. The possible reason could be the structure of the newly proposed fitness function and the novel optimization algorithm. In this case, the difference between the pixel value and its average value is considered for computing the variance. Variance is maximized here, which leads to higher objective function values. The average objective function values increase with increase in level of thresholds as expected. The number of function evaluations increase with higher levels of thresholding. This is the reason why one observes higher values of the objective function in Table 9. The best objective function values are obtained by using the objective function in four soft computing techniques ASSO, SSO, CRO and PSO. It is observed that values obtained with ASSO outperforms the other techniques.

Here, Table 10 displays the comparison results using average Structured Similarity (SSIM) index [34]. Table 11 shows the average Feature Similarity (FSIM) [35] index. These indices measure

Table 9
Best average objective function values (**Computed over 100 slices**).

m	Soft Computing Techniques	Algorithms		
		EGGA	2-D Otsu	1-D Otsu
2	ASSO	258.41	256.50	252.55
	SSO	255.88	253.96	250.04
	CRO	253.31	251.42	247.34
	PSO	250.79	248.91	245.06
3	ASSO	330.03	310.25	284.58
	SSO	326.76	307.18	281.76
	CRO	323.49	304.11	278.94
	PSO	320.26	301.07	276.15
4	ASSO	1.67E + 06	1.60E + 06	1.09E + 06
	SSO	1.66E + 06	1.59E + 06	1.08E + 06
	CRO	1.64E + 06	1.57E + 06	1.07E + 06
	PSO	1.63E + 06	1.56E + 06	1.06E + 06
5	ASSO	2.08E + 06	1.64E + 06	1.41E + 06
	SSO	2.07E + 06	1.63E + 06	1.40E + 06
	CRO	2.04E + 06	1.61E + 06	1.38E + 06
	PSO	2.03E + 06	1.60E + 06	1.37E + 06

Table 10
Average SSIM value (**Computed over 100 slices**).

m	Soft Computing Techniques	Algorithms		
		EGGA	2-D Otsu	1-D Otsu
2	ASSO	0.9811	0.9459	0.9293
	SSO	0.9714	0.9365	0.9201
	CRO	0.9617	0.9271	0.9109
	PSO	0.9521	0.9179	0.9018
3	ASSO	0.9890	0.9567	0.9406
	SSO	0.9792	0.9472	0.9313
	CRO	0.9694	0.9377	0.9220
	PSO	0.9597	0.9284	0.9128
4	ASSO	0.9939	0.9575	0.9437
	SSO	0.9841	0.9480	0.9344
	CRO	0.9743	0.9385	0.9251
	PSO	0.9645	0.9291	0.9158
5	ASSO	0.9980	0.9664	0.9499
	SSO	0.9910	0.9568	0.9405
	CRO	0.9782	0.9472	0.9311
	PSO	0.9684	0.9378	0.9218

Table 11
Average FSIM value (**Computed over 100 slices**)

m	Soft Computing Techniques	Algorithms		
		EGGA	2-D Otsu	1-D Otsu
2	ASSO	0.9965	0.9607	0.9438
	SSO	0.9915	0.9559	0.9391
	CRO	0.9865	0.9511	0.9344
	PSO	0.9816	0.9464	0.9297
3	ASSO	0.9959	0.9672	0.9510
	SSO	0.9949	0.9624	0.9463
	CRO	0.9899	0.9576	0.9416
	PSO	0.9850	0.9528	0.9369
4	ASSO	0.9974	0.9646	0.9508
	SSO	0.9964	0.9598	0.9461
	CRO	0.9914	0.9550	0.9414
	PSO	0.9865	0.9502	0.9367
5	ASSO	0.9990	0.9684	0.9519
	SSO	0.9980	0.9636	0.9472
	CRO	0.9931	0.9588	0.9425
	PSO	0.9884	0.9540	0.9378

the thresholding performance. They are considered as the standard measures for evaluating the thresholding performance. Higher is the value of average SSIM and FSIM, closer is the thresholded image to the original image. It is observed that both SSIM and FSIM are the highest for thresholding level 5. This indicates that image thresholded at level 5 is more identical to the original image. Further, the values are higher for EGGA as compared to other methods. Most importantly, we have achieved an improvement of about 3–4% over 2-D Otsu's method, in case of both SSIM and FSIM values. Usually, the SSIM and FSIM values depend on the covariance, which appears in the numerator term [34,35]. In the proposed method, the gray gradient between the pixel value and its average value reduces the information loss due to preservation of edges. As a result, the covariance is improved. This reason clarifies why the similarity indices are higher for our method.

