mathematics

Article

Comparison of Circular Symmetric Low-Pass Digital IIR Filter Design Using Evolutionary Computation Techniques

Omar Avalos [1], Erik Cuevas [1,*], Jorge Gálvez [1], Essam H. Houssein [2] and Kashif Hussain [3]

1 Departamento de Electrónica, Universidad de Guadalajara, CUCEI, Av. Revolución, Guadalajara 1500, Mexico; Omar.Avalos@cutonala.udg.mx (O.A.); georgej_galroz@hotmail.com (J.G.)
2 Department of Computer Science, Faculty of Computers & Information, Minia University, Minia 61519, Egypt; essam.halim@mu.edu.eg
3 Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu 610054, China; k.hussain@uestc.edu.cn
* Correspondence: erik.cuevas@cucei.udg.mx; Tel.: +52-33-1378-5900 (ext. 27714)

Received: 10 July 2020; Accepted: 25 July 2020; Published: 26 July 2020

Abstract: The design of two-dimensional Infinite Impulse Response (2D-IIR) filters has recently attracted attention in several areas of engineering because of their wide range of applications. Synthesizing a user-defined filter in a 2D-IIR structure can be interpreted as an optimization problem. However, since 2D-IIR filters can easily produce unstable transfer functions, they tend to compose multimodal error surfaces, which are computationally difficult to optimize. On the other hand, Evolutionary Computation (EC) algorithms are well-known global optimization methods with the capacity to explore complex search spaces for a suitable solution. Every EC technique holds distinctive attributes to properly satisfy particular requirements of specific problems. Hence, a particular EC algorithm is not able to solve all problems adequately. To determine the advantages and flaws of EC techniques, their correct evaluation is a critical task in the computational intelligence community. Furthermore, EC algorithms are stochastic processes with random operations. Under such conditions, for obtaining significant conclusions, appropriate statistical methods must be considered. Although several comparisons among EC methods have been reported in the literature, their conclusions are based on a set of synthetic functions, without considering the context of the problem or appropriate statistical treatment. This paper presents a comparative study of various EC techniques currently in use employed for designing 2D-IIR digital filters. The results of several experiments are presented and statistically analyzed.

Keywords: filter design; evolutionary computational techniques; signal processing; IIR filters

1. Introduction

Two-dimensional (2D) digital filters have been widely used in many areas of engineering, such as signal processing, computer vision, among others [1–3]. Based on their impulse response, 2D filters are categorized into two main groups: Infinite impulse response (IIR) filters and finite impulse response (FIR) models. In general, IIR filters produce higher performance than their similar FIR models considering fewer parameters in their configuration [4,5]. However, since 2D-IIR filters easily produce unstable transfer functions, it is quite difficult to design them in comparison with their counterparts [6].

The objective of filter design is to approximate the response of a determined filter to the desired behavior provided by the user requirements. The design methodologies for 2D-IIR filters can be grouped into two categories [7–9]: transformation methods and optimization-based techniques. In a transformation method, the design of a 2D filter is constructed from a one-dimensional prototype that fulfills the required response. Under this scheme, filters are designed in a fast way, but adversely with low accuracy. On the other hand, in optimization-based techniques, the design of a 2D-IIR filter is interpreted as an optimization problem. Therefore, optimization algorithms are used to find the best suitable filter over a set of candidate solutions. Due to their synthesizing problems, the design of 2D-IIR filters is normally conducted through optimization tools instead of analytical methods [10]. Under the optimization-based methodology, filters maintain a better accuracy than transformation methods. However, their design is highly dependent on the used optimization method. Normally, the generation of a filter through an optimization-based approach is computationally expensive and takes more time regarding transformation methods.

In optimization-based methods, the design process consists of finding the optimal parameters of a 2D-IIR filter that present the best possible resemblance with the desired response. To guide the optimization process, an objective function J that evaluates the similarity between the candidate filter and the desired response must be defined. However, since 2D-IIR filters can easily produce unstable transfer functions, they tend to generate multimodal error surfaces, which are computationally difficult to optimize [11]. This kind of objective function makes the operation of derivative-based techniques difficult since they can be easily trapped in a suboptimal solution [12]. Under such conditions, approaches that use the gradient descent method as basis algorithms are scarcely considered for the design of 2D-IIR filters [13].

