SN Computer Science (2024) 5:417
https://doi.org/10.1007/s42979-024-02674-y

ORIGINAL RESEARCH

Hybrid Noise Reduction Filter Using the Gaining–Sharing Knowledge-Based Optimization and the Whale Optimization Algorithms

Mehrdad Nabahat[1] · **Farzin Modarres Khiyabani**[1] · **Nima Jafari Navmipour**[2]

Received: 28 September 2023 / Accepted: 31 January 2024

Abstract

Noise reduction is one of the main challenges for researchers. Classical image de-noising methods reduce the image noise but sometimes lose image quality and information, such as blurring the edges of the image. To solve this challenge, this work proposes two optimal filters based on a generalized Cauchy (GC) distribution and two different nature-inspired algorithms that preserve image information while decreasing the noise. The generalized Cauchy filter and the bilateral filter are two parameter-based filters that significantly remove image noise. Parameter-based filters require proper parameter selection to remove the noise and maintain the edge details. To this end, two filters are considered. In the previous works, the parameters of the mask that was made with the GC function were optimized and the mask size was considered fixed. By studying different noisy images, we find that the selected mask size significantly impacts the designed filter performance. Therefore in this paper, a mask is designed using the GC function to formulate the first filter, and despite the optimization of the filter parameters, the selected mask size is also optimized using the peak signal-to-noise ratio (PSNR) as a fitness function. In most metaheuristic-based bilateral filters, only the domain and range parameters, which are based on Gaussian distribution, are optimized and the neighboring radius is a constant value. Filter results on different noisy images show that the neighboring radius has a major effect on the filter performance. Since the filter designed with the GC function causes significant noise removal, this function is effective, and on the other hand, it's almost similar behavior with the Gaussian function has caused it to be combined with the bilateral filter to design the second filter in this paper. The kernel of the domain and range is considered to be the GC function instead of the Gaussian function. The domain and range parameters and the neighboring radius are optimized using the PSNR as a fitness function. With the help of optimization algorithms such as the whale optimization algorithm and the Gaining sharing knowledge-based optimization algorithm, bilateral filter; and GC filter parameters are optimized. Finally, the performance of the proposed filters is investigated on images corrupted by Gaussian and impulse noise. It is compared with other classical filters, the particle swarm optimization (PSO) based GC filter, and two PSO-based bilateral filters on various images. The experimental findings demonstrate that the suggested filters outperform the others.

Keywords Bilateral filter · Gaining sharing knowledge-based optimization algorithm · Generalized Cauchy filter · Image de-noising · Image noise · Particle swarm optimization algorithm · Whale optimization algorithm

✉ Farzin Modarres Khiyabani
farzin.modarres@gmail.com

Mehrdad Nabahat
M_nteach2009@yahoo.com

Nima Jafari Navmipour
Jafari@iaut.ac.ir

1 Department of Mathematics, Tabriz Branch, Islamic Azad University, Tabriz, Iran

2 Department of Computer Engineering, Tabriz Branch, Islamic Azad University, Tabriz, Iran

Introduction

In recent years, digital images have found numerous applications in various analysis and engineering sciences, such as medical imaging, resonance imaging, computed tomography, satellite observation, etc. However, most sensor-captured images are corrupted by noise [1]. Various types of noises caused by hardware or atmospheric factors affect the image quality [2], so most researchers have considered image de-noising to improve image quality by removing noise from the image while preserving structural information [3]. Noise

Published online: 09 April 2024

removal should be done early, not affecting other stages of image analysis, such as image segmentation, image classification, etc. Image de-noising can be done by hardware or software approaches. Despite new advances in optics and hardware to reduce the adverse effects of image noise, software-based methods, including some parameter-based algorithms, have been highly considered because they are device-independent and widespread.

Three types of image de-noising methods exist: filter-based, transform-based, and non-local.

Recently, various filter-based methods have been divided into linear and non-linear categories. Among the linear filters, the Mean filter [4] can be mentioned, which helps eliminate image noise but blurs the edges of the image. Another linear filter that can be said is the Wiener filter [5]. This filter eliminates the noise and blurring of a signal that has damaged the image. It minimizes the square errors associated with inverse filtering and noise removal. Although the Wiener filter can effectively remove Gaussian noise, it loses some details about the image edge. The median filter [6], which is more suitable for salt and pepper noise while reducing noise, is one of the most common non-linear filters. The main idea behind this filter is to insert the current pixel value with median values adjacent to the stated pixel. This filter is complicated and expensive because it takes a long time to calculate the median in each window. The bilateral filter [7] is another non-linear filter, a spatial Mean filter that protects the edges of the image and is an efficient filter for noise removal. The performance of this filter depends on the correct selection of the parameters of the filter, which is not related to the image and requires experimental efforts. Non-linear filters are preferred over linear ones due to their superior image noise removal and edge preservation performance.

Transform-based methods are also efficient for image de-noising [8]. The wavelet transform is one of these transformations. In the transform-based methods, the image domain is first changed by applying some linear transformations on the image. Then, non-linear or multiple operations are performed in this domain, and an inverse linear transformation returns the image domain. One transform-based method is the BLS-GSM method [9], a wavelet domain method. The basic idea of this method is that when the images are split into wavelengths in the multidimensional display, the adjacency of each wavelet coefficient is modeled using Gaussian Scale Mix (GSM), and noise-free coefficients are estimated using Bayesian least squares (BLS). Another transform-based method is three-dimensional block-matching (BM3D) algorithm [10]; for reducing image noise. The idea of using this method to eliminate the noise is to enhance the dispersal of the image, which has scattered representations in the transform domain enhanced by two-dimensional grouping patches similar to the three-dimensional groups. A wavelet-based approach using the least square approach is proposed by Vishnu et al. [11]. The noisy image is considered and given as an input to different filters that perform decomposition and then entered into a Least Square weighted regularization stage. Wavelet-based algorithms have some defects, including a lack of good directionality and calculation complexity, and are time-consuming.

Other noise reduction methods are non-local methods that estimate the intensity of all pixels based on the information about the whole image and thus take advantage of similar patterns and features in an image; in this regard, the Non-Local Mean filter [12] can be mentioned. Unlike local mean filters, which smooth the image by replacing the mean value of a group of pixels located adjacent to the target pixels with the value of the target pixels, the non-local mean method smooths the image by calculating the mean of all the pixels which the amount of similarity between these pixels and the target pixels has weighted. An iterative point filtering algorithm based on the Bayesian non-local mean filter model for ultrasound images is proposed by Zhou et al. [13]. Mehta and Prasad [14] presented a method for speckle noise reduction and entropy minimization of medical contrast-enhanced ultrasound images. Their method has been implemented and tested on different images using a filter bank. The statistical feature of the noise is used to apply the Bayesian non-local mean model to reconstruct the image, obtain the critical probability density function, and provide an iterative filter. For reducing the deviations created in the noise-free patches, the nearest statistical neighbor has been used as a measure of the set of dissimilar neighbors [15]. This method works better for white and color noise than the traditional methods and improves the bilateral filter's image quality. A Gaussian lifting framework for bilateral and non-local filtering is provided by Young et al. [16], which appeals to similarities between separable wavelets transform and Gaussian pyramids. The precise implementation of this filter was important not only for image processing applications but also for several recently proposed bilateral regular inverse problems, in which the accuracy of the answer depends entirely on the precise execution of the filter. Gaussian lifting designs are also examined for bilateral and non-local filters.

Reviewing recent studies on image denoising revealed the drawbacks of various methods and techniques applied in this field. Therefore, new image-denoising methods using metaheuristic algorithms have been proposed. Karami and Tafakori [17] proposed a filter for noise removal. To design this filter, they made a mask with a fixed size and used the GC function in it and found that the GC function could be effective in removing the Gaussian noise. By applying this filter to different noisy images, it can be concluded that the mask size has a major effect on the filter performance. In this paper, a mask is designed, and the GC function parameters as well as the mask size are considered as parameters that should be optimized. Most meta-heuristic algorithms applied to Bilateral filters optimize the intensity and spatial domain parameters in the Gaussian function and assume the neighborhood radius to be constant except for Nabahat et al. [18] methods, or Wang et al. [19] method, which claims that the spatial domain parameter has little effect on the filter performance; therefore, It is assumed to be constant and optimizes the neighborhood radius and intensity domain parameter. In this paper, it is claimed that the neighborhood radius, as well as the spatial and intensity domain parameters, significantly affect the bilateral filter's performance. On the other hand, due to the almost similar behavior, and the effective and better performance of the GC function compared to the Gaussian function in noise reduction, the GC function is used in the spatial and intensity domain of the bilateral filter. Here the WOA and GSK algorithms were applied to solve the Non-deterministic polynomial time (NP) problem, which resulted from the anonymity of the exact values of parameters that must be optimized in the bilateral filter and GC filter. The GC filter parameters as well as mask size and the bilateral filter parameters such as the intensity domain, spatial domain, and spatial neighborhood radius were optimized using the WOA and GSK algorithms. The noise-free image was achieved and compared with classical noise removal filters such as Mean filter, Gaussian filter, Median filter, Wiener filter, Non-local mean filter, and three metaheuristic-based algorithms like PSO-based GC filter [17] (GC_PSO), and two PSO-based bilateral filters (Wang's method [19] 'BW_PSO' and Asokan's method [20] 'BA_PSO') on various images respectively.

The rest of the paper is organized as follows. "Preliminaries" includes preliminaries (explain the GC distribution, Bilateral filter, PSO, WOA, and GSK algorithms). "Proposed Method" presents the proposed filters based on the GC function. The experimental results and discussion of the proposed method and its description are explained in "Experimental Results and Discussion", and finally, the conclusions and suggestions for future work are presented in "Conclusion and Future Directions".

Related Works

By far, there have been several methods proposed for image denoising and restoration. This section reveals the recent studies conducted on image-denoising techniques.

Image denoising's main goal is to remove the noise effectively and preserve the original image details as much as possible, and to this end, many approaches have been considered [21].

Dhanushree et al. [22] used different filters to remove speckle noise on acoustic images and found that among the available filters, the bilateral filter; followed by the guided filter, further removes speckle noise from acoustic images.

A hierarchical sequence of development and creation of various Gaussian noise removal methods from the primary methods to more sophisticated hybrid techniques are reviewed by Goyal et al. [23]. Other de-noising techniques have been proposed in [24] and [25]. To train high-quality noise reduction models based on an unorganized group of corrupted images, a method described by Laine et al. [24]. This training eliminates the need for reference images using "blind spot" networks in the receiving field and can, therefore, be used in situations where access to such data is costly or impossible. This method also controls situations where the noise model parameters are variable and unclear in training and evaluation data. The model that adequately selects the regularization parameter in the total variation model was proposed by Pan et al. [25]. In this model, an iterative algorithm was used to estimate the optimal upper bound using the stability between the value of the fitting data term and the upper bound. Then, a dual-based method was applied that avoids calculating the Lagrangian coefficient associated with that constraint, to solve the constrained problem.

Various complex background noise and weak desired signals have severely limited the practical application of Distributed Fiber Optic Acoustic Detection (DAS) as a transformative technology in seismic exploration. A residual encoder-decoder deep neural network (RED-Net) enhanced by deep repetitive memory block (DMB) and channel aggregation block (CAB) called Residual Channel Aggregation Encoder-Decoder Network (RCEN) for vertical seismic profile record (VSP) received by DAS is presented for effective noise removal [26]. DMB uses the theory of weight accumulation to improve the feature extraction ability and achieve accurate noise removal; meanwhile, CAB enhances the performance of weak signal storage using multi-channel analysis architecture.

Modifying the measurement index, defining a constraint function, and considering the collision between readers and between readers and tags led to the development of an improved radio-frequency identification (RFID) reader anti-collision model [27]. Since the number of encoded variables increased because of the dense deployment of many readers and caused a high-dimensional problem that traditional algorithms cannot solve, the Distributed Parallel Cooperative Particle Swarm Optimization (DPCCPSO) is used. Inertia weights and learning factors are adjusted during evolution, an improved clustering strategy is obtained, and various combinations of random number generation functions are tested.

A gated attention mechanism and a linear fusion method construct a two-stream interactive recurrent feature transformation network (IRFR-Net) [28]. First, a context extraction module (CEM) is designed to obtain low-level, depth background information. Second, the gated attention fusion module (GAFM) obtains useful RGB depth information

(RGB-D) structural and spatial fusion features. Third, adjacent depth information is integrated globally to obtain complementary context features. A Weighted Atrous Spatial Pyramid (WASPP) fusion module extracts multi-scale local information of depth features. Finally, the global and local features are combined in a bottom-up scheme to highlight salient objects effectively.

An image segmentation method based on deep learning to segment key regions in mineral images using morphologic transformation to process mineral image masks is presented by Yang et al. [29]. Four aspects of the deep learning mineral image segmentation model are considered: backbone selection, module configuration, building the loss function, and its application in the classification of the mineral image. A new loss function suitable for mineral image segmentation is also presented, and the formation performance of Convolution neural network (CNN) based segmentation models under various loss functions is compared. Hussain and Vanlalruata [30] de-noised the image using CNN to improve character recognition. By classifying the noise types, they identified the kind of noise to select a specific model of de-noising to increase the image de-noising performance. A noisy and corresponding clean image is fed into the network for training. After that, the generated model de-noises the character image. Chaurasiya and Ganotra [31] have changed the receptive field and investigated its effect on image noise removal. For this purpose, they designed and compared the networks: CNN with expanded kernels, CNN without expansion but with increased kernel size with the same receptive field, and CNN without any expansion and without increasing the kernel size. After reviewing the previous three items, they added a fourth item with an optimized receptive field that improves advanced results.

In recent years, the use of meta-heuristic algorithms has received much attention, which plays an essential role in replacing human inspections and interpreting processed images. Meta-heuristic algorithms have demonstrated their effectiveness in solving high-dimensional optimization problems. Using random initial solutions, these algorithms generate optimal solutions for complex optimization problems [32]. They are divided into four categories. Evolution-based algorithms, swarm-based algorithms, physics-based algorithms, and human-related algorithms. Each category has several algorithms, and they have been used in real-world applications in various fields of engineering and science. The evolutionary algorithms that have gained widespread recognition include genetic algorithm (GA) [33] and differential evolution (DE) [34]. The swarm intelligence algorithms that are most commonly used are Artificial Bee Colony (ABC) [35], Firefly Algorithm (FA) [36], Particle Swarm Optimization (PSO) [37], Moth-Flame Optimization (MFO) [38], Salp Swarm Algorithm (SSA) [39], Grey Wolf Optimizer (GWO) [40], and WOA [41]. The simulated annealing algorithm (SA) [42] is a sample of the physics-based algorithm. Harmony Search (HS) [43], Teaching Learning-Based Optimization (TLBO) [44], and Gaining sharing knowledge-based optimization algorithm (GSK) [45] are the well-known human-based metaheuristic algorithms.

Improved Gray Wolf Optimization (IGWO) addresses the limitations of traditional Grey Wolf Optimization (GWO) by incorporating Dimension Learning-Based Hunting (DLH), inspired by wolf pack dynamics. DLH creates personalized neighborhoods for each wolf, allowing them to exchange information and maintain a balance between local and global search [46]. In response to the limitations of the traditional WOA algorithm, which can converge slowly and get stuck in local optima, a variant called Multi-Population Evolutionary Algorithm (MEWOA) was introduced in [47]. MEWOA divides the population into three subpopulations with different searching strategies: one that searches globally and locally, another that explores randomly, and a third that exploits the search space. This approach helps MEWOA find better solutions and avoid local optima more effectively.

Multi-Trials Vector-Based Differential Evolution (MTDE) [48] is a metaheuristic algorithm that combines multiple search algorithms to evolve better solutions. It uses a novel approach called Multi-Trial Vector (MTV), which adaptively adjusts the movement step size based on past successes. MTV incorporates three different Trial Vector Producer (TVP) strategies: Representative-based, Local Random, and Global Best History. These TVPs share their experiences through an archived database, allowing for more effective solution exploration.

Redundant or irrelevant features in datasets can degrade algorithms' performance. Effective feature selection through nature-inspired metaheuristics like the Aquila optimizer can improve accuracy and decision-making. A wrapper feature selection approach uses the Aquila optimizer to identify the most efficient feature subset, which was tested on medical datasets with binary feature selection methods (S-shaped binary Aquila optimizer (SBAO) and V-shaped binary Aquila optimizer (VBAO)) [49].

In [50], a Discrete Propeller-Flame Optimization Algorithm (DMFO-CD) is proposed for community detection in graphs. It adapts Continuous Moth-Flame Optimization (CMFO) for discrete problems by representing solution vectors, initializing, and moving strategy. DMFO-CD uses a locus-based adjacency representation and considers node relationships during initialization without assuming the number of communities. The movement strategy updates solutions with a two-point crossover for computing movements, a single-point neighbor-based mutation for improving exploration and balancing exploitation and exploration, and a single-point crossover based on modularity in the fitness function.

In “Monkey King Evolution” (MKE) [51], the combination of different methods and control parameters affects the convergence rate and balance between exploration and exploitation. By combining multiple strategies, the Multi-Trial Vector-Based Monkey King Evolution (MMKE) algorithm improves global search performance and avoids early convergence. GSK [45], is a novel algorithm that is derived from the concept of acquisition and distribution of knowledge during the human lifetime. Many efforts have been made in different fields with this algorithm. A binary-based GSK algorithm for feature choice was implemented in [52]. Modifications for the GSK algorithm are done in [53] for its performance enhancement. Agrawal et al. [54] use the GSK algorithm for solving stochastic programming problems.

An Adaptive genetic algorithm (AGA) and bilateral filtering [7] were combined to provide a noise reduction and image restoration filter [55]. The results obtained from this technique indicated that it offered better performance in de-noising all types of noisy images with a higher de-noising Peak signal to noise ratio (PSNR) [56], and it restored all images with high quality. Another automatic PSO-based [37] method for the bilateral filter [7] parameter selection was introduced [19]. The Structural similarity index measure (SSIM) [57] was used as a fitness function to optimize the intensity domain and radius parameters by applying the PSO algorithm. The de-noising performance of the bilateral filter was significantly improved in their method, and the low stability of the bilateral filter without parameter optimization was declared. The parameters of the bilateral filter [7] were also optimized using PSO, cuckoo search [58], and adaptive cuckoo search algorithms to reduce the satellite images that have been affected by Gaussian noise [20]. The proposed adaptive cuckoo search method and traditional filters were compared by evaluating the PSNR, Mean squared error (MSE), Feature Similarity Index (FSIM), Entropy, and CPU time. Their method is an edge-preserving filter with low complexity, and is faster than other optimization algorithms. The parameters of the bilateral filter, including the neighborhood radius, which was considered a parameter, were optimized by the WOA algorithm [41], and the obtained image was restored by optimizing the point spread function in the Richardson-Lucy algorithm (R-L) algorithm [18]. The morphological operation and Multi-objective particle swarm optimization (MOPSO) were used to design a de-noising filter [59]. In their approach, first, a series and parallel compound morphology filter were generated based on an open-close (OC) operation, and a structural element with various sizes aiming to remove all noises in a series link was chosen; after that, MOPSO was combined to solve the parameters’ setting of multiple structural elements. While smoothing the noise, the edges and texture details have been preserved in their methods. An APSO-based R-L algorithm was used for blurry elimination and restoration of the de-noised image using a Fuzzy-based median filter (FMF) [60]. They claimed that their FMF and APSO-RL methods have a higher value regarding PSNR and Second derivative like measure enhancement (SDME) than the other conventional filtering and restoration techniques. Singh et al. [61] used fuzzy linguistic quantifiers to remove impulse noise from images. They claimed that, since the median filter determines the median of a predefined mask, sometimes the estimated intensity of the median filter will again cause noise. The performance of the network can be increased by the size of the receptive field in noise removal. Some features of the GC distribution were used, and a mask was designed that reduces the image noise while preserving the edges and details of the image [17]. The parameters of the GC function optimized by the PSO algorithm [37] and MSE [62] value are selected as a fitness function. The result of this paper claims that maximum PSNR value can be achieved and it is an easily designed method.