Table 12
Average IDM value (**Computed over 100 slices**).

m	Soft Computing Techniques	Algorithms		
		EGGA	2-D Otsu	1-D Otsu
2	ASSO	0.0403	0.0439	0.0535
	SSO	0.0399	0.0435	0.0525
	CRO	0.0394	0.0431	0.0523
	PSO	0.0390	0.0423	0.0514
3	ASSO	0.0448	0.0537	0.0646
	SSO	0.0442	0.0529	0.0636
	CRO	0.0438	0.0524	0.0630
	PSO	0.0433	0.0518	0.0621
4	ASSO	0.0588	0.0666	0.0794
	SSO	0.0580	0.0660	0.0786
	CRO	0.0577	0.0652	0.0775
	PSO	0.0562	0.0647	0.0770
5	ASSO	0.0543	0.0657	0.0764
	SSO	0.0537	0.0649	0.0758
	CRO	0.0532	0.0641	0.0752
	PSO	0.0524	0.0638	0.0745

Table 13
Average Contrast value **(Computed over 100 slices)**

m	Soft Computing Techniques	Algorithms		
		EGGA	2-D Otsu	1-D Otsu
2	ASSO	1294.4037	1120.0471	1064.1518
	SSO	1283.568	1113.908	1057.576
	CRO	1271.7323	1105.7689	1046.0002
	PSO	1255.0250	1093.7412	1033.5302
3	ASSO	1009.9657	860.0875	829.9754
	SSO	999.966	853.552	823.738
	CRO	987.9663	843.0165	811.5006
	PSO	983.0667	839.5663	807.3456
4	ASSO	1127.9880	941.0491	864.0509
	SSO	1118.8	933.712	856.486
	CRO	1109.6120	923.3749	842.9211
	PSO	1094.5359	910.1311	832.4419
5	ASSO	1151.8461	969.6644	935.0380
	SSO	1143.412	962.044	923.8
	CRO	1135.9779	957.4236	917.5620
	PSO	1128.6581	940.8993	913.4164

Table 12 displays average inverse difference moment (IDM) values. Table 13 shows the average contrast values to measure the thresholding performance [36]. IDM and contrast are inversely related. Higher the IDM, lower the contrast. A high value of contrast is desired for a thresholded image at a particular level of thresholding. Consequently, a lower value of IDM is desired. It is observed that the contrast is the highest and the IDM is the lowest for EGGA technique as compared to other techniques at a particular thresholding level. More significantly, an improvement of about 15–20% is observed in the contrast values using our proposed technique as compared to the 2-D Otsu's method. The threshold values are obtained after optimizing the objective function defined in Eq. (16). When the brain MR image is thresholded with these optimum threshold values, the contrast feature is also improved. The objective function used with our proposed method yields better

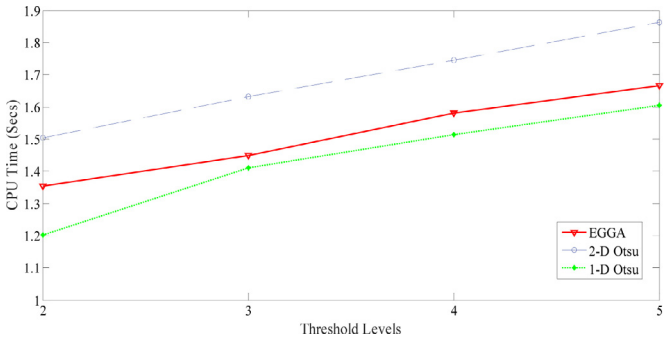


Fig. 10. CPU Time comparison.

values, thus improving the above performance measures. In addition, the shape of the edges is preserved by using the gray gradient information.