On the other hand, EC techniques are optimization methods motivated in our scientific understanding of natural or social behavior, which at some abstraction level can be considered as search procedures [14]. In the literature, EC approaches have shown better performance than those based on gradient methods in terms of robustness and accuracy [12]. As a result, several design methods for 2D-IIR filters have been proposed considering different EC techniques such as the Particle Swarm Optimization (PSO) [15], Genetic Algorithms (GA) [16], and Artificial Bee Colony (ABC) [17].

The capability of determining an optimal solution of an EC method is directly related to its ability to determine the balance between exploration and exploitation of the search space [18] correctly. In the exploration stage the EC technique examines new solutions through the search space, while in the exploitation stage the local search of previously visited locations is carried out in order to improve its quality. If the exploration stage is mainly conducted, the efficiency of the optimization process is degraded, but the capability of determining new potential solutions increases [19]. Otherwise, if the exploitation stage is privileged, the current solutions are highly refined. However, in this particular case, the technique achieves sub-optimal solutions as a result of a scarce exploration [20]. Furthermore, each specific problem usually requires a different trade-off between both stages [21]. Since the exploration–exploitation balance is an unsolved issue within EC community, each EC technique introduces an experimental solution through the combination of deterministic models and stochastic principles. Under such conditions, each EC technique holds distinctive characteristics that properly satisfy the requirements of particular problems [22–24]. Therefore, a particular EC algorithm is not able to solve all problems adequately.

Recently, it has been also identified that each EC scheme presents a different tendency to analyze repeatedly certain areas of the search space or even outside the search space with the independence of the obtained objective function values. This effect is presented as a result of the operation of the specific operators of each EC method or in the early stages of the evolution process. Since this fact adversely affects the obtained results for each EC method, some strategies [25] have been suggested to handle this problem. Among these techniques, the appropriate configuration of parameters represents one of the most common strategies reducing this effect and other undesired behaviors.

In order to determine the advantages and limitations of EC methods, their correct evaluation is an important task in the computational intelligence community. In general, EC algorithms are complex pieces of software with several operators and stochastic branches. Under such circumstances, appropriate statistical methods must be considered for obtaining significant conclusions.

Although several comparisons among EC methods have been reported in the literature [26–28], their conclusions are based on a set of synthetic functions, without considering the context of the problem or appropriate statistical treatment.

This paper reports a comparative study of various EC techniques when they are applied to the complex application of 2D-IIR filter design. In the comparison, special attention is paid to the most popular EC algorithms currently in use, such as PSO, ABC, DE, HS, GSA and FPA. In the experiments, the statistical analysis of Wilcoxon and Bonferroni has been conducted to validate the results obtained by these techniques.

The rest of this manuscript is organized as follows: In Section 2, a brief description of each algorithm used in the comparison is presented. In Section 3 the 2D-IIR filter design problem is formulated from the optimization point of view. The experimental results and the statistical analysis are presented in Section 4. In Section 5, a discussion is conducted. Finally, in Section 6 the conclusions are reported.

2. Evolutionary Computation Techniques

Evolutionary computation (EC) techniques are powerful tools for solving complex optimization problems. Different from classical methods, EC algorithms are mainly appropriated to face multimodal objective functions.

Most of the EC techniques are inspired by nature or social phenomena, which a certain abstraction level can be considered as an optimization process. In general, EC techniques are formed of two principal components: determinist rules and stochastic operations. Combining both elements, each EC method proposes a particular solution to the exploration–exploitation balance. Under such conditions, each EC technique holds distinctive characteristics that appropriately fulfill the requirements of particular problems.

Several EC methods have been proposed in the literature. According to their applications [14], the algorithms Particle Swarm Optimization (PSO) [29], the Artificial Bee Colony (ABC) algorithm [30], the Differential Evolution (DE) method [31], the Harmony Search (HS) strategy [32], the Gravitational Search Algorithm (GSA) [33] and the Flower Pollination Algorithm (FPA) [34] are considered the most popular currently in use. Therefore, they will be used in our comparison study. In this Section, a description of each algorithm used in the comparison is presented.