Spatial filters include some drawbacks; for instance, these filters smooth the data while decreasing noise and blurring edges in the image. Also, linear filters cannot effectively remove signal-dependent noise. Likewise, spatial frequency filtering and wavelet-based algorithms have some defects, including the calculation complexity and time-consuming. By removing these disadvantages, image denoising and consistency efficiency can be enhanced.

Filter-based denoising techniques can effectively reduce the noise, but they cannot preserve the image quality and useful information; so metaheuristic algorithms which play an important role in replacing human inspections and interpretation of processed images, have been used. Due to the novelty of the GSK algorithm, and the lack of wide applications in image processing, the GSK algorithm is used for noise removal purposes. The WOA and GSK algorithms are used to optimize the parameters due to having a high convergence speed. Since the WOA algorithm has two separate steps of exploration and exploitation in almost half of the iterations that prevent the possibility of getting stuck in local optima. The GSK algorithm's scalability ensures that it can effectively balance exploration and exploitation capabilities. Therefore, the current paper aimed to de-noise images through the WOA and GSK algorithms in the bilateral and GSK filters to optimize the filter parameters. Table 1 details the previous similar efforts and the context of motivating the proposed filters.

The advantages of the proposed method include simple design, significant noise removal, and preservation of image information. However, in the proposed methods, the original noiseless image must be accessible for comparison, which is one of the disadvantages of these methods.

The proposed de-noising filters are applied to the images corrupted with Gaussian and salt & pepper (SAP) noises. The results are compared with each other, and traditional

Table 1 Details of recently produced filters

Author	Filter	Metaheuristic algorithm	Function	Parameters	Fitness function	Neighboring radius
Karami and Tafakori [17]	GC	PSO	GC	GC parameters	MSE	Fix = 3
Nabahat et al. [18]	Bilateral	WOA	Gaussian	Domain, range, and radius	PSNR + SSIM	Variable
Asokan and Anitha [20]	Bilateral	PSO, CS, ACS	Gaussian	Domain, range	MSE	Fix = 3
Wang et al. [19]	Bilateral	PSO	Gaussian	Range, and radius	SSIM	Variable
Sakthidasan and Nagappan [55]	Bilateral	AGA	Gaussian	Domain, range	PSNR	Fix = 3
Proposed1	Bilateral	WOA, GSK	GC	GC parameters and radius	PSNR	Variable
Proposed2	Bilateral	WOA, GSK	GC	Domain, range, and radius	PSNR	Variable

methods such as Mean filter, Gaussian filter, Median filter, Wiener filter, Non-local mean filter, PSO-based GC filter [17] (GC_PSO), and two PSO-based bilateral filters (Wang's method [19] 'BW_PSO' and Asokan's method [20] 'BA_PSO') on various images that are corrupted by Gaussian noise.

The SSIM, PSNR, Figure of merit (FOM) [63], Edge Preservative Factor (EPF) [64] values, and execution time are calculated for this comparison, and the proposed filters' efficiency is determined.

Preliminaries

At first, we try to explain the applied functions and algorithms. The explanations about the generalized Cauchy distribution and the bilateral filter are given in "The GC Distribution" and "Bilateral Filter", respectively. Details about PSO, WOA, and GSK algorithms are given in "PSO Algorithm", "WOA Algorithm", and "GSK Algorithm", respectively.

The GC Distribution

The GC distribution is an asymmetric distribution with a bell-shaped density function, similar to the Gaussian distribution, but with a higher mass in the tails and is considered a particular distribution due to the heavy tails. The GC distribution family has properties that depend on the probability density function for the whole family and has algebraic tails that model many impulsive processes in real life [65]. Another parameterization of the GC distribution was performed by Miller and Thomas [66]. Later, the probability density function was given as follows, mainly used to eliminate radio speckle noise [67]. Details about the GC distribution have been described in reference [17].

$$f(x) = \frac{\mu\,\beta\,\Gamma(2/\beta)}{2(\Gamma(1/\beta))^2}\left(\mu^{\beta} + |x-\theta|^{\beta}\right)^{-\frac{2}{\beta}},\ \beta,\mu > 0\,,\ x,\theta \in \mathbb{R}, \tag{1}$$

where β corresponds to the tail constant (causes the sharpness or non-sharpness of the peak point of the curve and moving the peak point of the curve up or down), μ is the scale parameter (causes the tail of the curve to be closer or farther away) and θ refers to the tail of the curve moving from symmetry and $\Gamma(.)$ is the Gamma function.

Bilateral Filter

As mentioned in [20], the bilateral filter, which was proposed by Tomasi [7], has been a non-linear, edge-preserving, and noise-reducing smoothing filter, which is a combination of range and domain filtering and replaces the intensity of each pixel with the weighted Mean intensity of adjacent pixels. The weights are based on the Gaussian distribution, replaced by the GC distribution in the proposed method. According to the bilateral filter definition, the noisy image is filtered from the following formula:

$$I^{\text{filtered}} = \frac{1}{W}\sum_{x_i \in N} I(x_i) f_{\text{r}}\left(I(x_i) - I(x)\right) g_{\text{d}}\left(x_i - x\right). \tag{2}$$

The weight W is defined so that adjacent pixels within a neighborhood are compared with the central pixel, and the higher weights are assigned to pixels that are more similar and closer to the center pixel.

$$W = \sum_{x_i \in N} f_{\text{r}}\left(I(x_i) - I(x)\right) g_{\text{d}}\left(x_i - x\right), \tag{3}$$

where I^{filtered} and I represent the filtered and noisy images, respectively.x is the current pixel coordinate that needs to be filtered. N is the window centered in, x so $x_i \in N$ is another pixel. f_{r} and g_{d} are the range and domain kernel for smoothing the differences in intensities and coordinates.

PSO Algorithm

PSO [37] is a social search algorithm inspired by the social behavior of birds. The algorithm is based on particles representing a potential solution to the optimization problem. The algorithm aims to find the particle location in the response space that obtains the best value for the objective function. Each particle is considered a possible solution to the problem. The improvement in the solution provided by each particle comes from two sources; the first is using the particle's personal experience, called the cognitive component (pbest). The other is to improve the answer in the particle community, which is called the social component (gbest). pbest is the best solution that the particle has received so far from the implementation of the algorithm and gbest is the best solution experienced in the population so far from the implementation of the algorithm. To calculate the velocity of each particle in each location, pbest and gbest are used simultaneously. The cognitive and social components are combined to guide the particle to a better solution to define the particle velocity. The particle velocity in each iteration of the algorithm is calculated as follows.

$$v_i(t+1) = \omega\, v_i(t) + c_1 r_1 \left(\text{pbest}_i(t) - x_i(t)\right) + c_2 r_2 \left(\text{gbest}(t) - x_i(t)\right), \tag{4}$$

where $x_i(t)$ and $v_i(t)$ display the current particle location and current velocity, respectively, and $v_i(t+1)$ indicates the particle's new velocity to move from the current location to the new location. ω is the inertia weight, c_1 and c_2 are the acceleration constant, r_1 and r_2 are the random values in the range (0,1). The new location of each particle is obtained from the following equation.

$$x_i(t+1) = x_i(t) + v_i(t+1). \tag{5}$$

The weight w changes with the number of iterations and can be calculated according to Eq. (6) [68]

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{\text{iter}_{\max}} \times \text{iter}_{\text{current}}. \tag{6}$$

Moreover $w_{\min}$, and $w_{\max}$ are the minimum and maximum weights, respectively, $\text{iter}_{\text{current}}$ and $\text{iter}_{\max}$ indicate the current and maximum iterations.

WOA Algorithm

One nature-inspired algorithm that uses the humpback whale hunting strategy is the whale algorithm [41]. The humpback whales usually go 10–15 m underwater and form spiral bubbles to encircle the prey. Afterward, it moves towards the surface of the water and the prey. This type of whale behavior involves two phases exploration and exploitation.

The WOA starts with initializing the search agents (whales) and each search agent's position $Y_i, i = 1, 2, \ldots, n$, which n indicates the number of search agents. After initialization, the fitness function was evaluated for each search agent, and the best value among them was considered Y^*. Since the exact location of the prey in the search space is unknown, the best answer Y^* is to consider the location of the prey or close to it. The rest of the search agents update their position according to this answer; obtained from the following equations.

$$\vec{R} = \left|\vec{C} \cdot \vec{Y}^*(u) - \vec{Y}(u)\right|, \tag{7}$$

$$\vec{Y}^*(u+1) = \vec{Y}^*(u) - \vec{A} \cdot \vec{R}, \tag{8}$$

where u, $\vec{Y}^*$, $\vec{Y}$ signifies the current iteration, the position vector of the current best solution, and the position vector respectively, "| |" is the absolute value, "." is elementwise multiplication, and the vectors $\vec{A}$ and $\vec{C}$ are obtained from Eqs. (9) and (10). It should be noted that if there is a better solution, Y^* should be updated.

$$\vec{A} = 2\vec{m} \cdot \vec{n} - \vec{m}, \tag{9}$$

$$\vec{C} = 2\vec{n}. \tag{10}$$

In the exploration and exploitation phases, $\vec{n}$ is a random value in $[0, 1][0, 1]$, and $\vec{m}$ decreases from 2 to 0 during the iterations.

Two mechanisms of whale bubble network attack are mathematically modeled as follows:

- The shrinking surrounding mechanism is accomplished by reducing the value of $\vec{m}$ in Eq. (9); correspondingly, the amount of $\overleftarrow{A}$ also decreased.
- The spiral updating position mechanism in which whales imitate the helix-shaped movement to update the position between the prey and the whale is described in Eq. (11).

$$\vec{Y}(u+1) = \vec{R'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{Y^*}(u), \tag{11}$$

where $\vec{R'}$, l and b are the distance between the prey and ith whale (current optimal solution $\overline{Y^*}$), a random value in the range of $[-1, 1]$ and a constant that corresponds to the logarithmic shape of the helix, respectively.

The humpback whales can use both mechanisms simultaneously. Given the same probability of both mechanisms, the mathematical model is as follows.

$$\vec{Y}(u+1) = \begin{cases} \vec{Y}^*(u) - \vec{A} \cdot \vec{R} & p < 0.5 \\ \vec{R'} \cdot e^{bl} \cdot \cos(2\pi l) + \vec{Y}^*(u) & p \geq 0.5 \end{cases}, \tag{12}$$

where p is a random number in [0, 1].

Changes in $\vec{A}$ values are considered the exploration phase. In this phase, the humpback whales search arbitrarily according to the location of each one. The arbitrary values are in the range [− 1, 1], forcing the whale to travel far away from the reference whale. In the exploration phase, the position of the whales is updated according to the randomly selected whale.

$$\vec{R} = \left|\vec{C} \cdot \vec{Y}_{\text{rand}}(u) - \vec{Y}(u)\right|, \tag{13}$$

$$\vec{Y}(u+1) = \vec{Y}_{\text{rand}} - \vec{A} \cdot \vec{R}. \tag{14}$$

The WOA algorithm starts to find the best solution by arbitrarily tracing the whales in the search space. The whales update their location in each iteration according to the best or arbitrarily selected search agent. The value p reveals that the whales should have a spiral or shrinkage movement. The WOA algorithm ends when a predetermined termination condition is met.

GSK Algorithm

Gaining sharing knowledge-based optimization algorithm (GSK) [45] is a newly developed metaheuristic algorithm that follows the concept of gaining and sharing knowledge throughout the human lifetime. Let $\{y_1, y_2, \ldots, y_M\}$ be the individuals of the population size M. Each individual y_j is defined as $y_j = [x_{j1}, x_{j2}, \ldots, x_{jc}]$ where c is the branch of knowledge assigned to an individual. In each iteration, individuals are sorted in ascending order according to the value of the objective function and then use the junior gaining and sharing phase and the senior gaining and sharing phase to update the population of individuals together.

Junior GSK Phase

Each y_j gains knowledge from the two closest individuals, y_{j-1} (the best one) and y_{j+1} (the worst one). It also shares the knowledge of an individual y_{rand} randomly. The individuals are updated through Eq. (15).

$$y_j^{\text{new}} = \begin{cases} y_j + k_f.\left[(y_{j-1} - y_{j+1}) + (y_{\text{rand}} - y_j)\right], & \text{if} f(y_{\text{rand}}) < f(y_j) \\ y_j + k_f.\left[(y_{j-1} - y_{j+1}) + (y_j - y_{\text{rand}})\right], & \text{if} f(y_{\text{rand}}) \geq f(y_j) \end{cases}, \tag{15}$$

where y_j^{new} is a trial vector for y_j, f and k_f are the objective function value and knowledge factor, respectively.

Senior GSK Phase

After sorting individuals into ascending order (based on the objective function values) in this phase, the individuals are classified into three categories (best, middle, and worst). The best and worst levels each contain n $\times$ M ($n \in [0, 1]$) individuals, and the middle level has the rest $(1 - 2n) \times$ M individuals. For each individual y_j, it gains knowledge from three individuals of different groups using Eq. (16):

$$y_j^{\text{new}} = \begin{cases} y_j + k_f \cdot \left[(y_{\text{pb}} - y_{\text{pw}}) + (y_{\text{m}} - y_j)\right], & \text{if} f(y_{\text{m}}) < f(y_j) \\ y_j + k_f \cdot \left[(y_{\text{pb}} - y_{\text{pw}}) + (y_j - y_{\text{m}})\right], & \text{if} f(y_{\text{m}}) \geq f(y_j) \end{cases}, \tag{16}$$

where y_{pb}, y_{pw} and y_{m} are random individuals selected from the best, middle, and worst levels, respectively.

Both phases are done to update the different dimensions of an individual. Note that the numbers of dimensions that will be updated using the junior phase and the senior phase are calculated by the following formulation, respectively:

$$c_{ja} = \left(1 - \frac{u}{u_{\max}}\right)^k \times c, \tag{17}$$

$$c_{se} = c - c_{ja} \tag{18}$$

where $k > 0$, u, and $u_{\max}$ are a knowledge rate, the current iteration, and the maximum number of iterations, respectively. In Algorithms 2 and 4, $k_r \in [0, 1]$ the knowledge ratio controls the total amount of gained and shared knowledge that will be inherited during generations (the ratio between the current and acquired experience).

Proposed Method

Designing an effective filter that preserves the edges and structural information of the image is one of the challenges most researchers face in image processing. The primary purpose of this paper is to develop two automatic filters to reduce the noise using the GC distribution. The definition of the first and second proposed filters is explained in "Mask Design Using the GC Function" and "Bilateral Filter Using the GC Function (BL-GC)", respectively.

Mask Design Using the GC Function

A mask is designed to produce a noiseless image convolved with the noisy image to create an efficient filter. For this purpose, the parameters of the GC function β, μ, θ and mask size w are considered as parameters, and the optimal values of these parameters are found by maximizing the fitness function Eq. (24) defined in "Fitness Function" in the WOA and GSK algorithms. Karami and Tafakori [17] used some features of the GC function and designed a mask that reduces image noise. They optimized the GC distribution

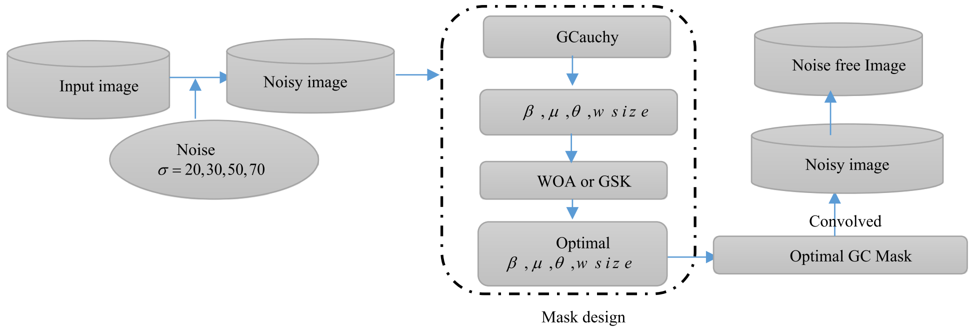

Fig. 1 Diagram of the first proposed method

parameters by considering the MSE [62] as a fitness function in the PSO [37]. Their method needed to recalculate the PSNR of the filtered image at the end of the algorithm. The selected mask size was considered constant and equal to 3, while in our proposed method, the mask size is regarded as a parameter that should be optimized. By repeating and examining their method on different images with different mask sizes, we found that the selected mask size significantly impacts filter performance, so the proposed method addresses these issues. The diagram of the first proposed method is shown in Fig. 1.

Considering that I_{input} is an $m \times n$ ordered noise-free grayscale image corrupted by an additive Noise, I_{noisy} (noisy image) is obtained.

$$I_{\text{noisy}} = I_{\text{input}} + \text{noise}. \tag{19}$$

The noisy image is convolved with the designed mask F, and the noise-free image is obtained.

$$I_{\text{output}} = I_{\text{noisy}} \times F. \tag{20}$$

To design the mask 'F', the bivariate GC function, an extension of the univariate function Eq. (1), is considered Eq. (21).

$$f(x,y) = \left(\frac{\mu \beta \Gamma(2/\beta)}{2(\Gamma(1/\beta))^2} \right)^2 (\mu^\beta + |x-\theta|^\beta)^{-\frac{2}{\beta}} (\mu^\beta + |y-\theta|^\beta)^{-\frac{2}{\beta}}, \quad \beta, \mu > 0, \, x, y, \theta \in \mathbb{R}. \tag{21}$$

A discretization must be performed to store the continuous generalized Cauchy function in the form of discrete pixels. This process is done, and the mask is produced. The designed mask size has an odd value like 3, 5, etc., which the optimal value is computed through the whale algorithm. For example, the 5×5 mask with $\beta = 1, \mu = 1$, $\theta = 0$ is as follows:

0.0042	0.0094	0.0375	0.0094	0.0042
0.0094	0.0211	0.0843	0.0211	0.0094
0.0375	0.0843	0.3371	0.0843	0.0375
0.0094	0.0211	0.0843	0.0211	0.0094
0.0042	0.0094	0.0375	0.0094	0.0042

In the WOA (and GSK), each search agent Y_i, $i = 1, \ldots, n$ (each individual $y_j, j = 1, \ldots, M$) has four parameters β, μ, θ, $w$size that the fitness function must optimize. At first, the number of n search agents (M individuals) containing four parameters β, μ, θ, $w$size is randomly initialized according to the range of parameters defined in "WOA Algorithm"". Different masks are generated according to these search agents (individuals) and convolved with the noisy image, so different noiseless images are obtained. The fitness function Eq. (24) is evaluated for these output images, and the maximum value is considered the best solution. The WOA (GSK) is continued according to pseudo-codes of Algorithms 1 or 2. Completing the predetermined number of iterations results in a noise-free image with the maximum fitness function. The pseudo-codes of the first proposed filter using the WOA and GSK will be as algorithms 1 and 2.

Algorithm 1 The pseudo-code of the first proposed WOA-based filter

```
Enter the original image
Corrupt the original image by different levels of the Gaussian and the SAP noise.
Initialize n search agents Y_i randomly, which each contain 4 parameters β,μ,θ,wsize .
Convolve the masks obtained in pervious step on the noisy image.
u=1
Calculate the fitness function for each of the filtered images, using Eq. (24) and consider Y* as the best solution.
while u ≤ max_u
  for all search agents
    Update m, A, C, p, l
    if p < 0.5
      if |A| < 1
        Update search agents location through Eq. (8)
      elseif |A| ≥ 1
          Choose a random search agent. Y_rand
          Update search agents location through Eq. (14)
      end if
    elseif p ≥ 0.5
      Update search agents location through Eq. (11)
    end if
  end for
    If the search agent is out of the explored area, turn it back
    Evaluate Eq. (24) for each search agent
 Update Y* if there is a better solution.
    u=u+1
end while
    return Y*
    The de-noised image with the maximum fitness function and optimal parameter values are obtained.
```

Algorithm 2 The pseudo-code of the first proposed GSK-based filter

Enter the original image
Corrupt the original image by different levels of the Gaussian noise.
Create an initial random population $y_j, j = 1,\ldots,M$, which each contains 4 parameters $\beta,\mu,\theta,wsize$.
Convolve the masks obtained in pervious step on the noisy image.
Evaluate the fitness for each individual using Eq. (24)
Initialize the iteration counter u= 1
While $u \le \max\_u$
Sort the individuals in the population by their fitness values in ascending order.
Compute the number of dimensions c_{ja} and c_{se} of the junior and senior steps applying Eqs. (17) and (18), respectively
for j = 1 to M do
for l = 1 to c do
if $rand(0,1) < k_r$ then
Junior GSK step
if $rand(0,1) < \frac{c_{ja}}{c}$ do
Create y^{new}_{jl} applying Eq. (15)
else
Senior GSK step
Create y^{new}_{jl} applying Eq. (16)
end if
else
$y^{new}_{jl} = y_{jl}$
end if
end for
end for
Calculate Eq. (24) for each sample vector
When the sample vector is superior to the objective individual, accept it
u = u+ 1
end while
The de-noised image with the maximum fitness function and optimal parameter values are obtained.