Table 14 shows the comparison results of different techniques based on average peak signal to noise ratio (PSNR) [36]. A high value of average PSNR indicates better thresholding performance. It is observed that PSNR is the highest for thresholding level 5. This indicates higher thresholding level generates an image closer to the original one. Further, the PSNR value is also higher for EGGA as compared to other techniques at all thresholding levels. Significant improvement in average PSNR values (10–15%) is achieved using our method, because the edges are accurately preserved. Moreover, the reduction of information loss due to consideration of gray gradient information also plays an important role in improving the PSNR.

Table 14
Average PSNR values (**Computed over 100 slices**).

m	Soft Computing Techniques	Algorithms		
		EGGA	2-D Otsu	1-D Otsu
2	ASSO	24.4377	22.3229	21.2523
	SSO	24.166	22.092	21.032
	CRO	23.9233	21.8811	20.8317
	PSO	23.6551	21.6554	20.6145
3	ASSO	27.1045	23.5194	21.7416
	SSO	26.846	23.336	21.556
	CRO	26.5675	23.1026	21.3104
	PSO	26.3118	22.8516	21.1370
4	ASSO	28.1022	25.6156	22.5090
	SSO	27.824	25.362	22.296
	CRO	27.5458	25.1084	22.0330
	PSO	27.2703	24.8573	21.8723
5	ASSO	29.6920	26.7024	24.4183
	SSO	29.398	26.438	24.226
	CRO	29.1040	26.1736	23.9637
	PSO	28.8130	25.9119	23.7439