2.1. Particle Swarm Optimization (PSO)

The PSO is probably the most popular optimization technique; it was introduced by Kennedy in 1995 [29], and is motivated by the action of bird flocking or fish schooling. The algorithm uses a group or swarm of particles as possible solutions $\mathbf{p}^k(\{\mathbf{p}_1^k, \mathbf{p}_2^k, \ldots, \mathbf{p}_N^k\})$ of N particles, which are evolved from an initial point ($k = 0$) to a maximum number of generations ($k = max\_gen$). Each particle $\mathbf{p}_i^k (i \in [1, \ldots, N])$ symbolizes a d-dimensional vector $\{\mathbf{p}_{i,1}^k, \mathbf{p}_{i,2}^k, \ldots, \mathbf{p}_{i,d}^k\}$ where each dimension is a decision variable of the treated problem. After initializing the particles randomly through the search space, each particle $\mathbf{p}_i^k$ is evaluated by the fitness function $f(\mathbf{p}_i^k)$, the best value of fitness during the evolution process is stored in a vector $\mathbf{g}(g_1, g_2, \ldots, g_d)$ with their respective positions $\mathbf{p}_i^*(\mathbf{p}_{i,1}^*, \mathbf{p}_{i,2}^*, \ldots, \mathbf{p}_{i,d}^*)$.

For practical proposes, the PSO implemented has been proposed by Lin in [35], which compute a new position $\mathbf{p}_i^{k+1}$ of each particle $\mathbf{p}_i^k$ as follows:

$$\begin{gathered} v_{i,j}^{k+1} = \omega \cdot v_{i,j}^k + c_1 \cdot r_1 \cdot (p_{i,j}^* - p_{i,j}^k) + c_2 \cdot r_2 \cdot (g_j - p_{i,j}^k), \\ p_{i,j}^{k+1} = p_{i,j}^k + v_{i,j}^{k+1} \end{gathered} \tag{1}$$

where ω is the inertia weight, and rules the current and the updated velocity, c_1 and c_2 are positive numbers that control the acceleration of each individual to the positions $\mathbf{g}$ and $\mathbf{p}_i^k$, respectively. r_1 and r_2 are random numbers in the interval [0, 1].

2.2. Artificial Bee Colony (ABC)

The ABC was introduced by Karaboga [30]. It is based on the intelligent foraging behavior of honeybee swarms. In the ABC implementation, the algorithm generates a population $\mathbf{X}^k\left(\left\{\mathbf{x}_1^k, \mathbf{x}_2^k, \ldots, \mathbf{x}_N^k\right\}\right)$ of N possible food locations, which are evolved from the initial point ($k = 0$) to the maximum number of generations ($k = max\_gen$). Each food position $\mathbf{X}_i^k (i \in [1, \ldots, N])$ symbolizes a d-dimensional vector $\left\{\mathbf{x}_{i,1}^k, \mathbf{x}_{i,2}^k, \ldots, \mathbf{x}_{i,d}^k\right\}$ where each dimension is a decision variable of the treated problem. The initial feasible food locations (solutions) are evaluated by an objective function to determinate which food location represents a satisfactory solution (nectar-amount) or not. Once the values from a cost function are computed, the candidate solution $\mathbf{x}_i^k$ evolves by different ABC operators (honeybee types), where each food location $\mathbf{x}_i^k$ generates a new food source $\mathbf{t}_i$ and is computed as follows:

$$\mathbf{t}_i = \mathbf{x}_i^k + \phi(\mathbf{x}_i^k - \mathbf{x}_r^k), \ i, r \in (1, 2, \ldots N) \tag{2}$$

where $\mathbf{x}_i^k$ is a food location randomly selected, which satisfies the condition $r \neq i$. A factor ϕ is a random number in the interval $[-1, 1]$. When a new solution $\mathbf{t}_i$ is produced, a probability associated with the fitness value of a particular solution $\text{fit}(\mathbf{t}_i)$ is calculated. The quality of the solution determined by the fitness function can be assigned to the solution $\mathbf{x}_i^k$ using the following expression:

$$\text{fit}(\mathbf{x}_i^k) = \begin{cases} \frac{1}{1+f(\mathbf{x}_i^k)} & \text{if } f(\mathbf{x}_i^k) \geq 0 \\ 1 + \left|f(\mathbf{x}_i^k)\right| & \text{if } f(\mathbf{x}_i^k) < 0 \end{cases} \tag{3}$$

where $f(\cdot)$ describes the cost function to be optimized. When the fitness values of the complete population are computed, a greedy selection between $\mathbf{t}_i$ and $\mathbf{x}_i^k$ is employed. If the quality of $\mathbf{t}_i$ is better than $\mathbf{x}_i^k$ then the solution $\mathbf{x}_i^k$ is changed by $\mathbf{t}_i$; otherwise, the original solution remains.