Bilateral Filter Using the GC Function (BL-GC)

In this section, a filter is designed to reduce the noise using a bilateral filter, in which the GC function is used instead of the Gaussian function.

Assume that the pixel in position (i,j) must be de-noised using the adjacent pixels, and one of the adjacent pixels is in position (k,l); in this case, the weight assigned to the pixel (k,l) for noise reduction of the pixel (i,j) is as follows:

$$w(i,j,k,l) = (\mu_d^{\beta_d} + |(\sqrt{(i-k)^2 + (j-l)^2}) - \theta_d|^{\beta_d})^{-\frac{2}{\beta_d}} (\mu_r^{\beta_r} + ||I(i,j) - I(k,l)| - \theta_r|^{\beta_r})^{-\frac{2}{\beta_r}}, \tag{22}$$

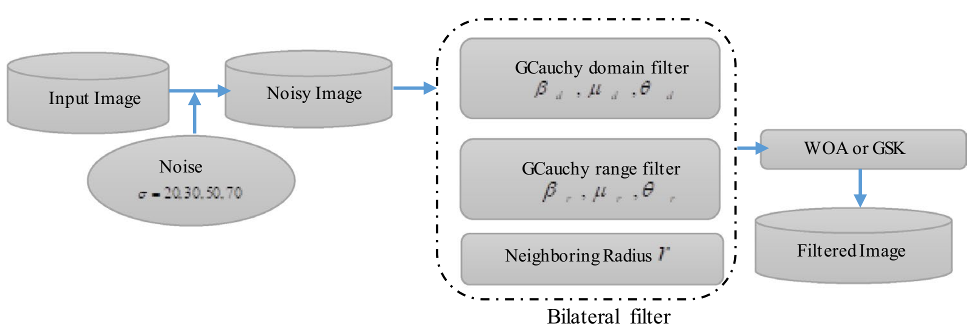

Fig. 2 Diagram of the second proposed method

where β_d, μ_d, θ_d are the smoothing parameters in the spatial domain and β_r, μ_r, θ_r are the smoothing parameters in the range domain, and $I(i,j), I(k,l)$ are the intensity of the corresponding pixels. The proposed filter output is calculated as follows:

$$I_D(i,j) = \frac{\sum_{k,l} I(k,l)\ w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}. \tag{23}$$

In Eq. (23), the fraction's denominator is the normalization factor and I_D is filtered pixel intensity at the location (i,j). Since the optimal values of the smoothing parameters require experimental and manual efforts, the WOA (and GSK) is used to obtain the optimal values of these parameters by considering Eq. (24) in "PSO Algorithm" as a fitness function. In the proposed filter, the neighboring radius r is also considered a parameter that should be optimized, and correspondingly, the window size ($w\text{size} = 2r + 1,\ r = 1, 2, \ldots$) is obtained. The diagram of the second proposed method is shown in Fig. 2

In WOA (or GSK), each search agent (each individual) has parameters β_d, μ_d, θ_d, β_r, μ_r, θ_r, r that achieve the optimum value according to the fitness function Eq. (24).

Suppose that I_{input} is a $m \times n$ noise-free grayscale image corrupted by additive noise and I_{noisy} obtained. Equation (23) is applied to all pixels of the noisy image, and the noiseless image is obtained. The noisy image is entered into the WOA (GSK), each search agent (each individual) which contains parameters "β_d, μ_d, θ_d, β_r, μ_r, θ_r, r" is initialized randomly in the range defined in "WOA Algorithm", and according to these parameters, Eq. (23) is applied to all pixels of the noisy image, and different noiseless images are obtained. The fitness of the output images is calculated through Eq. (24), the maximum value is considered the best solution, and the WOA (or GSK) is continued to complete a predetermined number of iterations. A noise-free image with a maximum fitness function is obtained when the algorithm terminates. The pseudo-code of the second proposed method using WOA and GSK will be as Algorithm 3 and 4.

Algorithm 3 The pseudo-code of the second proposed WOA-based filter

```
Enter the original image
Corrupt the original image by different levels of the Gaussian and the SAP noise.
Initialize n search agents Y_i randomly, which each contain parameters β_d, μ_d, θ_d, β_r, μ_r, θ_r, r .
Apply Eq. (8) to all of the noisy image pixels, according to the parameters that have been initialized.
Different de-noised images are obtained.
u=1
Calculate the fitness function for each of the filtered images, using Eq. (24) and consider Y* as the best solution.
while u ≤ max_u
  for all search agents
    Update m, A, C, p, l
    if p < 0.5
      if |A| < 1
        Update search agents location through Eq. (8)
      elseif |A| ≥ 1
          Choose a random search agent. Y_rnd
          Update search agents location through Eq. (14)
      end if
    elseif p ≥ 0.5
      Update search agents location through Eq. (11)
    end if
  end for
    If the search agent is out of the explored area, turn it back
    Evaluate Eq. (24) for each search agent
     Update Y* if there is a better solution.
    Iter=Iter+1
end while
    return Y*
    The de-noised image with the maximum fitness function and optimal parameter values are obtained.
```

Algorithm 4 The pseudo-code of the second proposed GSK-based filter

```
Enter the original image
Corrupt the original image by different levels of the Gaussian noise.
Create an initial random population y_j , j = 1,...,M , which contains 7 parameters β_d, μ_d, θ_d, β_r, μ_r, θ_r, r .
Apply Eq. (8) to all of the noisy image pixels, according to the parameters that have been initialized.
Different de-noised images are obtained
Evaluate the fitness for each individual using Eq. (24)
Initialize the iteration counter u = 1
While u ≤ max_u
    Sort the individuals in the population by their fitness values in ascending order.
    Compute the number of dimensions c_ja and c_se of the junior and senior steps applying Eqs. (17) and (18), respectively
    for j = 1 to M do
        for l = 1 to c do
            if rand(0,1) < k_r then
             Junior GSK step
                if rand(0,1) < c_ja / c do
                 Create y_new_jl applying Eq. (15)
                 else
                  Senior GSK step
                 Create y_new_jl applying Eq. (16)
                end if
            else
             y_new_jl = y_jl
        end if
      end for
    end for
  Calculate Eq. (24) for each sample vector
  When the sample vector is superior to the objective individual, accept it
   u = u+ 1
end while
The de-noised image with the maximum fitness function and optimal parameter values are obtained.
```

Fitness Function

The goal of optimization problems is to find the optimal solution to the problem. The primary purpose of this paper is to achieve the best results in the proposed filters. The search agents must get better in each iteration. These search agents must be evaluated according to the fitness function in each iteration. The desired fitness function for the first and second proposed filters is the *PSNR*, which is computed using Eq. (24) [62].

$$\mathrm{PSNR} = 10\log_{10}\frac{255^2}{\mathrm{MSE}}, \tag{24}$$

where *MSE* is the mean squared error calculated from Eq. (25) [62], I and I_D are $M \times N$ ordered original and de-noised images, respectively.

$$\mathrm{MSE} = \frac{1}{M \times N}\sum_{x=1}^{M}\sum_{y=1}^{N}\left(I(x,y) - I_D(x,y)\right)^2. \tag{25}$$

A higher *PSNR* value indicates a further improvement in the filtered image.

Parameter Setting

Since all meta-heuristic algorithms are parameter-based. Therefore, the analysis of these parameters plays a significant role in determining the optimal solution. According to the parameter values used in meta-heuristic algorithms to solve different types of problems in [41, 45, 46], and also the test results of meta-heuristics-based noise reduction filters on different images, the optimal values of the parameters can be considered as follows.

Table 2 The SSIM results for Gaussian noise reduction

	Noisy	Average	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA	P1_GSK	P2_GSK
Barbara													
$\sigma = 20$	0.9384	0.9570	0.9682	0.9541	0.9773	**0.9806**	0.9723	0.9774	0.9774	0.9723	0.9784	0.9723	0.9787
$\sigma = 30$	0.8739	0.9480	0.9378	0.9399	0.9562	0.9566	0.9573	0.9638	0.9612	0.9586	0.9646	0.9585	**0.9662**
$\sigma = 50$	0.7234	0.9196	0.8552	0.8992	0.9142	0.8444	0.9239	0.9420	0.9265	0.9368	**0.9443**	0.9366	**0.9443**
$\sigma = 70$	0.5843	0.8789	0.7599	0.8462	0.8582	0.7063	0.8814	0.9204	0.8826	0.9138	0.9229	0.9136	**0.9231**
Boats													
$\sigma = 20$	0.9178	0.9701	0.9614	0.9701	0.9778	0.9765	0.9755	0.9793	0.9793	0.9757	0.9797	0.9757	**0.9807**
$\sigma = 30$	0.8333	0.9577	0.9203	0.9506	0.9572	0.9470	0.9606	0.9663	0.9640	0.964	0.9695	0.9638	**0.9697**
$\sigma = 50$	0.6474	0.9190	0.8111	0.8953	0.8979	0.7899	0.9203	0.9404	0.9225	0.9399	0.9442	0.9397	**0.9443**
$\sigma = 70$	0.4950	0.8663	0.6948	0.8245	0.8291	0.6236	0.8671	0.9098	0.8638	0.9120	**0.9183**	0.9115	0.9171
Hill													
$\sigma = 20$	0.9253	0.9767	0.9658	0.9754	0.9804	0.9790	0.9801	0.9824	0.9826	0.9812	0.9839	0.9812	**0.9841**
$\sigma = 30$	0.8483	0.9657	0.9290	0.9584	0.9627	0.9532	0.9676	0.9750	0.9705	0.9730	0.9767	0.9731	**0.9768**
$\sigma = 50$	0.6759	0.9318	0.8310	0.9093	0.9116	0.8127	0.9327	0.9591	0.9355	0.9562	**0.9628**	0.9561	0.9626
$\sigma = 70$	0.5264	0.8835	0.7216	0.8448	0.8484	0.6583	0.8840	0.9379	0.8827	0.9337	0.9423	0.9336	**0.9428**
Couple													
$\sigma = 20$	0.9096	0.9649	0.9572	0.9648	0.9732	0.9727	0.9719	0.9751	**0.9755**	0.9719	**0.9755**	0.9719	0.9751
$\sigma = 30$	0.8179	0.9507	0.9117	0.9430	0.9500	0.9391	0.9548	0.9579	0.9582	0.9580	0.9621	0.958	**0.9622**
$\sigma = 50$	0.6221	0.9077	0.7931	0.8821	0.8850	0.7675	0.9096	0.9294	0.9124	0.9303	**0.9344**	0.93	**0.9344**
$\sigma = 70$	0.4664	0.8501	0.6692	0.8046	0.8111	0.5929	0.8512	0.8963	0.8485	0.8989	**0.9060**	0.8985	0.9037
Peppers													
$\sigma = 20$	0.9355	0.9682	0.9688	0.9784	0.9841	0.9818	0.9772	0.9857	0.9848	0.9773	0.9859	0.9773	**0.9869**
$\sigma = 30$	0.8671	0.9584	0.9363	0.9625	0.9570	0.9590	0.9639	0.9738	0.9704	0.9662	0.9764	0.9662	**0.9768**
$\sigma = 50$	0.7096	0.9282	0.8490	0.9173	0.9211	0.8361	0.9310	0.9452	0.9322	0.9436	0.9481	0.9436	**0.9501**
$\sigma = 70$	0.5763	0.8834	0.7491	0.8556	0.8605	0.6897	0.8850	0.9128	0.8827	0.9146	**0.9190**	0.9144	0.9189
House													
$\sigma = 20$	0.9149	0.9704	0.9605	0.9756	0.9803	0.9784	0.9761	0.9843	0.9827	0.9768	0.9857	0.977	**0.9858**
$\sigma = 30$	0.8284	0.9580	0.9184	0.9552	0.9600	0.9498	0.9612	0.9745	0.9687	0.9659	0.9763	0.9659	**0.9769**
$\sigma = 50$	0.6443	0.9207	0.8103	0.8973	0.9024	0.7909	0.9223	0.9539	0.9293	0.9439	0.9588	0.9444	**0.9589**
$\sigma = 70$	0.4938	0.8661	0.6942	0.8206	0.8317	0.6297	0.8671	0.9284	0.8697	0.9145	0.9347	0.9142	**0.9348**

Table 3 The PSNR results for Gaussian noise reduction

	Noisy	Average	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA	P1_GSK	P2_GSK
Barbara													
$\sigma = 20$	22.1719	24.1800	25.3115	23.8346	26.8623	**27.4603**	26.0062	26.9048	26.8703	26.0062	27.1251	26.006	27.1968
$\sigma = 30$	18.7903	23.3517	22.2932	22.6125	24.2574	23.8654	24.1580	24.956	24.5593	24.3498	25.0645	24.3605	**25.2578**
$\sigma = 50$	14.7543	21.5275	18.4235	20.1850	21.0986	17.9096	21.7443	23.0918	21.8692	22.7465	23.2571	22.7542	**23.3042**
$\sigma = 70$	12.3739	19.8597	16.0238	18.1097	18.9707	14.5886	19.9321	21.9397	19.8922	21.6707	22.0612	21.6786	**22.133**
Boats													
$\sigma = 20$	22.1789	27.0588	25.7513	26.9698	28.2744	27.9604	27.8698	28.5909	28.6106	27.9572	28.7337	27.9593	**28.9527**
$\sigma = 30$	18.741	25.5377	22.4583	24.7202	25.3931	24.3345	25.8282	26.6108	26.2146	26.2993	27.0309	26.3095	**27.058**
$\sigma = 50$	14.5894	22.745	18.3616	21.2302	21.5641	17.7155	22.8076	24.2803	22.9051	24.287	24.6381	24.3083	**24.6576**
$\sigma = 70$	12.1793	20.6256	15.9212	18.6715	19.3102	14.3822	20.6456	22.7568	20.4342	22.8973	**23.196**	22.919	23.1483
Hill													
$\sigma = 20$	22.1675	27.6232	25.8009	27.3318	28.3183	27.9863	28.2908	28.8102	28.8612	28.5627	29.249	28.59	**29.3074**
$\sigma = 30$	18.7561	25.9444	22.5049	24.9866	25.5041	24.4328	26.1781	27.3898	26.5771	27.0754	27.6857	27.089	**27.6982**
$\sigma = 50$	14.6764	23.0156	18.4608	21.3913	21.7282	17.8846	23.0669	25.4657	23.2105	25.1941	**25.8468**	25.2012	25.8315
$\sigma = 70$	12.2768	20.7836	16.0109	18.7919	19.4048	14.5605	20.7988	23.8126	20.6413	23.6728	24.1618	23.6821	**24.2649**
Couple													
$\sigma = 20$	22.1761	26.8282	25.7409	26.723	27.9258	27.7532	27.7644	28.2573	28.3247	27.7981	**28.3378**	27.7982	**28.3378**
$\sigma = 30$	18.7221	25.3453	22.4315	24.5596	25.1767	24.1651	25.6916	26.1009	26.0236	26.0875	26.5411	26.1217	**26.568**
$\sigma = 50$	14.5558	22.6115	18.3237	21.1496	21.4657	17.6222	22.6881	23.9933	22.7938	24.0951	24.3880	24.1376	**24.3903**
$\sigma = 70$	12.1335	20.5481	15.8747	18.6052	19.2702	14.3169	20.5707	22.6097	20.3734	22.7805	**23.0441**	22.8006	23.0372
Peppers													
$\sigma = 20$	22.2057	25.7355	25.6167	27.2629	27.9597	28.6288	27.1008	29.1406	28.8121	27.1593	29.2222	27.169	**29.5272**
$\sigma = 30$	18.7812	24.5598	22.4066	24.8198	24.6465	24.3613	25.1302	26.5736	25.9484	25.4925	27.0218	25.4931	**27.0914**
$\sigma = 50$	14.7065	22.2378	18.4244	21.2186	21.6552	17.8508	22.3859	23.5301	22.4364	23.4346	23.7697	23.4896	**23.9262**
$\sigma = 70$	12.3214	20.2227	15.9881	18.5792	19.2280	14.5022	20.2678	21.704	20.0937	21.9154	22.1373	21.9366	**22.1422**
House													
$\sigma = 20$	22.1284	27.1579	25.7423	27.954	28.8745	28.4083	28.0577	29.9409	29.4784	28.2282	30.3377	28.2509	**30.3866**
$\sigma = 30$	18.6774	25.6046	22.4181	25.2382	25.7535	24.65	25.936	27.8805	26.8841	26.5715	28.2072	26.5974	**28.297**
$\sigma = 50$	14.5834	22.8336	18.3727	21.3961	21.7798	17.7718	22.9101	25.4213	23.3086	24.5653	25.974	24.6109	**25.9817**
$\sigma = 70$	12.221	20.6231	15.9525	18.6171	19.3958	14.5343	20.6497	23.7784	20.6422	22.9725	**24.2212**	23.0041	24.2193

The WOA parameters follow the values given in "WOA Algorithm", and the PSO parameters are considered $c_1, c_2 = 2$, $w_{\max} = 0.9$, $w_{\min} = 0.4$. The GSK parameters " $M = 20$, $u_{\max} = 50, k = 10, n = 0.1, k_f = 0.5, k_r = 0.9$ " are as mentioned in "Experimental Results and Discussion". In the first proposed filter using WOA (or GSK) and the GC_PSO filter, the maximum number of iterations and the population size are 300 and 50, respectively. The GC function parameters are concluded from [17], so the mask size and GC parameters are set as follows.

$$\beta \in (0,5]\ ,\ \mu \in (0,2]\ ,\ \theta \in [-1,1]\ ,\ w\text{size} \in [3,11]. \tag{26}$$

The WOA (or GSK) obtains the optimal GC function parameters and mask size values. As shown in Table 4, the optimal mask size increases by increasing the Gaussian noise standard deviation. In the second proposed method, BW_PSO; and BA_PSO filters [19, 20], the population size and the maximum number of iterations are considered 20 and 50, respectively. The domain and range parameters of the GC function in the bilateral filter and the neighboring radius have been experimentally obtained by trying on different images and set as follows:

$$\beta_d \in (0,1]\ ,\ \mu_d \in [100,230]\ ,\ \theta_d \in [-0.5,0.5]\ ,$$
$$\beta_r \in [0.01,50]\ ,\ \mu_r \in [20,150]\ ,\ \theta_r \in [-70,70]\ , r \in [1,5]. \tag{27}$$

The domain and range parameters and the radius parameter in the BW_PSO filter [19] are considered as follows:

$$\sigma_d = 10\ ,\ \sigma_r \in [1,200]\ , d \in [1,5]. \tag{28}$$

The domain and range parameters in the BA_PSO filter [20] are considered as follows:

$$\sigma_d = [0.1,10]\ ,\ \sigma_r \in [1,200]\ . \tag{29}$$

Given the stochastic nature of the proposed methods and PSO-based filters, the presented results are an average of 20 times, the execution of these algorithms.