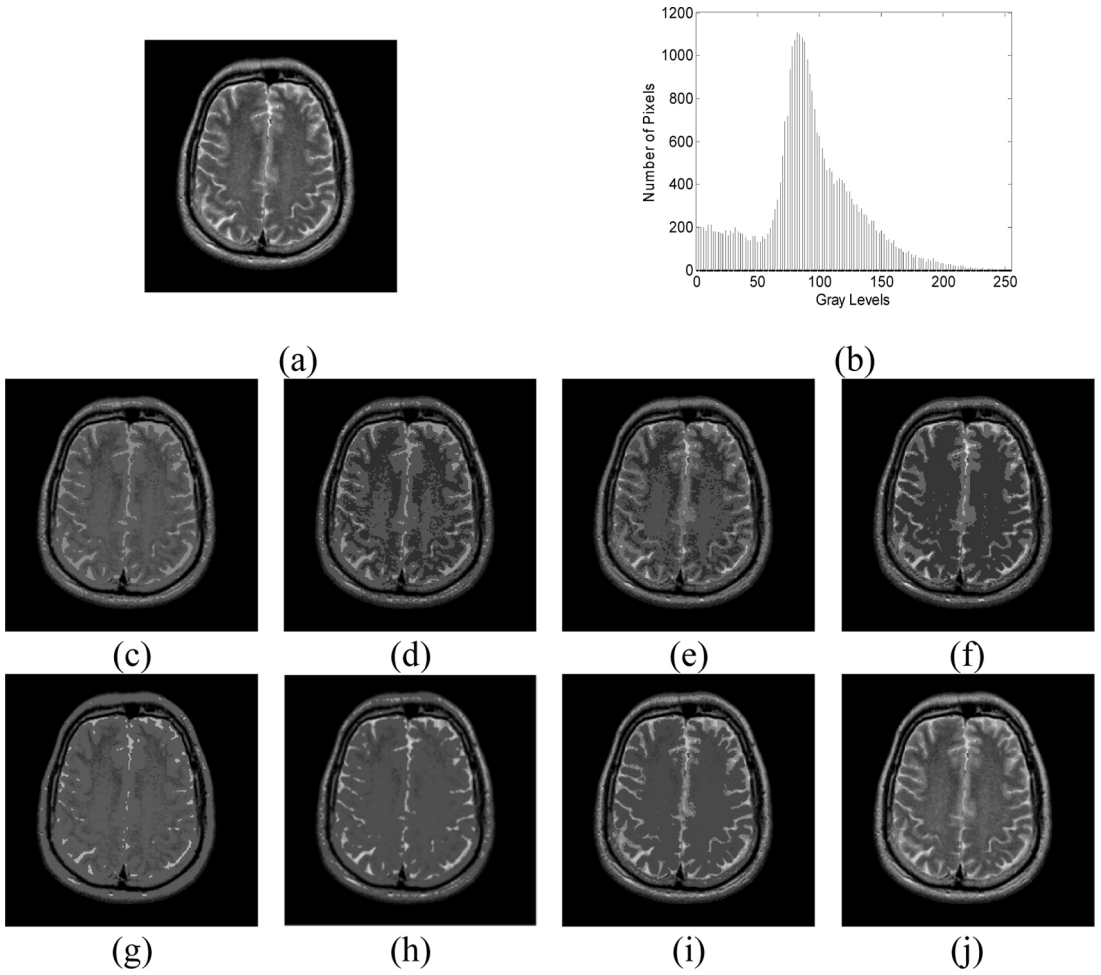


Fig. 11. Example of multilevel thresholded brain MR images. (a) Original example image of slice-95, (b) Histogram of (a), (c) Thresholded image at level-2 [91,138] with 2-D Otsu, (d) Thresholded image at level-3 [47,84,147] with 2-D Otsu, (e) Thresholded image at level-4 [46,79,112,187] with 2-D Otsu, (f) Thresholded image at level-5 [54,101,154,187,229] with 2-D Otsu, (g) Thresholded image at level-2 [90,175] with EGGA, (h) Thresholded image at level-3 [160,187,205]with EGGA, (i) Thresholded image at level-4 [80,143,169,203]with EGGA, (j) Thresholded image at level-5 [73,112,128,144,188] with EGGA.

Table 15
CPU Time computation.

Image	m	CPU Time in seconds		
		EGGA	2-D Otsu	1-D Otsu
Slice-65	2	1.3532	1.5036	1.2016
	3	1.4489	1.6321	1.4113
	4	1.5804	1.7448	1.5138
	5	1.6660	1.8622	1.6052

In addition, the noise information is reduced due to the difference operation involved in computing the gray gradient.

Table 15 shows the variation of CPU time for an example slice-65 of the T_2-weighted brain MR image. It is noteworthy to mention here that the 1-D Otsu's method is faster than other methods. This can be further made faster through projection sum or difference computations. In summary, the proposed EGGA has shown improved results with respect to different measures displayed in this work. The EGGA method based on the gray gradient information is the second contestant in this respect. Interestingly, a saving of about 10% in execution time is achieved using EGGA as compared to the 2-D Otsu's method. It is noteworthy to mention here that the optimum value of '*S*' is computed once only. The same value is used repeatedly for computing multiple threshold values$ST_1, ST_2, \ldots, ST_{k-1}$. However, 1-D multi Otsu's method shows the least computation time compared to other methods. The associated graph for CPU time comparison is displayed in Fig. 10.

Fig. 11 displays the thresholded brain MR images at levels 2,3,4,5 for an example slice-95. It is observed that brain images thresholded at level 5 are more identical to the original brain image. Further, thresholding using EGGA gives visually better images than the 2-D Otsu technique. The histograms of the brain images exhibit multimodality thereby justifying the use of multilevel thresholding. At lower levels of thresholds, the brain MR regions are not clearly distinguishable, whereas at higher levels of thresholds they are accurately identified. To be more specific, thresholding at level 5 prominently displays various regions in the brain image as compared to thresholding at level 2 or 3 or 4. As observed from the figures, the brain images contain very fine edges which need to be accurately identified. Our proposed method outperforms the standard Otsu's method, because the vital edge information is preserved.