If a candidate solution i (food source) cannot improve within a set trial value named as "*limit*", the candidate solution is expected to be discarded and the related bee becomes a scout. A scout explores the search space without any information. To verify if a candidate solution reached the fixed "*limit*", a counter is assigned to every food source i. The counter is incremented, resulting in a bee-operator failing to improve the food source's quality.

2.3. Differential Evolution (DE)

The DE was developed by Storn and Price [31], where the feasible solution is defined as a vector and is represented as follow:

$$\mathbf{x}_i^t = (\mathbf{x}_{1,i}^t, \mathbf{x}_{2,i}^t, \ldots, \mathbf{x}_{d,i}^t), \ i = 1, 2, \ldots n \tag{4}$$

where $\mathbf{x}_i^t$ is the i-th vector at iteration t. The standard DE procedure has three main stages: *(A) mutation*, *(B) crossover* and *(C) selection* which are described below:

(A) Mutation. This process is applied to a given vector $\mathbf{x}_i^t$ at the t generation. Firstly, three different vectors are randomly selected from the entire population: $\mathbf{x}_p$, $\mathbf{x}_q$ and $\mathbf{x}_r$, then, a new vector (candidate solution) is generated as follow:

$$\mathbf{v}_i^{t+1} = \mathbf{x}_p^t + F \cdot (\mathbf{x}_q^t - \mathbf{x}_r^t) \tag{5}$$

where $F \in [0, 1]$ is a parameter related to the differential weight and is adjusted depending on the problem to be optimized.

(B) Crossover. This operator controlled by the crossover rate $C_r \in [0, 1]$. The $C_r \in [0, 1]$ helps to modify only a few vectors of the population and it is applied on each dimension (or element) of a

solution. The crossover begins by generating a random number $r_i \in [0,1]$ and then, the j-th component $\mathbf{v}_i$ is handled as follow:

$$\mathbf{u}_{j,i}^{t+1} = \begin{cases} \mathbf{v}_{j,i} & r_i \leq C_r \\ \mathbf{x}_{j,i}^{t} & \text{otherwise} \end{cases} \quad , j = 1,2,\ldots,d; i = 1,2,\ldots,n \tag{6}$$

(C) Selection. The selection process is essentially the same as that used in Genetic Algorithms (GA) [16]. The new solution is evaluated by the fitness function; if the new candidate solution is better than the current, then it is replaced. We can consider the following criteria:

$$\mathbf{x}_i^{t+1} = \begin{cases} \mathbf{u}_i^{t+1} & \text{if } f(\mathbf{u}_i^{t+1}) \leq f(\mathbf{x}_i^t) \\ \mathbf{x}_i^t & \text{otherwise} \end{cases} \tag{7}$$

2.4. Harmony Search (HS)

The HS was proposed by Geem [32]; this algorithm is based on music. It was inspired by the observation that music aims are searching for the perfect state of harmony by improvisation. The analogy in optimization is the process of searching the optimal value of a problem. When a musician is improvising, he follows three rules to obtain a good melody: plays any well-known portion of music precisely from his memory, plays something similar to a famous piece and composes new music or plays random notes. In the HS operation those rules become; *(a) Harmony Memory (HM) consideration, (b) pitch adjustment* and *(c) initialize a random solution.*

(a) Harmony Memory (HM) Consideration. This stage will guarantee that the best harmonies will be transferred to a new harmony memory. To apply the HM more efficiently, it is assigned a parameter that helps to select the elements of HM. Such a parameter is the Harmony Memory Consideration Rate (*HMCR*) and is defined as *HMCR* $\in [0,1]$. If the value of the *HMCR* is low, only a few good harmonies are chosen and it can slowly converge. If such a value is notably high, most of the harmonies are employed in *HM*, and then additional harmonies are not used in the exploration stage.

(b) Pitch Adjustment. Each element achieved by memory consideration is additionally analyzed to decide whether it should be pitch-modified. For this action, the Pitch-Adjusting Rate (*PAR*) is described to select the recurrence of the adjustment and the Bandwidth value (*BW*) to regulate the local search of the chosen elements of the HM. Therefore, the pitch adjust determination is computed as:

$$x_{new} = \begin{cases} x_{new} \leftarrow x_{new} \pm \text{rand}(0,1) \cdot BW & \text{with probability } PAR \\ x_{new} & \text{with probability } (1\text{-}PAR) \end{cases} \tag{8}$$

Pitch modification is responsible for producing new likely harmonies by lightly adjusting original value positions. This action can be considered as the mutation rule in evolutionary techniques. Consequently, the decision variable is either modified by a random number in the range [0, *BW*] or remains with no changes.