Experimental Results and Discussion

To demonstrate the efficiency of the proposed filters using WOA and GSK, they are compared with each other, five classical filters, and GC_PSO, BW_PSO, and BA_PSO filters. To this end, six grayscale images such as Barbara "512 × 512", Boats "512 × 512", Hill "512 × 512", Couple "512 × 512", Peppers "256 × 256", and House "256 × 256" are considered as data sets. These are noiseless test images with a resolution of 8 bits per pixel and are taken from https://www.kaggle.com/datasets/saeedehkamjoo/standard-test-images. To evaluate the performance of the proposed methods in the presence of noise, each image is corrupted with different standard deviations of Gaussian noise and various densities of SAP noise. These images are corrupted with four standard deviations "$\sigma = 20, 30, 50, 70$" of the Gaussian noise and three different densities "$d = 0.02$, 0.03, 0.05" of SAP noise. Later for all levels of Gaussian noise standard deviation, the classical filters such as Mean filter, Gaussian filter, Median filter, Wiener filter, non-local mean filter (NLM), the GC_PSO filter, BW_PSO, BA_PSO, WOA-based P1_WOA filter (P1_WOA), GSK-based P1_WOA filter (P1_GSK), WOA-based P2_WOA filter (P2_WOA), and GSK-based P2_WOA (P2_GSK) filters are applied to the noisy images, for all densities of SAP noise, Mean filter, Gaussian filter, Median filter, Wiener filter, non-local mean filter, GC_PSO filter, BW_PSO filter, BA_PSO filter, P1_WOA filter, and P2_WOA filter are evaluated. The SSIM [26] value between the filtered image and the original image and the SSIM value between the original image and the noisy image are calculated and are listed in Table 2. The SSIM [57] value is calculated from Eq. (30). Note that the SSIM values between the filtered and the original noiseless images are calculated manually in all methods except for the BW_PSO, which is automatically calculated by the PSO algorithm. In the GC_PSO filter and the BA_PSO, the MSE value between the filtered and original images is calculated automatically, but the SSIM value between the filtered and original images is calculated manually. In the proposed filters, the PSNR value is calculated automatically by WOA (or GSK). To make a fair comparison between the noise reduction filters in all images, the PSNR value between the filtered and the original images is also calculated manually. The results are listed in Table 3.

The SSIM [57] is a number between zero and one and evaluates the similarity between two images; in contrast, brightness, and structure. The primary purpose is to assess this criterion between the original noiseless image and the filtered output image. Higher values indicate that the structural similarity of the resulting image is close to the original noiseless image, and the filter performance is better.

$$\text{SSIM} = \frac{\left(2\mu_I\mu_{I_n} + c_1\right)\left(2\sigma_{I,I_n} + c_2\right)}{\left(\mu_I^2 + \mu_{I_n}^2 + c_1\right)\left(\sigma_I^2 + \sigma_{I_n}^2 + c_2\right)}, \tag{30}$$

where, μ_I, μ_{I_n} and $\sigma_I^2, \sigma_{I_n}^2$ are the mean value and variance value of corresponding original and noisy images, and σ_{I,I_n} is the covariance between original and noisy image. $c_1 = (0.01 \times 255)^2, c_2 = (0.03 \times 255)^2$ are two constants.

As an example, the results of the calculations are shown on some images in Fig. 3, in which the images "Hill, Couple, Barbara and Boats" are corrupted with standard deviations 20, 30, 50 and 70 of the Gaussian noise, respectively. In

Fig. 3, "a, b, c, d, e, f, g, h, i, j, k, l, m and n" represent the Original, Noisy, Mean filter, Gaussian filter, Median filter, Wiener filter, NLM filter, GC_PSO filter, BW_PSO filter, BA_PSO filter, P1_WOA filter, P2_WOA filter, P1_GSK, and P2_GSK filter images respectively.

As shown in Tables 2 and 3, higher values are marked in bold, and the P2_GSK filter in almost all images and all standard deviations of the Gaussian noise have better SSIM and PSNR than other methods. After the P2_GSK filter, the P2_WOA filter performs better than other filters. BW_PSO filter has better SSIM and PSNR than others, even compared to the P1_WOA and P1_GSK. The comparison between methods shows that the metaheuristic-based methods have almost better fitness function and SSIM values. The fitness and SSIM value of the proposed filters will be even better than the GC_PSO method. The P2_WOA and P2_GSK filters act better than the P1_WOA and P1_GSK filters. BW_PSO filter acts better than the BA_PSO, P1_WOA, and P1_GSK filters. However, the proposed methods have better results than the classical filters and GC_PSO, which shows the proposed methods' superiority.

After implementing the P1_WOA method and GC_PSO, the corresponding optimal GC distribution parameters and window size values are obtained for each Gaussian noise standard deviation level. The results are placed in Table 4. By plotting the GC function diagram for the optimal parameters obtained from the WOA algorithm and increasing the image noise, the shape of the GC function will be close to the Gaussian function, which reduces undesirable noise effects. Figure 4 shows an example of the GC distribution shape concerning the parameters obtained from the WOA for different noise levels in the Hill image to explain the above claim. Figure 4; a, b, c, and d are the GC distribution shape concerning the optimal parameters for $\sigma = 20, 30, 50, 70$ the Gaussian noise in the Hill image, respectively.

Table 4 shows the PSNR value of the P1_WOA filter and GC_PSO filter for all Gaussian noise standard deviations. All images practically depend on the optimal choice of filter parameters and the selected mask size. For $\sigma = 20$ both methods have almost similar results in all images, and even for some images, the P1_WOA method provides more desirable results. By increasing the Gaussian noise standard deviation, the P1_WOA filter has better PSNR results than the GC_PSO filter. So, the role of the selected mask size in the filter efficiency can be better understood

To assess whether the proposed methods preserve the edges of the image or not, the figure of merit (FOM) [63] and Edge Preservative Factor (EPF) [64] are evaluated. The FOM is a method for quantitative comparison between edge detection algorithms in image processing and has a value between zero and one. The closer the importance of this criterion is to one, the better it shows the edge values and is formulated in Eq. (31) [63].

$$R = \frac{1}{\mathrm{Max}\left(N_1, N_2\right)} \sum_{i=1}^{N_2} \frac{1}{1 + Cd^2(i)}. \tag{31}$$

Furthermore N_1, N_2 representing the number of actual edges and detected edges achieved by the Sobel edge detector, C represents a constant value equal to 1/9, $d(i)$ representing the distance between the actual edge and the detected edge.

The EPF is a measure that computes the details preservation ability of the filtered image and is computed from Eq. (32) [64].

$$\mathrm{EPF} = \frac{\sum (I_L - \mu_{I_L}) \times (DI_L - \mu_{\mathrm{DI}_L})}{\sqrt{\sum (I_L - \mu_{I_L})^2 \times \sum (DI_L - \mu_{\mathrm{DI}_L})^2}}, \tag{32}$$

where I_L and DI_L are the Laplacian operators of the original and filtered image, respectively, μ_{I_L} and μ_{DI_L} are the corresponding mean values. The higher EPF value indicates that the filtered image has more details. The FOM and EPF values are calculated for all filtered images and placed in Tables 5 and 6, respectively.

A filter with a higher PSNR, SSIM, FOM, and EPF value and less computational complexity and computational time is efficient. The proposed filter's computational time is described in "Bilateral Filter Using the GC Function (BL-GC)". It is claimed that all de-noising filters produced by meta-heuristic algorithms are convergent, whose convergence is reviewed in "Fitness Function".

According to Table 5, the highest FOM values are marked in bold, and the FOM value for the P2_GSK filter compared to other filters in most images and almost all Gaussian noise standard deviation levels has the highest value. After P2_GSK, the P2_WOA filter has better performance than the others. In some images, for some levels of Gaussian noise standard deviation, the BW_PSO filter performed better.

As shown in Table 6, the highest EPF values of the filters are marked in bold and vary in different images and different standard deviations of Gaussian noise.

To accurately compare the performance of filters with different images and for different levels of Gaussian noise standard deviation, in terms of criteria like PSNR, SSIM, FOM, and EPF, Friedman's algorithm is used, which is mentioned in "Mask Design Using the GC Function".

It is clear from Fig. 3 that the images "l" and "n" obtained by P2_WOA and P2_GSK filters respectively; for all standard deviations of Gaussian noise perform better than other images in noise reduction.

The exact process applies to all images with SAP noise. At first, all images are corrupted with a density of 0.02, 0.03, and 0.05 SAP noise. The noisy images are de-noised with the abovementioned filters, and the noiseless images have

Fig. 3 Images before and after Gaussian noise removal

Table 4 Optimal parameter values for different images

Method	PSNR	σ	β	μ	θ	wsize
Barbara (512 × 512)						
GC_PSO	26.0062	20	1	1.081	− 0.0036	3
P1_WOA	26.0062	20	0.71387	2	− 0.0037	3
GC_PSO	24.1580	30	0.9714	1.9997	0.0087	3
P1_WOA	**24.3498**	30	4.1724	0.5747	0.0002	7
GC_PSO	21.7443	50	1.4756	1.9993	0.0096	3
P1_WOA	**22.7465**	50	1.6732	1.0356	0.0121	9
GC_PSO	19.9321	70	1.7865	1.9992	0.0287	3
P1_WOA	**21.6707**	70	1.2793	1.7659	0.0191	9
Boats (512 × 512)						
GC_PSO	27.8698	20	1.261	1.4027	0.0245	3
P1_WOA	**27.9572**	20	4.9417	0.5944	0.0083	5
GC_PSO	25.8282	30	1.4281	1.9885	0.0142	3
P1_WOA	**26.2993**	30	4.0744	0.7681	0.0074	5
GC_PSO	22.8076	50	2.1409	1.9977	− 0.0047	3
P1_WOA	**24.287**	50	2.4962	1.0993	0.0089	7
GC_PSO	20.6456	70	2.4775	2	0.0138	3
P1_WOA	**22.8973**	70	2.4648	1.2722	− 0.0018	9
Hill (512 × 512)						
GC_PSO	28.2908	20	2	1.0014	0.0749	3
P1_WOA	**28.5627**	20	1.6464	0.8056	0.1122	5
GC_PSO	26.1781	30	1.5026	2	0.0872	3
P1_WOA	**27.0754**	30	1.5081	1.0385	0.1304	7
GC_PSO	23.0669	50	2.2173	1.9792	0.0800	3
P1_WOA	**25.1941**	50	1.7451	1.3417	0.1725	9
GC_PSO	20.7988	70	2.6376	2	0.0578	3
P1_WOA	**23.6728**	70	2.1347	1.5234	0.1505	9
Couple (512 × 512)						
GC_PSO	27.7644	20	1.2806	1.2919	− 0.0308	3
P1_WOA	**27.7981**	20	4.9917	0.5898	− 0.0202	5
GC_PSO	25.6916	30	1.3674	1.9569	− 0.0346	3
P1_WOA	**26.0875**	30	4.9221	0.7265	− 0.0148	5
GC_PSO	22.6881	50	2.0281	2	− 0.0353	3
P1_WOA	**24.0951**	50	2.1412	1.1273	− 0.0347	7
GC_PSO	20.5707	70	2.2755	2	− 0.0218	3
P1_WOA	**22.7805**	70	1.8349	1.4029	− 0.0493	9
Peppers (256 × 256)						
GC_PSO	27.1008	20	0.9968	1.2048	− 0.1314	3
P1_WOA	**27.1593**	20	4.9713	0.5349	− 0.0872	5
GC_PSO	25.1302	30	1.9996	1.0643	− 0.1240	3
P1_WOA	**25.4925**	30	2.1202	0.7869	− 0.1159	5
GC_PSO	22.3859	50	1.9999	1.6973	− 0.1660	3
P1_WOA	**23.4346**	30	1.2866	1.5553	− 0.2009	5
GC_PSO	20.2678	70	2.4408	1.6512	− 0.1108	3
P1_WOA	**21.9154**	70	2.1939	1.2401	− 0.1495	7
House (256 × 256)						
GC_PSO	28.0577	20	2.8144	0.7981	0.0792	3
P1_WOA	**28.2282**	20	1.8432	0.7200	0.1003	5
GC_PSO	25.936	30	1.8416	1.4047	0.1176	3
P1_WOA	**26.5715**	30	1.7616	1.0008	0.1399	5
GC_PSO	22.9101	50	2.1653	1.9064	0.1179	3

Table 4 (continued)

Method	PSNR	σ	β	μ	θ	$w$size
P1_WOA	**24.5653**	50	0.7789	1.8633	0.2983	7
GC_PSO	20.6497	70	2.3981	1.9761	0.1194	3
P1_WOA	**22.9725**	70	1.9132	1.5903	0.1444	7

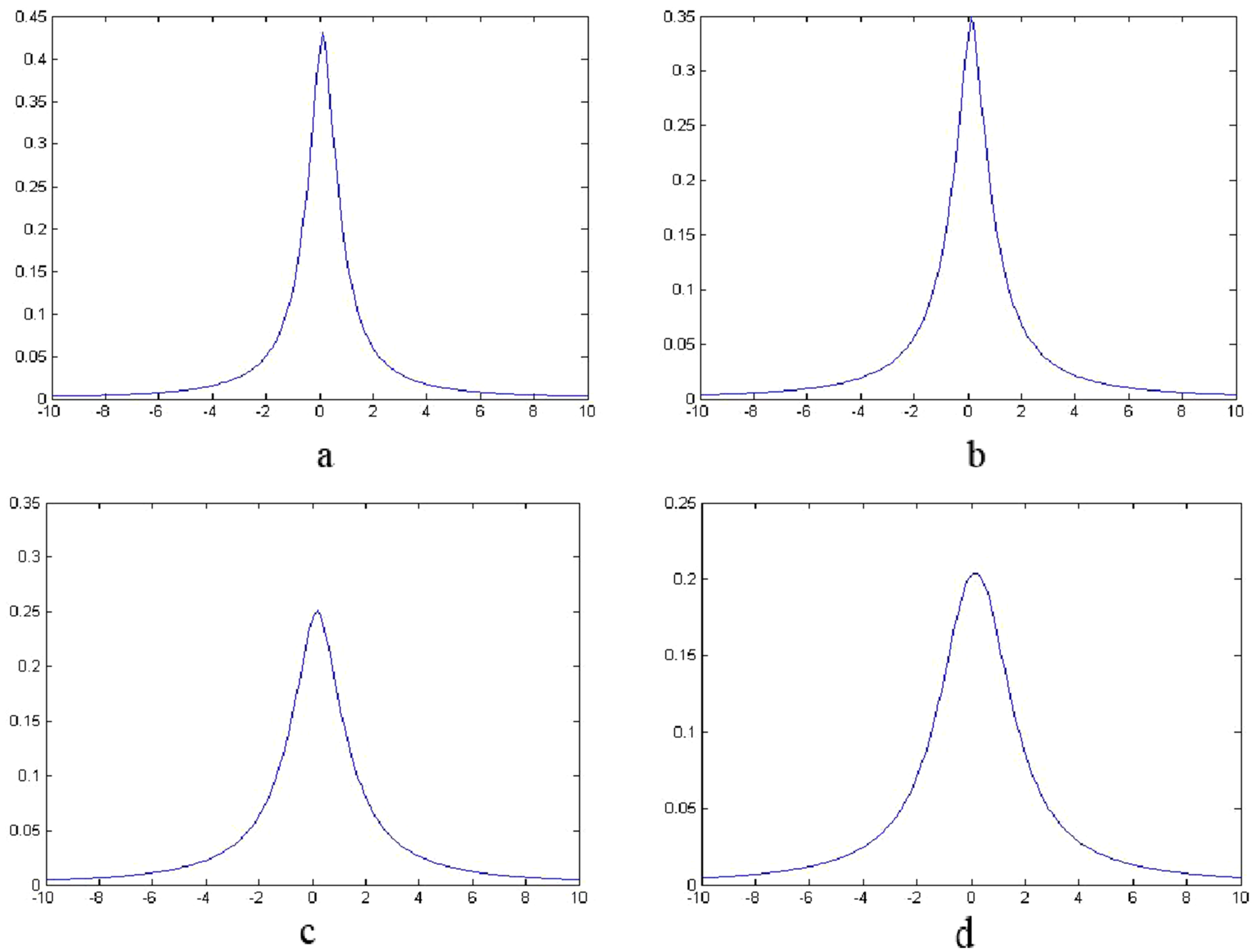

Fig. 4 Cauchy distribution diagram for optimal parameters for the Hill image

been obtained. Since the SAP noise affects some pixels of the image, it should not be directly applied to the designed filters. To create a unique procedure with Gaussian-noised images, the proposed filters are also applied to all the pixels of the image contaminated with SAP noise. The PSNR, SSIM, FOM, and EPF values are calculated to compare noise reduction filters.

As shown in Tables 7, 8, 9, 10, according to all criteria, the median filter and the P2_WOA filter have better results in some images and some noise densities. To check the performance of filters in terms of PSNR, SSIM, FOM, and EPF in the presence of SAP noise, Friedman's algorithm explained in "Mask Design Using the GC Function" is used, the results of which are placed in Table 13.

Figure 5 displays the practical results in the House, Peppers, and Couple images, which have been corrupted with SAP noise densities of 0.02, 0.03, and 0.05, respectively. In Fig. 5, "a, b, c, d, e, f, g, h, i, j, k, and l" are the images represented by "Original," "Noisy," "Mean filter," "Gaussian filter," "Median filter," "Wiener filter," "NLM filter," "GC_PSO filter," "BW_PSO filter," "BA_PSO filter," "P1_WOA filter," and "P2_WOA filter" respectively. As shown in Fig. 5, the median and P2_WOA filters not only remove the noise but also maintain image quality, unlike the other methods that cause a loss of image quality. Therefore, it can

Table 5 The FOM results for Gaussian noise reduction

	Noisy	Mean	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA	P1_GSK	P2_GSK
Barbara													
$\sigma = 20$	0.4288	0.6918	0.4692	0.6086	0.8009	0.7154	0.5027	0.9195	0.6443	0.5028	0.9118	0.623	**0.9609**
$\sigma = 30$	0.3818	0.4847	0.4062	0.4732	0.7066	0.6131	0.4388	0.7682	0.4665	0.4537	0.7483	0.4602	**0.8278**
$\sigma = 50$	0.3312	0.4023	0.3646	0.4113	0.4310	0.3970	0.3972	0.5239	0.3962	0.4154	0.5147	0.4262	**0.5761**
$\sigma = 70$	0.298	0.3768	0.3338	0.3822	0.3981	0.3380	0.3744	0.4438	0.3782	0.4082	0.4674	0.4112	**0.495**
Boats													
$\sigma = 20$	0.3610	0.7341	0.4205	0.5914	0.7514	0.6703	0.5619	0.6859	0.6786	0.6288	0.8717	0.6384	**0.9714**
$\sigma = 30$	0.3196	0.4726	0.3455	0.4321	0.47	0.5591	0.429	**0.7999**	0.4465	0.4936	0.7784	0.5051	0.7737
$\sigma = 50$	0.2719	0.3535	0.3017	0.3586	0.3611	0.3294	0.345	0.4334	0.3491	0.4165	0.5135	0.4197	**0.5352**
$\sigma = 70$	0.2426	0.3186	0.2727	0.3257	0.3244	0.2739	0.3188	0.39	0.3171	0.3818	0.4076	0.3865	**0.412**
Hill													
$\sigma = 20$	0.3986	0.5521	0.4326	0.528	0.5908	0.5768	0.4937	0.544	0.5277	0.5327	0.7477	0.5295	**0.8317**
$\sigma = 30$	0.3651	0.4428	0.3873	0.4416	0.467	**0.558**	0.4259	0.5531	0.431	0.4709	0.5448	0.485	0.556
$\sigma = 50$	0.3159	0.3842	0.3501	0.3961	0.4033	0.3772	0.3833	0.4368	0.3818	0.4229	0.4566	0.421	**0.4644**
$\sigma = 70$	0.2854	0.3628	0.3211	0.3684	0.3777	0.3218	0.3603	0.4036	0.3643	0.3963	0.4104	0.4064	**0.4297**
Couple													
$\sigma = 20$	0.3026	0.7931	0.3643	0.6172	0.7724	0.7406	0.564	0.7475	0.6798	0.516	**0.8723**	0.6672	0.7638
$\sigma = 30$	0.2528	0.4378	0.2804	0.369	0.4121	0.5127	0.3762	**0.8321**	0.3865	0.4601	0.7745	0.4712	0.7971
$\sigma = 50$	0.2129	0.2871	0.2354	0.2831	0.2905	0.2583	0.2822	0.372	0.2826	0.3469	0.4491	0.3486	**0.4527**
$\sigma = 70$	0.1878	0.2531	0.2126	0.2543	0.2528	0.2137	0.2509	0.3087	0.2475	0.3081	**0.327**	0.3121	**0.3622**
Peppers													
$\sigma = 20$	0.2568	0.7759	0.3094	0.5809	0.8129	0.6449	0.5381	0.9095	0.7391	0.5249	0.911	0.5465	**0.9161**
$\sigma = 30$	0.2189	0.3912	0.2394	0.3264	0.8096	0.4313	0.3637	**0.853**	0.3383	0.3922	0.825	0.3952	0.8348
$\sigma = 50$	0.1861	0.2509	0.2056	0.2485	0.2520	0.2239	0.2412	0.3759	0.2451	0.288	**0.3896**	0.2995	0.3448
$\sigma = 70$	0.1646	0.2243	0.1850	0.2187	0.2267	0.1852	0.2229	0.2672	0.2223	0.2707	0.3065	0.2837	**0.324**
House													
$\sigma = 20$	0.3141	0.8769	0.4399	0.9409	0.9344	0.9326	0.8582	0.9227	**0.9655**	0.8828	0.9237	0.8848	0.9262
$\sigma = 30$	0.2442	0.6489	0.2781	0.4566	0.5238	0.6205	0.5069	0.9193	0.6029	0.7612	0.9154	0.7781	**0.9204**
$\sigma = 50$	0.2009	0.2926	0.2265	0.2812	0.2828	0.2451	0.2821	0.6842	0.2844	0.5689	**0.8206**	0.6107	0.8169
$\sigma = 70$	0.1762	0.2495	0.1994	0.2389	0.2461	0.2013	0.2485	0.4166	0.2449	0.3912	**0.5438**	0.4113	0.5421

be concluded that the P2_WOA filter performs better than the other filters after the median filter.