6. Conclusion

In the paper, the authors have implemented the idea of gray gradient information for brain MR image thresholding. As the brain image contains many regions, multilevel thresholding is used. A relatively encouraging evolutionary computing technique has been used to optimize the gray gradient information to get the optimal multilevel threshold values. It is observed from evaluation metrics that the EGGA yields significantly better results as compared to the Otsu's method using the 1-D and the 2-D histogram. The reason is that the maximization of gray gradient information provides us the best objective function values. In addition, the EGGA is faster than 2-D Otsu's method presented here for multilevel thresholding. The adaptive swallow swarm optimization algorithm is also very effective as it involves very few parameters for tuning. The adaptive nature of ASSO prevents initializing any control parameters for convergence rate control. The proposed scheme seems to be very useful for brain MR image segmentation, because the shape of the edges is found prominently. Gray gradient information based method may lead to further research in the area of multilevel image thresholding.

References

[1] E. Smistad, T.L. Falch, M. Bozorgi, A.C. Elster, F. Lindseth, Medical image segmentation on GPUs-A comprehensive review, Med. Image Anal. 20 (1) (2015) 1–18.
[2] P.K. Sahoo, S. Soltani, A.K. Wong, A survey of thresholding techniques, Comput. Vis. Graph. Image Process. 41 (2) (1988) 233–260.
[3] B. Sankur, M. Sezgin, Image Thresholding Techniques: a survey over categories, Pattern Recogn. 34 (2) (2001) 1573–1583.
[4] D. Mortazavi, A.Z. Kouzani, H. Soltanian-Zadeh, Segmentation of multiple sclerosis lesions in MR images: a review, Neuroradiology 54 (4) (2012) 299–320.
[5] A.K. Bhandari, V.K. Singh, A. Kumar, G.K. Singh, Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy, Expert Syst. Appl. 41 (7) (2014) 3538–3560.
[6] N.S. Raja, S.A. Sukanya, Y. Nikita, Improved PSO based multi-level thresholding for cancer infected Breast thermal images using otsu, Procedia Comput. Sci. 48 (2015) 524–529.
[7] A.K. Bhandari, A. Kumar, G.K. Singh, Modified artificial bee colony based computationally efficient multilevel thresholding for satellite image segmentation using Kapur's, Otsu and Tsallis functions, Expert Syst. Appl. 42 (3) (2015) 1573–1601.
[8] N. Nabizadeh, N. John, C. Wright, Histogram-based gravitational optimization algorithm on single MR modality for automatic brain lesion detection and segmentation, Expert Syst. Appl. 41 (17) (2014) 7820–7836.
[9] A. Ortiz, J.M. Gorriz, J. Ramirez, D. Salas-Gonzalez, Improving MR brain image segmentation using self-organizing maps and entropy-gradient clustering, Inf. Sci. 262 (2014) 117–136.
[10] S. Manikandan, K. Ramar, M.W. Iruthayarajan, K.G. Srinivasagan, Multilevel thresholding for segmentation of medical brain images using real coded genetic algorithm, Measurement 47 (2014) 558–568.
[11] S. Hadjidimitriou, V. Charisis, G. Sergiadis, L. Hadjileontiadis, V. Kouloulias, M. Papathanasiou, D. Kelekis, Fast histogram-based brain segmentation from T1-weighted MR images using morphological processing and geometric criteri, Phys. Med. 30 (2014) e88.
[12] R. Wang, C. Li, J. Wang, X. Wei, Y. Li, Y. Zhu, S. Zhang, Threshold segmentation algorithm for automatic extraction of cerebral vessels from brain magnetic resonance angiography images, J. Neurosci. Methods 241 (2015) 30–36.
[13] P. Mesejo, A. Valsecchi, L. Marrakchi-Kacem, S. Cagnoni, S. Damas, Biomedical image segmentation using geometric deformable models and metaheuristics, Comput. Med. Imaging Graph. 43 (2015) 167–178.
[14] K. Hammouche, M. Diaf, P. Siarry, A comparative study of various meta-Heuristic techniques applied to the multilevel thresholding problem, Eng. Appl. Artif. Intell. 23 (5) (2010) 676–688.
[15] D.Y. Huang, C.H. Wang, Optimal multi-level thresholding using a two-stage Otsu optimization approach, Pattern Recognit. Lett. 30 (3) (2009) 275–284.
[16] B. Akay, A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding, Appl. Soft Comput. 13 (6) (2013) 3066–3091.
[17] S. Sarkar, S. Das, S.S. Chaudhuri, A multilevel color image thresholding scheme based on minimum cross entropy and differential evolution, Pattern Recognit. Lett. 54 (2015) 27–35.
[18] S. Roy, U. Kumar, D. Chakraborty, S. Nag, A. Mallick, S. Dutta, Comparative Analysis of Cuckoo Search Optimization-Based Multilevel Image Thresholding, In Intelligent Computing, Communication and Devices, Springer, India, 2015, pp. 327–342.
[19] I. Brajevic, M. Tuba, Cuckoo Search and Firefly Algorithm Applied to Multilevel Image Thresholding In Cuckoo Search and Firefly Algorithm, Springer International Publishing, 2014, pp. 115–139.
[20] Y. Liu, C. Mu, W. Kou, J. Liu, Modified particle swarm optimization-based multilevel thresholding for image segmentation, Soft Comput. 19 (5) (2014) 1311–1327.
[21] K. Chen, Y. Zhou, Z. Zhang, M. Dai, Y. Chao, J. Shi, Multilevel image segmentation based on an improved firefly algorithm, Math. Probl. Eng. (2016) 1–12.
[22] A.K. Bhandari, A. Kumar, S. Chaudhary, G.K. Singh, A novel color image multilevel thresholding based segmentation using nature inspired optimization algorithms, Expert Syst. Appl. 63 (2016) 112–133.
[23] F. Hamdaoui, A. Sakly, A. Mtibaa, An efficient multi level thresholding method for image segmentation based on the hybridization of modified PSO and otsu's method, computational intelligence applications in modeling and control, Springer International Publishing, 2015, pp. 343–367.
[24] P.D. Sathya, R. Kayalvizhi, Optimal multilevel thresholding using bacterial foraging algorithm, Expert Syst. Appl. 8 (12) (2011) 15549–15564.
[25] M. Maitra, A. Chatterjee, A novel technique for multilevel optimal magnetic resonance brain image thresholding using bacterial foraging, Measurement 41 (10) (2008) 1124–1134.
[26] M. Neshat, G. Sepidnam, M. Sargolzaei, Swallow swarm optimization algorithm: a new method to optimization, Neural Comput. Appl. 23 (2) (2013) 429–454.
[27] S. Agrawal, R. Panda, L. Dora, A study on fuzzy clustering for magnetic resonance brain image segmentation using soft computing approaches, Appl. Soft Comput. 24 (2014) 522–533.

[28] N. Otsu, A threshold selection method from gray level histograms, Automatica 11 (285–296) (1975) 23–27.
[29] D. Martin, C. Fowlkes, D. Tal, J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Computer Vision, 2001. ICCV 2001, 2 (2001) 416–423.
[30] J. Kennedy, R. Eberhart, Particle swarm optimization, IEEE Int. Conf. Neural Netw. 4 (1995) 1942–1948.
[31] S. Salcedo-Sanz, J. Del Ser, I. Landa-Torres, S. Gil-López, J.A. Portilla-Figueras, The Coral Reefs Optimization algorithm: a novel metaheuristic for efficiently solving optimization problems, Sci. World J. (2014) 1–12.
[32] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, evolutionary computation, IEEE Trans. 3 (2) (1999) 82–102.
[33] Harvard Medical School. Retrieved June 2015, from http://www.med.harvard.edu/AANLIB [online].
[34] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process. 13 (4) (2004) 600–612.
[35] L. Zhang, L. Zhang, X. Mou, D. Zhang, FSIM: a feature similarity index for image quality assessment, IEEE Trans. Image Process. 20 (8) (2011) 2378–2386.
[36] A. Baraldi, F. Parmiggiani, An investigation of the textural characteristics associated with gray level Co-occurrence matrix statistical parameters, IEEE Trans. Geosci. Remote Sens. 33 (2) (1995) 293–304.