To investigate the effect of the proposed filters on real-world problems, we not only examined their impact on standard images corrupted with Gaussian noise or SAP noise but also considered a medical image (Brains MRI with dimensions of 454 × 448) corrupted by various standard deviations of Gaussian noise. We applied the mentioned filters to the image and calculated criteria such as PSNR, SSIM, FOM, and EPF. The results were then recorded in Table 11.

In Table 11, the highest values of the criteria are marked in bold. To determine which filter performs better for all standard deviations of Gaussian noise, Friedman's method is applied for each criterion and the results are listed in Table 14.

Statistical Analysis

Two non-parametric statistical hypothesis tests are utilized to examine the quality and performance of algorithms, such as the Friedman test and the multi-problem Wilcoxon signed-rank test [69]. The null assumption represents no meaningful divergence between the proficiency of the methods, and the alternative assumption is the opposite of the null assumption. According to the obtained p-value, it is decided to reject or accept the null assumption. If the p-value exceeds 0.05, the null assumption is accepted; otherwise, it is rejected.

For all standard deviations of Gaussian noise, the mean rank of the de-noising filters is obtained in different images using the Friedman test regarding the PSNR, SSIM, FOM, and EPF. Columns 2 to 5 and 8 to 12 of Tables 12 and 13 list the mean rank of the filters for all standard deviations of Gaussian noise according to the Friedman test. The 6th and

Table 6 The EPF results for Gaussian noise reduction

	Mean	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA	P1_GSK	P2_GSK
Barbara												
$\sigma = 20$	0.2042	0.5825	0.2042	0.7014	**0.7609**	0.623	0.6772	0.7027	0.623	0.7028	0.5028	0.7113
$\sigma = 30$	0.1447	0.4218	0.1294	0.5093	**0.5693**	0.4755	0.5121	0.5302	0.4443	0.538	0.439	0.5608
$\sigma = 50$	0.0872	0.2572	0.0658	0.2908	0.2769	0.2359	0.2973	0.2525	0.255	0.346	0.2437	**0.3468**
$\sigma = 70$	0.0619	0.1811	0.047	0.183	0.1733	0.1316	0.1885	0.1601	0.16	0.229	0.1525	**0.2325**
Boats												
$\sigma = 20$	0.1932	0.3812	0.3019	0.5088	0.5364	0.4517	0.534	0.5424	0.4639	0.5009	0.467	**0.5569**
$\sigma = 30$	0.1405	0.2576	0.193	0.3442	0.3224	0.2902	0.3505	0.3451	0.3661	0.4374	**0.3676**	0.4345
$\sigma = 50$	0.084	0.1507	0.0935	0.1737	0.1371	0.1291	0.1512	0.153	0.2519	0.2492	**0.2552**	0.2405
$\sigma = 70$	0.0554	0.1074	0.0546	0.1063	0.0949	0.0758	0.1048	0.1134	0.1882	0.1558	**0.189**	0.1255
Hill												
$\sigma = 20$	0.1038	0.3066	0.1397	0.3344	0.3797	0.3359	0.3258	0.3501	0.3595	**0.4076**	0.3741	0.4043
$\sigma = 30$	0.0736	0.2068	0.0792	0.2113	0.2252	0.1853	0.2056	0.1673	0.2745	0.2754	**0.2788**	0.2761
$\sigma = 50$	0.0475	0.1261	0.0392	0.1151	0.1065	0.078	0.1303	0.0904	0.1759	0.1806	**0.1844**	0.1714
$\sigma = 70$	0.0325	0.0916	0.0228	0.0773	0.0768	0.0456	0.0936	0.0743	0.1258	0.097	**0.1291**	0.0813
Couple												
$\sigma = 20$	0.2259	0.4083	0.2733	0.4911	0.5514	0.5211	0.5169	**0.5535**	0.4601	0.5366	0.5159	0.5274
$\sigma = 30$	0.1617	0.2727	0.1719	0.3179	0.3155	0.3527	0.3064	0.3466	0.4158	0.3931	**0.4175**	0.4008
$\sigma = 50$	0.0955	0.1546	0.0868	0.1503	0.1237	0.1533	0.1174	0.155	0.2869	0.235	**0.2964**	0.2354
$\sigma = 70$	0.0715	0.1044	0.0593	0.0962	0.0819	0.1015	0.0853	0.1198	0.2191	0.1429	**0.2283**	0.1483
Peppers												
$\sigma = 20$	0.1588	0.4552	0.6026	0.7131	0.6692	0.4534	0.7172	0.735	0.5241	0.7222	0.5304	**0.7576**
$\sigma = 30$	0.1057	0.3003	0.4443	0.4817	0.4192	0.3065	0.5161	0.5208	0.388	0.6060	0.3931	**0.6178**
$\sigma = 50$	0.0531	0.1609	0.2398	0.2627	0.162	0.1336	0.1617	0.1373	0.1966	0.2229	0.2018	**0.2704**
$\sigma = 70$	0.0324	0.1018	**0.1485**	0.1336	0.0988	0.0709	0.0625	0.0756	0.1222	0.1342	0.1174	0.116
House												
$\sigma = 20$	0.201	0.2835	0.2945	0.4148	0.4396	0.4018	0.4401	0.4305	0.3753	0.5008	0.3927	**0.5119**
$\sigma = 30$	0.1365	0.1857	0.1755	0.2572	0.2307	0.2601	0.2687	0.214	0.2871	0.3527	0.3057	**0.3785**
$\sigma = 50$	0.0817	0.1108	0.0821	0.1321	0.0945	0.1191	0.0982	0.1238	0.1887	**0.2383**	0.2088	0.238
$\sigma = 70$	0.0584	0.0833	0.0499	0.0926	0.076	0.0778	0.1101	0.0957	**0.1653**	0.1495	0.162	0.1479

12th column of Tables 12 and 13 shows the overall mean rank of the filters obtained by Friedman's test, and the 7th and last columns deal with the ranking of the filters. The p-value computed through the Friedman test is zero and less than 0.05. Thus, we can conclude that there is a significant difference between the performances of the algorithms.

According to the 7th column of the first and second parts of Table 12, P2_GSK, P2_WOA, and BW_PSO filters have the first to third rank, respectively, in terms of SSIM and PSNR. P1_WOA and P1_GSK filters have the fourth rank regarding SSIM and PSNR, respectively. The following ranks are assigned to other filters; the Gaussian filter has the lowest rank. By paying attention to filters with lower rankings, it is understandable that the Gaussian, NLM, and median filters are not appropriate for eliminating Gaussian noise on average. As the first part of Table 12, the last column shows, the P2_GSK filter has the highest Mean ranking in terms of FOM, and P2_WOA, BW_PSO, P1_GSK, and P1_WOA filters have the second to fifth rank, respectively. The BA_PSO and GC_PSO filters have lower rankings on average and perform poorly in terms of FOM. In the last columns of the second section of Table 12, the P2_GSK filter has the highest EPF ranking, and the P2_WOA, P1_GSK, and P1_WOA filters have the second to fourth ranking. Generally, it can be said that the P2_GSK filter has the highest ranking concerning all of the measures, and after that, the P2_WOA filter has a better performance. The BW_PSO filter has the third rank in terms of PSNR, SSIM, and FOM and the fourth rank in terms of EPF. Considering all criteria, filters P2_GSK, P2_WOA, BW_PSO, P1_GSK, P1_WOA, Wiener, BA_PSO, GC_PSO, "Mean = NLM," Median, and Gaussian, respectively have the best performances. It is clear that the GC_PSO filter has a weaker performance than all the proposed methods, and the P2_GSK and P2_WOA filters have a better performance than all other filters. Generally, it can be said that filters based on evolutionary algorithms

Table 7 The PSNR results for SAP noise reduction

	Mean	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA
PSNR										
Barbara										
0.02	24.1193	25.1864	25.3117	22.9355	26.5057	25.8689	25.1009	25.7000	25.8689	**29.9373**
0.03	23.7673	23.7986	25.2456	21.8626	24.6001	24.9739	24.7226	24.9799	24.9833	**28.2636**
0.05	23.0115	21.6403	25.1260	20.3852	21.4305	23.6501	23.9549	23.9637	23.9009	**25.9869**
Boats										
0.02	27.0469	25.9150	30.7923	23.6432	26.6561	27.8775	28.3796	28.3972	27.9274	**31.4941**
0.03	26.3754	24.3395	**30.6731**	22.5537	24.8696	26.9035	27.6960	27.6961	27.0924	30.3135
0.05	25.2046	22.1310	**30.3880**	21.1679	21.9530	25.4459	26.3748	26.3759	25.9076	28.5384
Hill										
0.02	27.6547	26.0340	31.5010	23.4676	26.4968	28.3613	29.1271	29.1322	28.5948	**31.5356**
0.03	26.7858	24.2056	**31.4269**	22.1621	24.2773	27.1851	28.2189	28.2189	27.6271	30.6612
0.05	25.4883	22.0147	**31.2026**	20.8303	21.4725	25.6803	27.2680	26.8312	26.5744	28.5363
Couple										
0.02	26.8534	25.9347	30.4245	23.5685	26.7286	27.8295	28.1511	28.2654	27.8297	**32.4309**
0.03	26.1936	24.3176	**30.3386**	22.4916	24.8207	26.7974	27.4607	27.4615	26.9384	30.2547
0.05	25.1290	22.1916	**30.1385**	21.2319	22.0983	25.4390	26.3360	26.3361	25.8598	28.4323
Peppers										
0.02	25.6504	25.5635	31.9137	23.7168	26.5977	27.0038	27.6115	27.6114	27.0038	**32.3402**
0.03	25.2042	24.2555	**31.5953**	22.7513	24.8785	26.1636	26.9488	26.9486	26.2005	30.0984
0.05	24.2244	21.9294	**30.9877**	21.0299	21.7250	24.6937	25.6319	25.5290	25.0119	27.9545
House										
0.02	27.1020	25.9202	**33.6791**	23.8168	27.4759	28.0657	30.0053	30.0052	28.2066	33.2476
0.03	26.3571	24.1749	**33.4412**	22.4982	24.6628	26.9567	28.9272	28.9278	27.2347	31.1523
0.05	25.3683	22.3009	**33.2887**	21.3790	22.3642	25.6976	27.9390	27.5635	26.2431	29.3716

Table 8 The SSIM results for SAP noise reduction

	Mean	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA
SSIM										
Barbara										
0.02	0.9556	0.9667	0.9671	0.9444	**0.9758**	0.9709	0.9649	0.9697	0.9709	**0.9888**
0.03	0.9517	0.9544	0.9666	0.9292	0.9627	0.9640	0.9617	0.9640	0.9637	**0.9836**
0.05	0.9417	0.9256	0.9657	0.9007	0.9242	0.9503	0.9540	0.9540	0.9521	**0.9718**
Boats										
0.02	0.9696	0.9622	0.9875	0.9370	0.9680	0.9752	0.9778	0.9779	0.9752	**0.9894**
0.03	0.9644	0.9462	**0.9872**	0.9197	0.9522	0.9687	0.9740	0.9740	0.9697	0.9860
0.05	0.9530	0.9119	**0.9863**	0.8905	0.9090	0.9557	0.9646	0.9645	0.9594	0.9789
Hill										
0.02	0.9765	0.9671	0.9905	0.9414	0.9702	0.9801	0.9834	0.9834	0.9810	**0.9906**
0.03	0.9711	0.9501	**0.9904**	0.9215	0.9508	0.9737	0.9795	0.9795	0.9760	0.9885
0.05	0.9606	0.9184	**0.9899**	0.8943	0.9087	0.9625	0.9739	0.9716	0.9686	0.9811
Couple										
0.02	0.9646	0.9585	0.9849	0.9290	0.9651	0.9721	0.9740	0.9747	0.9721	**0.9906**
0.03	0.9587	0.9403	**0.9846**	0.9099	0.9464	0.9644	0.9695	0.9695	0.9651	0.9843
0.05	0.9468	0.9045	**0.9839**	0.8810	0.9027	0.9508	0.9603	0.9603	0.9541	0.9760
Peppers										
0.02	0.9671	0.9679	0.9926	0.9516	0.9749	0.9763	0.9793	0.9793	0.9763	**0.9933**
0.03	0.9634	0.9570	**0.9920**	0.9399	0.9631	0.9711	0.9759	0.9759	0.9709	0.9886
0.05	0.9536	0.9271	**0.9908**	0.9110	0.9251	0.9588	0.9672	0.9662	0.9611	0.9813
House										
0.02	0.9696	0.9614	**0.9935**	0.9383	0.9730	0.9758	0.9844	0.9844	0.9763	0.9927
0.03	0.9638	0.9429	**0.9931**	0.9173	0.9492	0.9686	0.9800	0.9800	0.9702	0.9883
0.05	0.9542	0.9135	**0.9928**	0.8941	0.9158	0.9577	0.9744	0.9726	0.9621	0.9824

Table 9 The FOM results for SAP noise reduction

	Mean	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA
FOM										
Barbara										
0.02	0.7164	0.9112	0.9090	0.8170	0.9027	0.8711	0.8597	0.8934	0.8728	**0.9269**
0.03	0.6447	0.8292	0.9050	0.8195	0.8961	0.7696	0.8349	0.8004	0.8002	**0.9183**
0.05	0.5343	0.6807	**0.9091**	0.7923	0.8297	0.6042	0.6663	0.6458	0.6339	0.8842
Boats										
0.02	0.6702	0.7487	0.9255	0.8276	0.8288	0.6725	0.7880	0.7418	0.6912	**0.9277**
0.03	0.5873	0.6681	**0.9311**	0.7490	0.8452	0.5868	0.6978	0.6977	0.6204	0.9306
0.05	0.4813	0.5441	**0.9288**	0.6266	0.7663	0.4827	0.5864	0.5739	0.5239	0.9187
Hill										
0.02	0.6278	**0.9081**	0.8974	0.7970	0.7345	0.7021	0.7002	0.6942	0.6900	0.8854
0.03	0.5669	0.8066	**0.8988**	0.8113	0.7512	0.5974	0.6251	0.6250	0.5991	0.8743
0.05	0.4947	0.6697	**0.8985**	0.7640	0.8033	0.5017	0.6203	0.5495	0.5249	0.8847
Couple										
0.02	0.6827	0.6229	0.9062	0.6662	0.8862	0.6164	0.7697	0.6750	0.6164	**0.9480**
0.03	0.5794	0.5383	0.9067	0.5853	0.8229	0.5264	0.6804	0.6742	0.5734	**0.9180**
0.05	0.4400	0.4296	**0.9065**	0.4889	0.6270	0.4184	0.5352	0.5352	0.4908	0.9094
Peppers										
0.02	0.5857	0.6106	0.9553	0.6786	0.8438	0.5543	0.7056	0.7056	0.5543	**0.9704**
0.03	0.4947	0.5239	0.9539	0.5963	0.7374	0.4802	0.6335	0.6335	0.5067	**0.9544**
0.05	0.3857	0.4127	**0.9525**	0.4758	0.5777	0.3726	0.5136	0.4817	0.4065	0.8664
House										
0.02	0.7152	0.6005	**0.9723**	0.6168	0.8627	0.6131	0.8228	0.8228	0.6397	**0.9521**
0.03	0.6661	0.5188	**0.9695**	0.5548	0.7418	0.5270	0.7669	0.7675	0.6386	**0.9394**
0.05	0.5429	0.4190	**0.9681**	0.4755	0.6013	0.4504	**0.8482**	0.6537	0.6327	0.8419

Table 10 The EPF results for SAP noise reduction

	Mean	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA
EPF										
Barbara										
0.02	0.2171	0.6544	0.3945	0.4841	0.8009	0.6411	0.3453	0.5560	0.6411	**0.8540**
0.03	0.1827	0.5774	0.3894	0.3906	0.7209	0.5739	0.3024	0.4716	0.5405	**0.7944**
0.05	0.1327	0.4694	0.3781	0.2703	**0.5816**	0.4516	0.2444	0.2698	0.4360	0.5312
Boats										
0.02	0.1971	0.4114	0.5822	0.2657	0.4651	0.4438	0.3010	0.3533	0.4500	**0.6774**
0.03	0.1690	0.3540	**0.5731**	0.2197	0.3882	0.3826	0.2755	0.2760	0.4101	0.5133
0.05	0.1317	0.2672	**0.5485**	0.1571	0.2731	0.2655	0.2174	0.2180	0.3383	0.3612
Hill										
0.02	0.1150	0.3957	0.4171	0.1442	0.3321	0.3748	0.1541	0.1687	0.4098	**0.4351**
0.03	0.0943	0.3239	0.4120	0.1180	0.2534	0.2660	0.1269	0.1270	0.3277	**0.4534**
0.05	0.0714	0.2578	**0.3970**	0.1068	0.1912	0.1705	0.0670	0.1091	0.2863	0.1335
Couple										
0.02	0.2335	0.4679	0.5510	0.2531	0.5012	0.5318	0.3191	0.4453	0.5320	**0.7549**
0.03	0.2059	0.3948	0.5430	0.2040	0.4041	0.4606	0.2839	0.2937	0.4811	**0.5604**
0.05	0.1603	0.3109	0.5300	0.1626	0.2995	0.3412	0.2369	0.2373	0.4177	**0.3890**
Peppers										
0.02	0.1715	0.5315	0.8232	0.4843	0.6548	0.5525	0.4138	0.4138	0.5525	**0.8346**
0.03	0.1513	0.4628	**0.8157**	0.4253	0.5717	0.4930	0.3495	0.3496	0.4663	0.7084
0.05	0.1156	0.3582	**0.7899**	0.3225	0.4256	0.3616	0.2735	0.2338	0.3991	0.5566
House										
0.02	0.2091	0.3433	**0.6628**	0.2018	0.4011	0.4037	0.2585	0.2585	0.3897	0.5738
0.03	0.1727	0.2682	**0.6471**	0.1569	0.2911	0.3346	0.2130	0.2140	0.3195	0.4463
0.05	0.1373	0.2119	**0.6365**	0.1239	0.2194	0.2609	0.1180	0.1770	0.2923	0.2877

work better than other filters. As the results show, the GSK algorithm performs better than the WOA algorithm on the proposed filters, which shows the GSK algorithm's superiority in noise removal.

As shown in column 2, parts 1 and 2 of Table 13, for the noise density of 0.02, on average, in all images, the P2_WOA filter, the Median filter, and the BA_PSO filter have the first to third ranking in terms of PSNR and SSIM, respectively. Similarly, for the noise density of 0.03, the Median filter, the P2_WOA filter, and the BA_PSO filter have the first to third ranking regarding PSNR and SSIM, respectively. For the noise density of 0.05, the Median, the P2_WOA, and the BW_PSO filters have the first to third ranking in PSNR and SSIM, respectively. According to the 6th column, parts 1 and 2 of Table 13, for all noise densities, it is determined that in terms of PSNR and SSIM, the Median, the P2_WOA, and the BA_PSO filters are ranked first to third, respectively. According to parts 1 and 2 of the 7th column of Table 13, for the noise density of 0.02, the FOM and the EPF values of the P2_WOA, the Median, and the NLM filters are ranked first to third, respectively. For the noise density 0.03, on average, the FOM mean rank of the Median and P2_WOA will be the same and equal to one, and the NLM and the BW_PSO filters are ranked second to third, respectively, and in terms of EPF, the P2_WOA, the Median, and the P1_WOA filters are ranked first to third, respectively. For the noise density of 0.05, on average, in terms of FOM, the Median, the P2_WOA, and the NLM are ranked first to third, respectively, and in terms of EPF, the Median filter ranks first, the P1_WOA and P2_WOA filters have the second rank simultaneously, and the NLM filter ranks third. According to section 1 of the last column of Table 13, the Median and the P2_WOA filters both have the first rank in terms of FOM, the NLM and the BW_PSO are located in the second to the third rank, respectively, the BA_PSO has the fourth rank, the Wiener, the Gaussian, the P1_WOA, the Mean, and the GC_PSO filters are placed in the fifth to ninth rank, respectively. On average, the P2_WOA, the Median, and the P1_WOA filters are ranked first to third in terms of EPF, respectively, and the NLM, the GC_PSO, the Gaussian, the BA_PSO, the BW_PSO, the Wiener, and the Mean filters are ranked fourth to tenth, respectively as shown in last column of section 2 of Table 13. Based on all criteria, the Median, the P2_WOA, the BA_PSO, the BW_PSO, the P1_WOA, the NLM, the GC_PSO, the Gaussian, the Wiener, and the Mean filters perform better in reducing SAP noise, respectively. It can be concluded that based on all criteria, the median filter is an efficient filter to remove SAP noise, and after that, the P2_WOA filter is efficient.

Table 14 columns 2 to 5, denotes the mean rank of filters, the higher values are bolded and indicate the filter's better performance. The 6th column of Table 14 shows the mean rank according to all considered criteria, and the last column shows the ranking of the filters. As the last column of Table 14 shows, the P2_GSK method performs better than other filters in Gaussian noise reduction, followed by the P2_WOA filter. In general, the first proposed method was not successful in the noise removal of this image, and bilateral-based filters are better than other filters. Figure 6 shows the results of the proposed filters on the brain MRI image. In that figure, a, b, c, and d represent the original image, noisy (with different standard deviations) images, and filtered-out images with P1_GSK and P2_GSK, respectively.

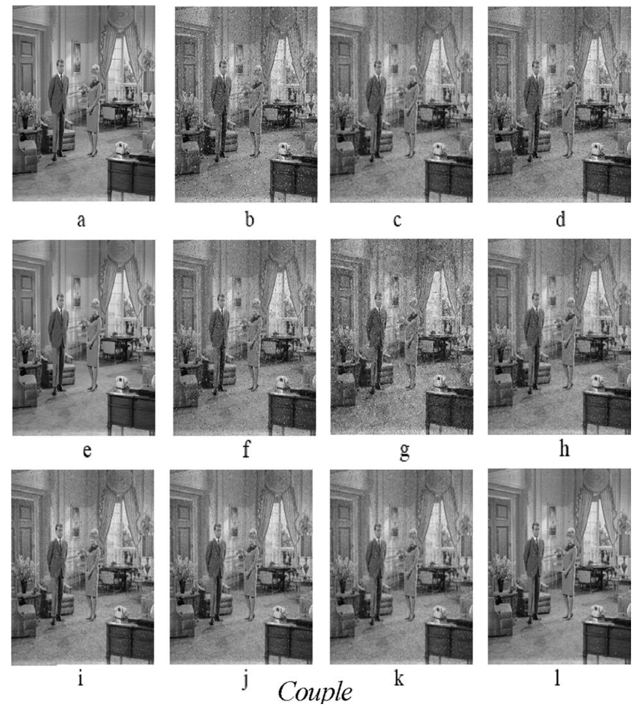

Fig. 5 Images before and after SAP noise reduction

A multi-problem Wilcoxon signed-rank test is used to check the differences between all algorithms. In this method, S + represents the sum of ranks for all images, which describes the first algorithm performs better than the other one, and S – indicates the opposite of the previous one. Larger ranks indicate more considerable performance differences. The *p* value is used for comparison. The null hypothesis is rejected if the *p* value is less than or equal to the assumed significance level of 0.05. The following results show the *p* values and decisions corresponding to the *p* values in bold, and the test is performed with SPSS 26.00. For each standard deviation of Gaussian noise, the performance of the GSK-based proposed filters is compared to other filters in terms of SSIM, PSNR, FOM, and EPF using the Wilcoxon method and listed in Tables 15, 16, 17, 18.

Table 11 All considered criteria results for Gaussian noise reduction of Brains MRI image

MRI	Noisy	Average	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA	P1_GSK	P2_GSK
PSNR													
$\sigma = 20$	23.3199	25.4896	25.9226	25.3979	27.4339	**27.8509**	26.6316	27.7755	27.7768	26.6318	27.4622	26.6318	27.5592
$\sigma = 30$	19.8598	23.7408	22.7093	24.0950	24.0741	24.1174	24.1682	24.7444	24.7478	24.1896	24.9137	24.1902	**24.9185**
$\sigma = 50$	15.8831	20.8137	18.7646	21.6863	20.0983	18.8126	20.9075	21.4665	21.1119	21.1968	21.7373	21.1972	**21.9012**
$\sigma = 70$	13.3471	18.5108	16.1812	19.6038	17.5627	15.0816	18.5401	19.2771	18.6856	19.0572	19.7449	19.0574	**19.7882**
SSIM													
$\sigma = 20$	0.9372	0.9577	0.9627	0.9595	0.9726	**0.9754**	0.9673	0.9746	0.9746	0.9673	0.9734	0.9673	0.9738
$\sigma = 30$	0.8661	0.935	0.9224	0.9446	0.9410	0.9426	0.9411	0.9485	0.9484	0.9408	0.9513	0.9409	**0.9513**
$\sigma = 50$	0.7049	0.8744	0.8186	**0.9041**	0.8588	0.8221	0.8774	0.8897	0.8831	0.8823	0.8968	0.8824	0.8998
$\sigma = 70$	0.5491	0.7992	0.6999	**0.8498**	0.7634	0.6469	0.8009	0.8243	0.8067	0.8164	0.8404	0.8164	0.8401
FOM													
$\sigma = 20$	0.419	0.784	0.5536	0.7704	0.8803	**0.9537**	0.6963	0.8662	0.9050	0.6961	0.8385	0.6915	0.9134
$\sigma = 30$	0.2909	0.5834	0.3485	0.5078	0.6147	**0.6865**	0.485	0.5607	0.6184	0.5323	0.5646	0.5321	0.6105
$\sigma = 50$	0.2245	0.3136	0.2495	0.3472	0.4055	0.3650	0.3013	0.4507	0.3404	0.3527	0.489	0.3597	**0.5734**
$\sigma = 70$	0.1932	0.2473	0.2185	0.2965	0.3121	0.2488	0.2477	0.3592	0.2630	0.2993	0.3517	0.2984	**0.3908**
EPF													
$\sigma = 20$	0.4421	0.243	0.5070	0.3007	0.7069	0.7129	0.5178	0.7121	**0.7362**	0.5179	0.6855	0.5195	0.7352
$\sigma = 30$	0.2987	0.1827	0.3616	0.2185	0.5281	0.4370	0.3577	0.5369	0.5564	0.3881	0.5343	0.3883	**0.5584**
$\sigma = 50$	0.1865	0.1131	0.2292	0.1467	**0.3173**	0.2115	0.1847	0.2999	0.2716	0.2374	0.2791	0.2407	0.3044
$\sigma = 70$	0.1311	0.077	0.1652	0.1078	**0.2087**	0.1481	0.1093	0.1815	0.1497	0.1517	0.177	0.1525	0.1978

Table 12 Friedman test results for Gaussian noise removal

SSIM						FOM						
Filters σ	20	30	50	70	Total rank	Ranking	20	30	50	70	Total rank	Ranking
Mean	10.75	8.83	8	7.33	8.63	9	5.33	7	7.33	7.83	7.00	7
Gaussian	11.5	12	11	11	11.50	12	12	12	12	12	12.00	12
Median	10.25	9.83	10	10	10.00	10	7.83	9.83	7.83	7.67	8.25	9
Wiener	5.5	9.17	9	9	8.00	8	3.33	6.67	6	6.83	5.38	5
NLM	5.67	10.17	12	12	10.25	11	5.5	4.33	10.83	11	8.00	8
PSO_GC	8.67	7	7	6.33	7.25	7	10.67	10.33	9.5	8.67	10.25	11
BPSO_W	3.58	3.5	3.33	4	3.50	3	4.67	1.67	2.67	3.67	3.00	3
BPSO_A	3.42	4.25	6	7.33	5.13	5	6.17	9	9.17	9	8.75	10
P1_WOA	7.58	5	4.08	3.5	5.00	4	9.5	6.67	5	4.83	6.63	6
P2_WOA	2.08	2	1.67	1.5	1.88	2	2.67	3	1.83	1.83	2.25	2
P1_GSK	7.42	5.25	4.58	4.5	5.75	6	8.17	5.67	4.33	3.5	5.25	4
P2_GSK	1.58	1	1.33	1.5	1.13	1	2.17	1.83	1.5	1.17	1.25	1
PSNR							EPF					
Mean	10.5	8.83	8	7.33	8.75	9	11.92	11.83	11.5	11.17	12.00	11
Gaussian	11.67	12	11	11	11.5	12	10	9.67	7.17	6.17	8.25	8
Median	10.33	9.83	10	10	10	10	10.25	10.33	10.17	10.17	11.00	10
Wiener	5.67	8.5	9	9	7.75	8	6.17	6.33	5.17	6	5.88	5
NLM	6	10.67	12	12	10.25	11	3.17	6.17	8	8.5	6.75	6
PSO_GC	8.58	7.17	7	6	7.25	7	7.75	7.67	9.17	9.33	9.25	9
BPSO_W	3.5	3.17	3.67	4	3.5	3	5.33	6	7	6.67	5.75	4
BPSO_A	3.5	5	6	7.67	5.63	5	3.33	6.33	7.33	6.83	6.88	7
P1_WOA	7.58	5.5	4.67	4.5	6	6	7.92	5.17	4	3.17	4.88	3
P2_WOA	2	2	1.83	1.5	1.88	2	3	2.5	2.67	2.83	1.75	1
P1_GSK	7	4.33	3.67	3.5	4.38	4	7.33	4.17	3.33	3.17	3.88	2
P2_GSK	1.67	1	1.17	1.5	1.13	1	1.83	1.83	2.5	4	1.75	1

In Tables 15, 16, 17, 18 the 7th and 13th columns (F1?F2) represent the efficiency between two filters. For $\sigma = 20, 50, 70$ according to the 7th column of Tables 15 and 16, P2_GSK outperforms other filters in terms of SSIM and PSNR except for the P2_WOA filter. For $\sigma = 30$, P2_GSK outperforms all other filters. As the 7th column of Table 17 shows $\sigma = 20$, P2_GSK is better than other filters in terms of FOM, except Wiener and P2_WOA filters. For $\sigma = 30$, P2_GSK is better than other filters, except the BW_PSO filter. For $\sigma = 50$, P2_GSK is better than other filters, except BW_PSO and P2_WOA filters. For $\sigma = 70$, P2_GSK acts better than all other filters.

As the 7th column of Table 18 shows, the following results were obtained in terms of EPF: $\sigma = 20$, P2_GSK is better than other filters, except NLM, BW_PSO, and P2_WOA filters. For $\sigma = 30, 50$, P2_GSK is better than other filters, except P1_WOA, P2_WOA, and P1_GSK filters. For $\sigma = 70$, P2_GSK acts better than other filters, except Wiener, P1_WOA, P2_WOA, and P1_GSK filters. In general, it can be concluded that the P2_GSK filter performs better than other methods on average in terms of all criteria, and this shows the superiority of the GSK-based proposed filter.

According to the 13th column of Table 15, which shows the results of the Wilcoxon method for the P1_GSK filter in terms of SSIM, we have: for $\sigma = 20$, P1_GSK is better than Mean, Gaussian and Median filters, but it is weaker than Wiener, BW_PSO, BA_PSO, P2_WOA, and P2_GSK filters, and P1_GSK does not have the significant difference with NLM, GC_PSO, and P1_WOA filters. For $\sigma = 30$, P1_GSK is better than Mean, Gaussian, Median, Wiener, NLM, and GC_PSO filters, but it is weaker than BW_PSO, P2_WOA, and P2_GSK filters, P1_GSK does not have a significant difference with BA_PSO, and P1_WOA filters. For $\sigma = 50$, P1_GSK is better than Mean, Gaussian, Median, Wiener, NLM, GC_PSO, and BA_PSO filters, but it is weaker than BW_PSO, P2_WOA, and P2_GSK filters; it does not have a significant difference with P1_WOA filter. For $\sigma = 70$, P1_GSK is better than Mean, Gaussian, Median, Wiener, NLM, GC_PSO, and BA_PSO filters, but it is weaker than P1_WOA, P2_WOA, and P2_GSK filters, and it does not have a significant difference with BW_PSO filters.

Table 13 Friedman test results for SAP noise reduction

PSNR							FOM				
Filters	d	0.02	0.03	0.05	Mean rank	Ranking	0.02	0.03	0.05	Mean rank	Ranking
Mean		7.67	7.17	7.00	7.33	7	8.17	8.50	9.00	8.83	8
Gaussian		8.67	8.67	8.17	8.67	9	5.67	7.00	7.17	6.50	6
Median		2.50	**1.17**	**1.17**	**1.33**	**1**	2.00	**1.50**	**1.17**	**1.50**	**1**
Wiener		10.00	9.83	10.00	10.00	10	6.33	5.50	5.50	6.00	5
NLM		6.67	7.67	8.83	8.00	8	3.50	3.67	3.50	3.00	2
GC_PSO		5.50	5.67	6.00	6.00	6	8.33	9.50	9.33	10.00	9
BW_PSO		4.33	4.08	3.50	3.83	4	5.50	4.58	4.58	4.00	3
BA_PSO		3.67	3.42	3.50	3.17	3	5.83	5.42	5.58	5.67	4
P1_WOA		4.83	5.50	5.00	5.00	5	8.17	7.83	7.17	8.00	7
P2_WOA		**1.17**	1.83	1.83	1.67	2	**1.50**	**1.50**	2.00	**1.50**	**1**
SSIM							EPF				
Mean		8.00	7.33	7.00	7.33	7	9.83	9.67	9.50	10.00	10
Gaussian		8.50	8.83	8.17	8.67	9	5.17	5.17	4.67	5.83	6
Median		2.50	**1.17**	**1.17**	**1.33**	**1**	2.83	2.50	**1.83**	1.67	2
Wiener		10.00	10.00	10.00	10.00	10	8.50	8.67	8.17	8.67	9
NLM		6.50	7.67	8.83	8.00	8	3.83	4.17	3.83	3.83	4
GC_PSO		5.33	5.42	6.00	6.00	6	4.33	4.17	4.67	5.00	5
BW_PSO		4.42	4.08	3.17	3.67	4	8.17	8.33	8.83	8.33	8
BA_PSO		3.58	3.50	3.83	3.33	3	7.17	7.00	7.50	7.00	7
P1_WOA		5.00	5.17	5.00	5.00	5	4.00	3.83	3.00	3.17	3
P2_WOA		**1.17**	1.83	1.83	1.67	2	**1.17**	**1.50**	3.00	**1.50**	**1**

Table 14 Friedman test results for Gaussian noise reduction in Brains MRI image

Filters	Criteria	SSIM	PSNR	FOM	EPF	Total rank	Ranking
Mean		2.75	3	4.75	1	2.13	10
Gaussian		1.75	1.75	1	5	1.75	11
Median		8.5	6.25	4.75	2	5.13	8
Wiener		4.5	4	9.25	10	7	4
NLM		5.5	5	9	6.25	6.63	6
PSO_GC		5.25	5	2.75	3.25	3.38	9
BPSO_W		9.63	9.25	8.75	9.75	9.63	2
BPSO_A		8.38	8.25	7.5	9	8.25	3
P1_WOA		5.38	6.63	5.5	5.5	5.75	7
P2_WOA		10.13	10.25	8.75	8.5	9.63	2
P1_GSK		5.88	7.38	5	6.5	6.75	5
P2_GSK		**10.38**	**11.25**	**11**	**11.25**	**12**	**1**

According to the 13th column of Table 16, which shows the results of the Wilcoxon method for the P1_GSK filter in terms of PSNR, we have: for $\sigma = 20$, P1_GSK is better than Mean, Gaussian, Median, and GC_PSO filters, but it is weaker than BW_PSO, BA_PSO, P2_WOA, and P2_GSK filters, and it does not have the significant difference with Wiener, NLM, and P1_WOA filters. For $\sigma = 30$, P1_GSK is better than Mean, Gaussian, Median, Wiener, NLM, GC_PSO, and P1_WOA filters, but it is weaker than BW_PSO, P2_WOA, and P2_GSK filters, and it does not have a significant difference with BA_PSO filter. For $\sigma = 50, 70$, P1_GSK is better than Mean, Gaussian, Median, Wiener, NLM, GC_PSO, BA_PSO, and P1_WOA filters, but it is weaker than P2_WOA and P2_GSK filters, and it does not have a significant difference with BW_PSO filter.

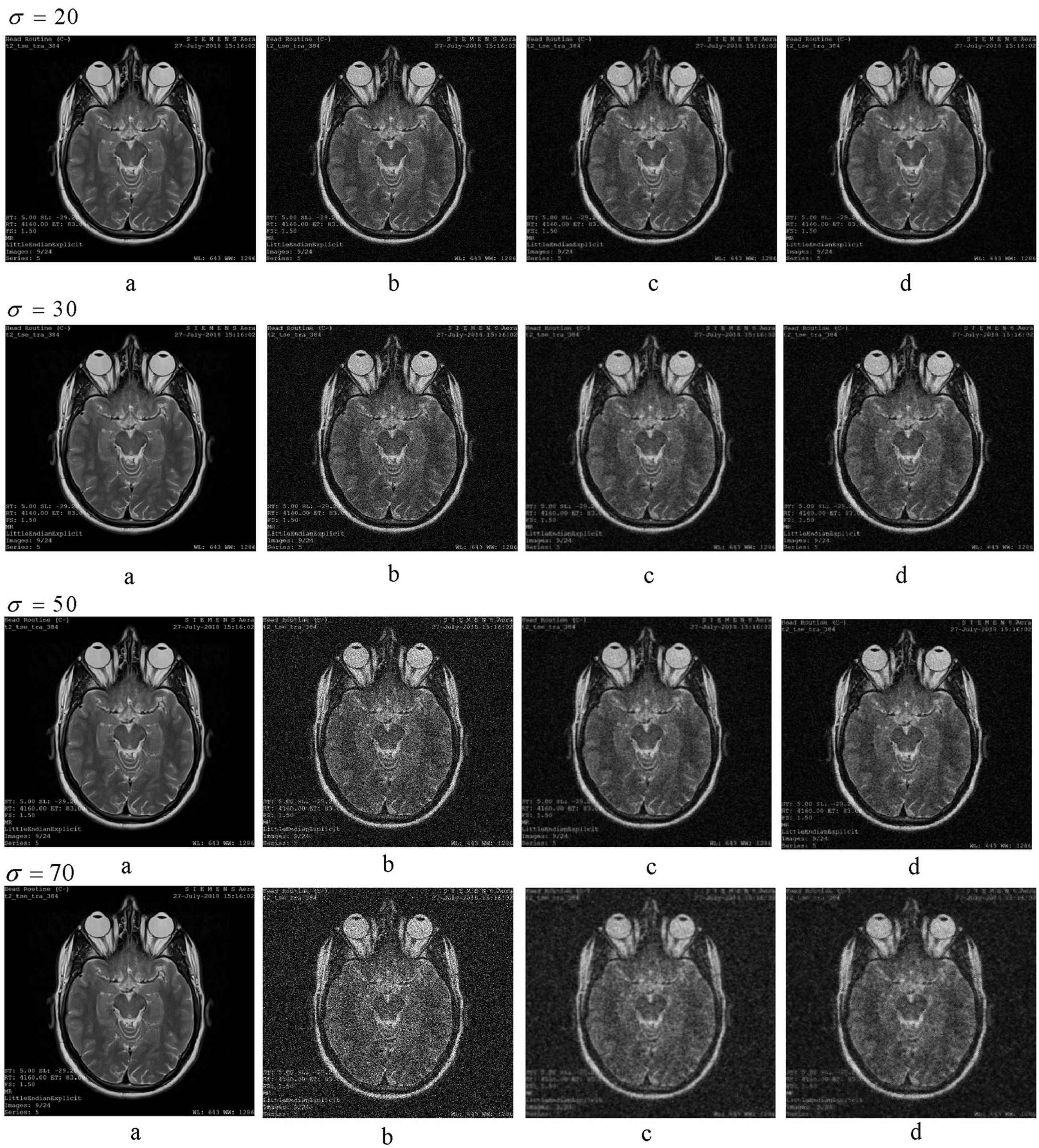

Fig. 6 Brains MRI Images before and after Gaussian noise reduction

According to the 13th column of Table 17, which shows the results of the Wilcoxon method for the P1_GSK filter in terms of FOM, we have: for $\sigma = 20$, P1_GSK is better than Gaussian, and GC_PSO filters, but it is weaker than Mean, Wiener, NLM, BW_PSO, BA_PSO, P2_WOA, and P2_GSK filters, and it does not have the significant difference with Median and P1_WOA filters. For $\sigma = 30$, P1_GSK is better than Gaussian, Median, GC_PSO, BA_PSO, and P1_WOA filters, but it is weaker than BW_PSO, P2_WOA, and P2_GSK filters, and it does not have a significant difference with Mean, Wiener, and NLM filters. For $\sigma = 50$, P1_GSK is better than Mean, Gaussian, Median, Wiener, NLM,

Table 15 The results of the Wilcoxon test, according to SSIM

SSIM												
σ	Filters (F)	S+	S−	*p* value	Decision	F1?F2	Filters	S+	S−	*p* value	Decision	F1?F2
20	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	20	1	0.046	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	1	20	0.046	Reject	<
	P2_GSK vs. NLM	20	1	0.046	Reject	>	P1_GSK vs NLM	4	17	**0.172**	**Retain**	–
	P2_GSK vs GC_PSO	21	0	0.027	Reject	>	P1_GSK vs GC_PSO	10	0	**0.068**	**Retain**	–
	P2_GSK vs BW_PSO	15	0	0.043	Reject	>	P1_GSK vs BW_PSO	0	21	0.028	Reject	<
	P2_GSK vs BA_PSO	20	1	0.046	Reject	>	P1_GSK vs BA_PSO	0	21	0.027	Reject	<
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	1	0	**0.317**	**Retain**	–
	P2_GSK vs P2_WOA	17	4	**0.172**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
30	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	21	0	0.028	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	21	0	0.028	Reject	>
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	21	0	0.028	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.027	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	1	20	0.046	Reject	<
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	3	18	**0.115**	**Retain**	–
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	1.5	4.5	**0.414**	**Retain**	–
	P2_GSK vs P2_WOA	21	0	0.028	Reject	>	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
50	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	21	0	0.028	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	21	0	0.028	Reject	>
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	21	0	0.028	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	1	20	0.046	Reject	<
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	21	0	0.028	Reject	>
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	5	10	**0.498**	**Retain**	–
	P2_GSK vs P2_WOA	7	3	**0.461**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
70	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	21	0	0.028	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	21	0	0.028	Reject	>
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	21	0	0.028	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	6	15	**0.345**	**Retain**	–
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	21	0	0.028	Reject	>
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	0	21	0.027	Reject	<
	P2_GSK vs P2_WOA	8.5	12.5	**0.674**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<

Table 16 The results of the Wilcoxon test, according to PSNR

PSNR												
σ	Filters	S+	S−	*p* value	Decision	F1?F2	Filters	S+	S−	*p* value	Decision	F1?F2
20	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	20	1	0.046	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	2	19	**0.075**	**Retain**	–
	P2_GSK vs. NLM	20	1	0.046	Reject	>	P1_GSK vs NLM	6	15	**0.345**	**Retain**	–
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	20	1	0.046	Reject	>
	P2_GSK vs BW_PSO	20	1	0.046	Reject	>	P1_GSK vs BW_PSO	0	21	0.028	Reject	<
	P2_GSK vs BA_PSO	20	1	0.046	Reject	>	P1_GSK vs BA_PSO	0	21	0.028	Reject	<
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	19	2	**0.075**	**Retain**	–
	P2_GSK vs P2_WOA	17	4	**0.173**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
30	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	21	0	0.028	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	21	0	0.028	Reject	>
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	21	0	0.028	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	1	20	0.046	Reject	<
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	9	12	**0.753**	**Retain**	–
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	21	0	0.028	Reject	>
	P2_GSK vs P2_WOA	21	0	0.028	Reject	>	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
50	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	21	0	0.028	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	21	0	0.028	Reject	>
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	21	0	0.028	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	4	17	**0.173**	**Retain**	–
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	21	0	0.028	Reject	>
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	21	0	0.028	Reject	>
	P2_GSK vs P2_WOA	18	3	**0.116**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
70	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	21	0	0.028	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	21	0	0.028	Reject	>
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	21	0	0.028	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	9	12	**0.753**	**Retain**	–
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	21	0	0.028	Reject	>
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	21	0	0.028	Reject	>
	P2_GSK vs P2_WOA	18	3	**0.6**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<

Table 17 The results of the Wilcoxon test according to FOM

FOM												
σ	Filters	S+	S−	*p* value	Decision	F1?F2	Filters	S+	S−	*p* value	Decision	F1?F2
20	P2_GSK vs. Mean	20	1	0.046	Reject	>	P1_GSK vs. Mean	1	20	0.046	Reject	<
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	20	1	0.046	Reject	>	P1_GSK vs Median	12	9	**0.753**	**Retain**	–
	P2_GSK vs Wiener	18	3	**0.116**	**Retain**	–	P1_GSK vs Wiener	0	21	0.028	Reject	<
	P2_GSK vs. NLM	20	1	0.046	Reject	>	P1_GSK vs NLM	0	21	0.028	Reject	<
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	0	21	0.028	Reject	<
	P2_GSK vs BA_PSO	20	1	0.046	Reject	>	P1_GSK vs BA_PSO	1	20	0.046	Reject	<
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	19	2	**0.075**	**Retain**	–
	P2_GSK vs P2_WOA	15	6	**0.345**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
30	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	19	2	**0.075**	**Retain**	–
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	20	1	0.046	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	11	10	**0.917**	**Retain**	–
	P2_GSK vs. NLM	20	1	0.046	Reject	>	P1_GSK vs NLM	6	15	**0.345**	**Retain**	–
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	9	12	**0.753**	**Retain**	–	P1_GSK vs BW_PSO	0	21	0.028	Reject	<
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	20	1	0.046	Reject	>
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	21	0	0.028	Reject	>
	P2_GSK vs P2_WOA	20	1	0.046	Reject	>	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
50	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	21	0	0.028	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	20	1	0.046	Reject	>
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	21	0	0.028	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	19	2	**0.075**	**Retain**	–	P1_GSK vs BW_PSO	0	21	0.028	Reject	<
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	21	0	0.028	Reject	>
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	19	2	**0.075**	**Retain**	–
	P2_GSK vs P2_WOA	14	7	**0.463**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
70	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	21	0	0.028	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	21	0	0.028	Reject	>
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	21	0	0.028	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	8	13	**0.6**	**Retain**	–
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	21	0	0.028	Reject	>
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	21	0	0.028	Reject	>
	P2_GSK vs P2_WOA	20	1	0.046	Reject	>	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<

Table 18 The results of the Wilcoxon test, according to EPF

EPF												
σ	Filters	S+	S−	*p* value	Decision	F1?F2	Filters	S+	S−	*p* value	Decision	F1?F2
20	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	18	3	**0.116**	**Retain**	–
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	20	1	0.046	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	5	16	**0.249**	**Retain**	–
	P2_GSK vs. NLM	15	6	**0.345**	**Retain**	–	P1_GSK vs NLM	0	21	0.028	Reject	<
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	12	9	**0.753**	**Retain**	–
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	3	18	**0.116**	**Retain**	–
	P2_GSK vs BA_PSO	17	4	**0.173**	**Retain**	–	P1_GSK vs BA_PSO	1	20	0.046	Reject	<
	P2_GSK vs P1_WOA	21	0	0.028	Reject	>	P1_GSK vs P1_WOA	15	6	**0.345**	**Retain**	–
	P2_GSK vs P2_WOA	17	4	**0.173**	**Retain**	–	P1_GSK vs P2_WOA	0	21	0.028	Reject	<
	P2_GSK vs P1_GSK	21	0	0.028	Reject	>	P1_GSK vs P2_GSK	0	21	0.028	Reject	<
30	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	21	0	0.028	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	20	1	0.046	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	12	9	**0.753**	**Retain**	–
	P2_GSK vs. NLM	20	1	0.046	Reject	>	P1_GSK vs NLM	14	7	**0.463**	**Retain**	–
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	19	2	**0.075**	**Retain**	–
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	12	9	**0.753**	**Retain**	–
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	12	9	**0.753**	**Retain**	–
	P2_GSK vs P1_WOA	19	2	**0.075**	**Retain**	–	P1_GSK vs P1_WOA	16	5	**0.249**	**Retain**	–
	P2_GSK vs P2_WOA	19	2	**0.075**	**Retain**	–	P1_GSK vs P2_WOA	3	18	**0.116**	**Retain**	–
	P2_GSK vs P1_GSK	18	3	**0.116**	**Retain**	–	P1_GSK vs P2_GSK	3	18	**0.116**	**Retain**	–
50	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	21	0	0.028	Reject	>	P1_GSK vs Gaussian	20	1	0.046	Reject	>
	P2_GSK vs. Median	21	0	0.028	Reject	>	P1_GSK vs Median	20	1	0.046	Reject	>
	P2_GSK vs. Wiener	21	0	0.028	Reject	>	P1_GSK vs Wiener	18	3	**0.116**	**Retain**	–
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	20	1	0.046	Reject	>
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	21	0	0.028	Reject	>	P1_GSK vs BW_PSO	19	2	**0.075**	**Retain**	–
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	20	1	0.046	Reject	>
	P2_GSK vs P1_WOA	14	7	**0.463**	**Retain**	–	P1_GSK vs P1_WOA	16	5	**0.249**	**Retain**	–
	P2_GSK vs P2_WOA	11	10	**0.917**	**Retain**	–	P1_GSK vs P2_WOA	8	13	**0.6**	**Retain**	–
	P2_GSK vs P1_GSK	14	7	**0.463**	**Retain**	–	P1_GSK vs P2_GSK	7	14	**0.463**	**Retain**	–
70	P2_GSK vs. Mean	21	0	0.028	Reject	>	P1_GSK vs. Mean	21	0	0.028	Reject	>
	P2_GSK vs. Gaussian	20	1	0.046	Reject	>	P1_GSK vs Gaussian	19	2	**0.075**	**Retain**	–
	P2_GSK vs. Median	20	1	0.046	Reject	>	P1_GSK vs Median	20	1	0.046	Reject	>
	P2_GSK vs Wiener	19	2	**0.075**	**Retain**	–	P1_GSK vs Wiener	18	3	**0.116**	**Retain**	–
	P2_GSK vs. NLM	21	0	0.028	Reject	>	P1_GSK vs NLM	19	2	**0.075**	**Retain**	–
	P2_GSK vs GC_PSO	21	0	0.028	Reject	>	P1_GSK vs GC_PSO	21	0	0.028	Reject	>
	P2_GSK vs BW_PSO	20	1	0.046	Reject	>	P1_GSK vs BW_PSO	19	2	**0.075**	**Retain**	–
	P2_GSK vs BA_PSO	21	0	0.028	Reject	>	P1_GSK vs BA_PSO	20	1	0.046	Reject	>
	P2_GSK vs P1_WOA	6	15	**0.345**	**Retain**	–	P1_GSK vs P1_WOA	9.5	11.5	**0.833**	**Retain**	–
	P2_GSK vs P2_WOA	5	16	**0.249**	**Retain**	–	P1_GSK vs P2_WOA	14	7	**0.463**	**Retain**	–
	P2_GSK vs P1_GSK	5.5	15.5	**0.293**	**Retain**	–	P1_GSK vs P2_GSK	15.5	5.5	**0.293**	**Retain**	–

Table 19 The execution time of filters

σ	Mean	Gaussian	Median	Wiener	NLM	GC_PSO	BW_PSO	BA_PSO	P1_WOA	P2_WOA	P1_GSK	P2_GSK
Barbara												
20	0.03	0.035	0.034	0.118	3.0552	153.987	2978.01	2975.495	433.703	8448.866	153.017	7082.656
30	0.028	0.038	0.035	0.119	3.1867	133.483	3131.713	2964.487	574.267	8049.411	283.712	6847.685
50	0.029	0.031	0.035	0.126	3.1807	173.738	3166.065	2968.972	864.318	8708.544	411.023	8189.58
70	0.031	0.028	0.035	0.158	3.1856	176.225	3183.529	3189.06	503.529	8856.344	251.99	8039.307
Boats												
20	0.029	0.038	0.035	0.097	3.1182	237.754	3131.951	3138.784	516.07	5671.362	134.964	5991.875
30	0.043	0.028	0.035	0.102	3.1958	185.77	3110.684	3166.83	528.203	6467.656	168.834	5585.727
50	0.031	0.028	0.035	0.101	3.5141	226.432	3059.916	2982.108	662.117	7219.754	199.986	6023.622
70	0.028	0.029	0.035	0.118	3.1378	203.974	2723.735	2920.032	657.787	7198.452	264.384	6684.572
Hill												
20	0.03	0.029	0.035	0.103	3.1093	142.19	3045.954	2961.877	693.109	6156.326	189.768	6263.08
30	0.034	0.034	0.035	104	3.3549	157.756	3098.803	2929.649	702.494	7923.84	227.011	6207.691
50	0.043	0.029	0.035	0.101	3.1906	139.762	3114.774	2924.958	727.982	7223.56	383.12	6911.45
70	0.031	0.033	0.035	0.098	3.2889	132.498	2983.197	2927.853	699.869	8137.202	344.301	6961.083
Couple												
20	0.03	0.051	0.035	0.226	3.1363	108.48	3164.815	2764.66	408.898	6417.769	138.454	6253.925
30	0.084	0.029	0.037	0.136	3.0271	100.214	3796.254	3579.322	495.274	7142.277	165.349	6223.353
50	0.032	0.031	0.045	0.134	3.2091	133.046	3811.057	3630.811	671.423	8868.97	247.843	8666.027
70	0.031	0.044	0.057	0.215	3.1837	140.862	3785.207	3575.181	549.774	10,664.148	263.223	8479.445
Peppers												
20	0.027	0.033	0.034	0.053	1.111	40.769	777.276	713.808	322.837	1558.812	40.129	1385.312
30	0.027	0.027	0.034	0.051	1.103	37.345	764.437	712.063	309.317	1514.819	54.781	1441.287
50	0.028	0.029	0.034	0.077	1.1571	40.965	756.805	723.496	313.31	1822.188	54.345	1454.856
70	0.027	0.029	0.033	0.051	1.1063	38.055	772.014	619.194	308.231	1782.606	72.953	1541.356
House												
20	0.04	0.029	0.035	0.053	1.0732	40.035	733.319	820.3	167.909	1493.068	51.217	1449.7278
30	0.026	0.015	0.102	0.236	1.0012	31.367	919.927	920.513	168.923	2016.288	61.659	1475.073
50	0.05	0.03	0.036	0.061	1.0877	31.928	922.958	931.894	194.68	2222.874	70.711	1533.73
70	0.038	0.029	0.042	0.061	1.0851	31.597	904.798	932.659	202.541	2338.861	83.017	1709.991

GC_PSO, and BA_PSO filters, but it is weaker than BW_PSO, P2_WOA, and P2_GSK filters, and it does not have a significant difference with P1_WOA filter. For $\sigma = 70$, P1_GSK is better than Mean, Gaussian, Median, Wiener, NLM, GC_PSO, BA_PSO, and P1_WOA filters, but it is weaker than P2_WOA and P2_GSK filters, and it does not have a significant difference with BW_PSO filter.

According to the 13th column of Table 18, which shows the results of the Wilcoxon method for the P1_GSK filter in terms of EPF, we have: for $\sigma = 20$, P1_GSK is better than Mean and Median filters. However, it is weaker than NLM, BA_PSO, P2_WOA, and P2_GSK filters, and it does not have a significant difference with Gaussian, Wiener, GC_PSO, BW_PSO, and P1_WOA filters. For $\sigma = 30$, P1_GSK is better than Mean, Gaussian, and Median filters, and it does not have a significant difference with Wiener, NLM, GC_PSO, BW_PSO, BA_PSO, P1_WOA P2_WOA, and P2_GSK filters. For $\sigma = 50$, P1_GSK is better than Mean, Gaussian, Median, NLM, GC_PSO, and BA_PSO filters, and it does not have a significant difference with Wiener, BW_PSO, P1_WOA, P2_WOA, and P2_GSK. For $\sigma = 70$, P1_GSK is better than Mean, Median, GC_PSO, and BA_PSO filters, and it does not have a significant difference with Gaussian, Wiener, NLM, BW_PSO, P1_WOA, P2_WOA, and P2_GSK filters. On average, it can be concluded that the P1_GSK performs better than the P1_WOA and GC_PSO filters.

Algorithms Complexity

The computational complexity of the de-noising filters is described below:

$$O\left(N_{\text{iter}} \times N_{\text{pop}} \times O(\text{fitness})\right), \tag{33}$$

where N_{iter} and N_{pop} indicate the maximum number of iterations and population size, respectively.

In algorithms that neighboring radius "r" and mask size "w" are considered as optimization parameters like BW_PSO, P1_WOA, P2_WOA, P1_GSK, and P2_GSK $O(fitness)$ is as follows:

$$O(\text{fitness}) = O(\text{Image}_{\text{size}} \times w_{\text{size}})$$

$$w_{\text{size}} = w \times w \text{ or } w_{\text{size}} = (2r + 1) \times (2r + 1),\ r = 1, 2, \ldots \tag{34}$$

In other algorithms like GC_PSO and BA_PSO, the computational complexity of the fitness function "$O(fitness)$" is as follows:

$$O(\text{fitness}) = O(\text{Image}_{\text{size}}) \tag{35}$$

where $\text{Image}_{\text{size}}$ describes the size of an image.

The execution time of nature-inspired filters depends on factors such as the number of iterations, population size, the number of parameters, the length of variable ranges, and window size, whether fixed or considered an optimization parameter. Therefore, the execution time of P2_WOA and P2_GSK filters will be longer than others because these algorithms must optimize the number of 7 parameters. The execution time of P1_WOA and P1_GSK filters is also more than GC_PSO because the mask size is an optimization parameter. Regardless of the variables range, the execution time of the BW_PSO filter is more extended than BA_PSO since the neighboring radius is an optimization parameter. One of the variable ranges in both BW_PSO and BA_PSO filters is the same. However, the size of the second variable ranges in the BW_PSO filter is smaller than that of the second variable ranges in the BA_PSO filter, increasing the processing speed. However, in the BW_PSO method, the radius of the neighborhood is variable, which increases the evaluation time of this algorithm. In general, considering the same variable ranges for both filters, "BW_PSO and BA_PSO filters," the execution time of the BW_PSO will be longer due to the variable neighboring radius. The execution time of the filters is listed in Table 19. All computations were implemented and executed using MATLAB R2012b running on a PC with core i5-2410 M (2.30 GHz) CPU and 4 GB RAM running Win7 OS.

Table 19 shows that the execution time of P2_GSK and P2_WOA filters are more extended than all filters, and subsequently, the execution time of BW_PSO and BA_PSO filters is longer. The execution time of P2_GSK and P1_GSK filters is longer than P2_WOA and P1_WOA, respectively. This shows that the GSK algorithm is faster than WOA.

Convergence Curve

Figure 7 illustrates the convergence attributes based on the fitness of all algorithms to analyze the convergence attitude of algorithms. For instance, the convergence rate of algorithms $\sigma = 20$ in the 'Couple' image is represented. In Fig. 7, "a, b, c, d, and e" represent the convergence rates of GC_PSO, BW_PSO, BA_PSO, P1_WOA-P2_GSK, and P1_WOA-P1_GSK filters, respectively. The GSK-based filter convergence velocity is higher in the initial stages of the optimization procedure.

In fact, due to the use of the generalized Cauchy function instead of the Gaussian function in the spatial and intensity domain of the bilateral filter, which is a heavy-tailed function compared to the Gaussian function, and neighboring radius optimization, the second proposed filter performs better than other filters. Since the mask size optimization is considered in the first proposed filter, this filter performs better than the GC_PSO filter. On the other hand, this filter has a weaker performance than the second filter because the generated mask is swept over the noisy image and does not consider spatial information of the image pixels. Because the neighborhood radius is optimized in the BW_PSO filter, this filter has a more robust performance than the BA_PSO filter. One of the disadvantages of the second proposed filter is its long execution time compared to the first one.

Conclusion and Future Directions

To summarize, we present two effective WOA-based and GSK-based filters for noise reduction. The GSK's novelty, the lack of widespread application in image processing, and the ability to solve complex and large-scale problems, as well as WOA's ability to solve image processing problems, have led to the use of GSK and WOA in this research. Since the constancy of the selected mask size can affect the efficiency of the filter made by the GC function. First, a mask was designed using the GC function, and the parameters of this function and the size of the chosen mask were optimized by maximizing the PSNR value as the fitness function with WOA and GSK. In most bilateral-based designed filters, the neighborhood radius is constant and the parameters of the intensity and spatial domain, are the Gaussian function parameters that should be optimized. The similar and better performance of the GC function in noise removal, compared to the Gaussian function, as well as the lack of its usage in the bilateral filter in previous works, has caused it to be used in the bilateral filter. So, a hybrid filter was designed by replacing the GC function with a Gaussian function in the bilateral filter, and the domain and range parameters of the GC function, as well as the size of the neighboring radius,

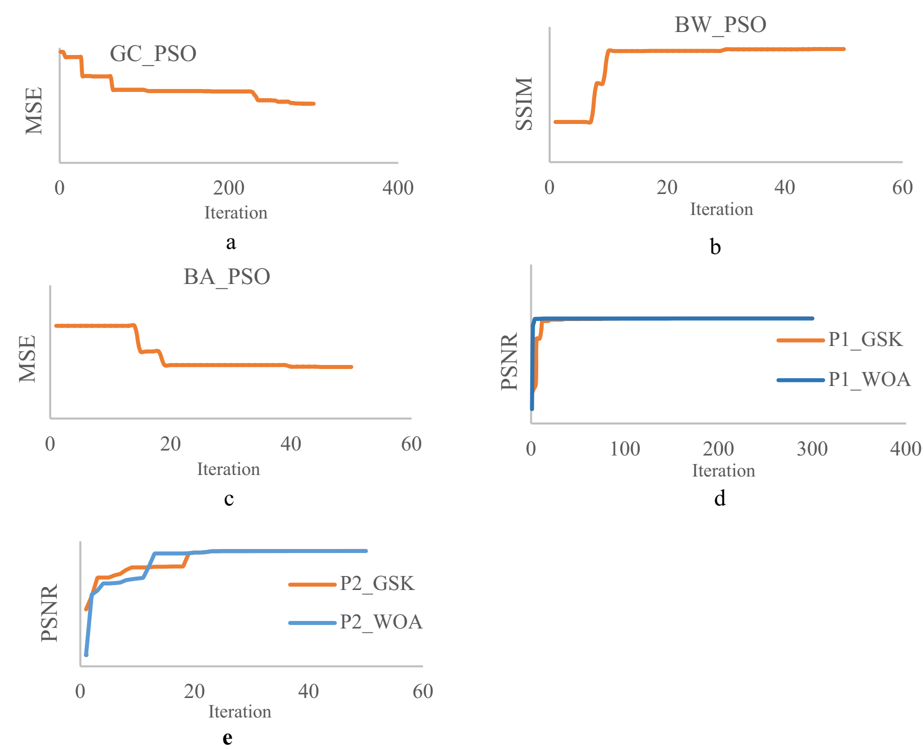

Fig. 7 Convergence curve of the algorithms

were optimized by maximizing the PSNR as a fitness function using WOA and GSK. The GSK and WOA-based proposed filters are compared with each other and classical filters, as well as the PSO-based GC filter and two PSO-based bilateral filters (BW_PSO, BA_PSO) on various images corrupted with a different standard deviation of Gaussian noise. Also, a comparison is made between the WOA-based proposed filters and other filters on images corrupted with various densities of the SAP noise. On average, the superior performance of the P2_GSK and P2_WOA filters is achieved in terms of PSNR and SSIM in Gaussian noise removal. However, the P1_GSK and P2_GSK filters have a more robust performance than those of P1_WOA and P2_WOA in Gaussian noise removal. Following the P2_GSK and P2_WOA filters in terms of PSNR and SSIM, the BW_PSO, P1_GSK, P1_WOA, BA_PSO, and GC_PSO filters work better than traditional methods in Gaussian noise removal. It was also found that as the Gaussian noise standard deviation increased, the parameters of the GC function were quantified by the WOA (or GSK) so that the GC distribution curve was closer to the Gaussian distribution curve. However, the tail was heavier than the Gaussian distribution. Since the mean ranking of the P2_GSK and P2_WOA filters is high in terms of the FOM and EPF, the second proposed method preserves the edges and structural details of the image. The P1_GSK filter performs better than P1_WOA in terms of EPF, and in terms of FOM, it performs better than P1_WOA either. In general, it can be said that the second proposed filter is better than all other filters in terms of PSNR, SSIM, FOM, and EPF. The first proposed filter is also better than all other filters in terms of PSNR, SSIM, and EPF, and in terms of FOM, they are better than other filters after BW_PSO and Wiener.

In the SAP noise removal, after the median filter, the P2_WOA filter is more efficient than other methods, and after these filters, the BA_PSO, the BW_PSO, the P1_WOA, and the GC_PSO methods work better than classical filters in terms of PSNR and SSIM. The P2_WOA filters out the SAP noise and after the median filter is better than all other filters in terms of EPF and FOM.

The P1_WOA filter performs better than other filters in terms of EPF, and in terms of FOM performs weaker than NLM, BW_PSO, BA_PSO, Wiener, and Gaussian filters in SAP noise removal. According to the obtained results, it can be said that the second proposed filter performs better than other filters in SAP noise removal after the median filter, and the first proposed filter is weak in removing the SAP noise compared to other filters. In other words, the proposed filters are not suitable for SAP noise reduction.

The results obtained in "Statistical Analysis" on brain MRI images indicate that the second proposed filter with GSK and WOA algorithm works better than other filters in terms of all criteria. The second proposed filter is ranked fifth with the GSK algorithm and seventh with the WOA algorithm. In general, the proposed filters with the GSK algorithm perform better than the WOA algorithm. However, the first proposed filter is more successful in removing the noise than the second proposed filter.

Since both the neighborhood radius and the parameters of the GC function are optimized instead of the parameters of the Gaussian function in the bilateral filter, and considering that the GC function has a better performance compared to the Gaussian function, the second proposed filter performs better than other filters. The selected mask size is optimized in the first proposed filter, so this filter is better than the GC_PSO method. GSK-based filters get the first ranking among all other filters. Non-parametric tests like Friedman's and Wilcoxon's tests are utilized to statistically evaluate the performance of proposed filters with a significance level of 0.05. The results of the used non-parametric tests show that the GSK-based proposed filters are better than WOA-based filters.

One of the advantages of the proposed filters is the significant noise reduction and their easy design. GSK-based filters are more robust and faster, which is one of their advantages. One of the disadvantages of the second proposed filter is its long execution time, and another disadvantage is the lack of precise upper and lower bounds of the GC function parameters. A method can be investigated for future work to find the exact limits for the parameters. Adaptive meta-heuristic algorithms can also be used to increase the convergence speed. Algorithms can also be used to avoid premature convergence, local optimal trapping, and imbalance between exploration and exploitation. Implementing the proposed method for filtering other noise types, such as speckle noise, is also possible. Also, the proposed methods can be examined on color, ultrasound, and satellite images.

Author Contributions The authors confirm contribution to the paper as follows: study conception and design: MN, FMK, NJN; data collection: MN; analysis and interpretation of results: MN; draft manuscript preparation: MN. All authors reviewed the results and approved the final version of the manuscript.

Funding This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data Availability Data sets are in the form of images that are available at: https://www.kaggle.com/datasets/saeedehkamjoo/standard-test-images and the result data sets are generated during the current study and are available from the corresponding author.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Research involving human and/or animals The research does not involve humans and/or animals.

Informed consent Since the research does not involve humans and/or animals, there is no informed consent statement.

References

1. Cho TS, Zitnick CL, Joshi N, Kang SB, Szeliski R, Freeman WT. Image restoration by matching gradient distributions. IEEE Trans Pattern Anal Mach Intell. 2012;34:683–94.
2. Zhang H, Yang J, Zhang Y, Huang TS. Image and video restorations via nonlocal kernel regression. IEEE Trans Cybernet. 2013;43:1035–46.
3. Sankaran KS, Bhuvaneshwari S, Nagarajan V. A new edge preserved technique using iterative median filter. In: Communications and signal processing (ICCSP), 2014 international conference on; 2014. pp. 1750–1754.
4. Gonzalez C, Woods E. Digital image processing. New York: Addison-Wesley; 1995.
5. Kazubek M. Wavelet domain image denoising by thresholding and Wiener filtering. IEEE Signal Process Lett. 2003;10:324–6.
6. Pitas I, Venetsanopoulos AN. Nonlinear digital filters: principles and applications, vol. 84. Berlin: Springer; 2013.
7. Tomasi C, Manduchi R. Bilateral filtering for gray and color images. In: Iccv; 1998. pp. 2.
8. Hassani H, Mahmoudvand R, Yarmohammadi M. Filtering and denoising in linear regression analysis. Fluctuat Noise Lett. 2010;9:343–58.
9. Portilla J, Strela V, Wainwright MJ, Simoncelli EP. Image denoising using scale mixtures of Gaussians in the wavelet domain. IEEE Trans Image Process. 2003;12:1338–51.
10. Dabov K, Foi A, Katkovnik V, Egiazarian K. Image restoration by sparse 3D transform-domain collaborative filtering. In: Image processing: algorithms and systems VI; 2008. pp. 681207.
11. Vishnu SD, Rajan S, Sowmya V, Soman K. Hyperspectral image denoising: A least square approach using wavelet filters. In: 2017 International conference on advances in computing, communications and informatics (ICACCI); 2017. pp. 805–811.
12. Buades A, Coll B, Morel JM. A non-local algorithm for image denoising. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05); 2005. pp. 60–65.
13. Zhou Y, Zang H, Xu S, He H, Lu J, Fang H. An iterative speckle filtering algorithm for ultrasound images based on bayesian nonlocal means filter model. Biomed Signal Process Control. 2019;48:104–17.
14. Mehta N, Prasad S. Speckle noise reduction and entropy minimization approach for medical images. Int J Inf Technol. 2021;13:1457–62.
15. Frosio I, Kautz J. Statistical nearest neighbors for image denoising. IEEE Trans Image Process. 2018;28:723–38.
16. Young SI, Girod B, Taubman D. Gaussian lifting for fast bilateral and nonlocal means filtering. IEEE Trans Image Process. 2020;29:6082–95.
17. Karami A, Tafakori L. Image denoising using generalised Cauchy filter. IET Image Proc. 2017;11:767–76.
18. Nabahat M, Modarres Khiyabani F, Jafari Navmipour N. Optimization of bilateral filter parameters using a whale optimization algorithm. Res Math. 2022;9:2140863.
19. Wang C, Xue B, Shang L. PSO-based parameters selection for the bilateral filter in image denoising. In: Proceedings of the genetic and evolutionary computation conference; 2017. pp. 51–58.
20. Asokan A, Anitha J. Adaptive Cuckoo Search based optimal bilateral filtering for denoising of satellite images. ISA Trans. 2020;100:308–21.
21. Karnati V, Uliyar M, Dey S. Fast non-local algorithm for image denoising. In: 2009 16th IEEE international conference on image processing (ICIP); 2009. pp. 3873–3876.
22. Dhanushree M, Priyadharsini R, Sree Sharmila T. Acoustic image denoising using various spatial filtering techniques. Int J Inf Technol. 2019;11:659–65.

23. Goyal B, Dogra A, Agrawal S, Sohi B, Sharma A. Image denoising review: from classical to state-of-the-art approaches. Inf Fusion. 2020;55:220–44.
24. Laine S, Karras T, Lehtinen J, Aila T. High-quality self-supervised deep image denoising. In: Advances in neural information processing systems; 2019. pp. 6970–6980.
25. Pan H, Wen Y-W, Zhu H-M. A regularization parameter selection model for total variation based image noise removal. Appl Math Model. 2019;68:353–67.
26. Zhong T, Cheng M, Lu S, Dong X, Li Y. RCEN: a deep-learning-based background noise suppression method for DAS-VSP records. IEEE Geosci Remote Sens Lett. 2021;19:1–5.
27. Cao B, Gu Y, Lv Z, Yang S, Zhao J, Li Y. RFID reader anticollision based on distributed parallel particle swarm optimization. IEEE Internet Things J. 2020;8:3099–107.
28. Zhou W, Guo Q, Lei J, Yu L, Hwang JN. IRFR-Net: interactive recursive feature-reshaping network for detecting salient objects in RGB-D images. In: IEEE transactions on neural networks and learning systems; 2021.
29. Liu Y, Zhang Z, Liu X, Wang L, Xia X. Efficient image segmentation based on deep learning for mineral image classification. Adv Powder Technol. 2021;32:3885–903.
30. Hussain J, Vanlalruata J. Image denoising to enhance character recognition using deep learning. Int J Inf Technol. 2022;14:3457–69.
31. Chaurasiya R, Ganotra D. Deep dilated CNN based image denoising. Int J Inf Technol. 2023;15:137–48.
32. Zhang Z, Ding S, Jia W. A hybrid optimization algorithm based on cuckoo search and differential evolution for solving constrained engineering problems. Eng Appl Artif Intell. 2019;85:254–68.
33. Goldberg DE. Genetic algorithm. In: Search, optimization and machine learning; 1989. pp. 343–349.
34. dos Santos Coelho L, Mariani VC. Improved differential evolution algorithms for handling economic dispatch optimization with generator constraints. Energy Convers Manag. 2007;48:1631–9.
35. Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim. 2007;39:459–71.
36. Yang X-S. Firefly algorithm. Nat Inspir Metaheurist Algorithms. 2008;20:79–90.
37. Eberhart R, Kennedy J. A new optimizer using particle swarm theory. In: Micro machine and human science, 1995. MHS'95., Proceedings of the sixth international symposium on; 1995. pp. 39–43.
38. Mirjalili S. Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl-Based Syst. 2015;89:228–49.
39. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM. Salp Swarm Algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw. 2017;114:163–91.
40. Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw. 2014;69:46–61.
41. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95:51–67.
42. Yao X. A new simulated annealing algorithm. Int J Comput Math. 1995;56:161–8.
43. Lee KS, Geem ZW. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput Methods Appl Mech Eng. 2005;194:3902–33.
44. Rao RV, Savsani VJ, Vakharia D. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aid Des. 2011;43:303–15.
45. Mohamed AW, Hadi AA, Mohamed AK. Gaining-sharing knowledge based algorithm for solving optimization problems: a novel nature-inspired algorithm. Int J Mach Learn Cybern. 2020;11:1501–29.
46. Nadimi-Shahraki MH, Taghian S, Mirjalili S. An improved grey wolf optimizer for solving engineering problems. Expert Syst Appl. 2021;166: 113917.
47. Shen Y, Zhang C, Gharehchopogh FS, Mirjalili S. An improved whale optimization algorithm based on multi-population evolution for global optimization and engineering design problems. Expert Syst Appl. 2023;215: 119269.
48. Nadimi-Shahraki MH, Taghian S, Mirjalili S, Faris H. MTDE: an effective multi-trial vector-based differential evolution algorithm and its applications for engineering design problems. Appl Soft Comput. 2020;97: 106761.
49. Nadimi-Shahraki MH, Taghian S, Mirjalili S, Abualigah L. Binary aquila optimizer for selecting effective features from medical data: a COVID-19 case study. Mathematics. 2022;10:1929.
50. Nadimi-Shahraki MH, Moeini E, Taghian S, Mirjalili S. DMFO-CD: a discrete moth-flame optimization algorithm for community detection. Algorithms. 2021;14:314.
51. Nadimi-Shahraki MH, Taghian S, Zamani H, Mirjalili S, Elaziz MA. MMKE: multi-trial vector-based monkey king evolution algorithm and its applications for engineering optimization problems. PLoS ONE. 2023;18: e0280006.
52. Agrawal P, Ganesh T, Mohamed AW. A novel binary gaining–sharing knowledge-based optimization algorithm for feature selection. Neural Comput Appl. 2021;33:5989–6008.
53. Mohamed AW, Abutarboush HF, Hadi AA, Mohamed AK. Gaining-sharing knowledge based algorithm with adaptive parameters for engineering optimization. IEEE Access. 2021;9:65934–46.
54. Agrawal P, Alnowibet K, Wagdy Mohamed A. Gaining-sharing knowledge based algorithm for solving stochastic programming problems. 2022.
55. Sakthidasan K, Nagappan NV. Noise free image restoration using hybrid filter with adaptive genetic algorithm. Comput Electr Eng. 2016;54:382–92.
56. Moreno J, Jaime B, Saucedo S. Towards no-reference of peak signal to noise ratio. Editorial Preface, vol. 4; 2013.
57. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP. Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process. 2004;13:600–12.
58. Yang XS, Deb S. Cuckoo search via Lévy flights. In: Nature & biologically inspired computing, 2009. NaBIC 2009. World Congress on; 2009. pp. 210–214.
59. Dou L, Xu D, Chen H, Liu Y. Image de-noising based on mathematical morphology and multi-objective particle swarm optimization. In: Ninth international conference on digital image processing (ICDIP 2017); 2017. p. 104202I.
60. Kumar N, Shukla H, Tripathi R. Image restoration in noisy free images using fuzzy based median filtering and adaptive particle swarm optimization-richardson-lucy algorithm. Int J Intell Eng Syst. 2017;10:50–9.
61. Singh R, Vashishath M, Qamar S. Application of 'most'fuzzy linguistic quantifier to filter impulse noise. Int J Inf Technol. 2019;11:141–8.
62. Huynh-Thu Q, Ghanbari M. Scope of validity of PSNR in image/video quality assessment. Electron Lett. 2008;44:800–1.
63. Akram A, Ismail A. "Comparison of edge detectors. Int J Comput Sci Inf Technol Res. 2013;1:16–24.
64. Amitab K, Maji AK, Kandar D. Speckle noise filtering in SAR images using fuzzy logic and particle swarm optimization. J Comput Methods Sci Eng. 2018;18:859–73.
65. Carrillo RE, Aysal TC, Barner KE. Generalized Cauchy distribution based robust estimation. In: 2008 IEEE international conference on acoustics, speech and signal processing; 2008. pp. 3389–3392.
66. Miller J, Thomas J. Detectors for discrete-time signals in non-Gaussian noise. IEEE Trans Inf Theory. 1972;18:241–50.

67. Arce GR. Nonlinear signal processing: a statistical approach. Oxford: Wiley; 2005.
68. Shi Y, Eberhart RC. Empirical study of particle swarm optimization. In: Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406); 1999. pp. 1945–1950.
69. Derrac J, García S, Molina D, Herrera F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput. 2011;1:3–18.