(c) Initialize a random solution. This operation tries to increase the diversity of the solutions. As is mentioned in a), if a new solution is not taken from the harmony memory, it must be created randomly in the search space. It helps to explore new areas avoiding the local optimal. This process is described as follows:

$$x_{new} = \begin{cases} x_i \in \{x_1, x_2, \ldots, x_{HMS}\} & \text{with probability } HMCR \\ l + (u - l \cdot \text{rand}(0,1) & \text{with probability } (1\text{-}HMCR) \end{cases} \tag{9}$$

where *HMS* is the size of the HM, x_i is an element chosen from the HM, and u and l are the upper and lower bounds. Finally, rand is a random value uniformly distributed between 0 and 1.

2.5. Gravitational Search Algorithm (GSA)

The GSA is an evolutionary technique inspired by the Newtonian laws of gravity [33]. The GSA has been used widely in different engineering fields due to its capabilities for determining competitive solutions [36]. In GSA each mass (solution) represents a d-dimensional vector $\mathbf{x}_i(t) = (x_i^1, \ldots, x_i^d)$; $(i \in 1, \ldots, N)$, at time t, the mass i acts to the mass j in the variable h $(h \in 1, \ldots, d)$ with force defined as:

$$F_{ij}^h(t) = G(t) \cdot \frac{Mp_i(t) \cdot Ma_j(t)}{R_{ij}(t) + \varepsilon} \cdot (x_j^h(t) - x_i^h(t)) \tag{10}$$

where Ma_j is the active gravitational mass associated with the solution j, Mp_i represents the passive gravitational mass of the element i, $G(t)$ is a gravitational constant at the time t, ε is a small constant, and R_{ij} is the distance between the elements i and j. The $G(t)$ function is adjusted through the evolution process for balancing the exploration and exploitation by the modification of attraction forces between solutions. The total force operating over the solution i is represented as follows:

$$F_i^h(t) = \sum_{j=1, j \neq i}^{N} F_{ij}^h(t) \tag{11}$$

The acceleration at time t of the candidate solution is estimated as below:

$$a_i^h(t) = \frac{F_i^h(t)}{Mn_i(t)} \tag{12}$$

where Mn_i describes the inertial mass of the solution i. The new location of each individual i is calculated as:

$$\begin{aligned} x_i^h(t+1) &= x_i^h(t) + v_i^h(t+1) \\ v_i^h(t+1) &= \mathrm{rand}(\cdot) \cdot v_i^h(t) + a_i^h(t) \end{aligned} \tag{13}$$

The inertial and gravitational masses of each individual are estimated directly by its fitness function value. Consequently, the inertial and gravitational masses are updated by the following expressions:

$$Ma_i = Mp_i = M_{ii} = M_i \tag{14}$$

$$m_i(t) = \frac{f(\mathbf{x}_i(t)) - \mathrm{worst}(t)}{\mathrm{best}(t) - \mathrm{worst}(t)} \tag{15}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{16}$$

where $f(\cdot)$ describes the cost function that determines the quality of the solutions.

2.6. Flower Pollination Algorithm (FPA)

The FPA was introduced by Xin-She Yang in 2012 [34]. FPA stimulates the flower pollination process of plants. In the biological approach, this process is key for the reproduction of a plant and is associated with the transfer of pollen. This transfer process is performed by pollinator agents (insects, birds, the wind, etc.). Pollination can be classified into two forms: abiotic and biotic. Most of the plants use biotic pollination, in which the use of external pollinator is necessary; this process is known as cross-pollination. On the other hand, abiotic pollination is known as self-pollination; it does not require the intervention of any external pollinator. Contrarily to the biological approach, the FPA for simplicity contemplates just one flower that provides one pollen gamete. In the FPA, a solution $\mathbf{f}_i$ corresponds to a flower or pollen gamete. In the implementation, the algorithm generates a population $\mathbf{F}^k = \left\{\mathbf{f}_1^k, \mathbf{f}_2^k, \ldots, \mathbf{f}_m^k\right\}$ of m flowers, where each flower is represented by a d-dimensional vector $\left\{f_{i,1}^{k}, f_{i,2}^{k}, \ldots, f_{i,d}^{k}\right\}$. The dimension vector d corresponds to the number of decision variables of the problem to treat.

(A) Lévy flights. One of the most powerful operators in flower pollination algorithm is the *Lévy* Flights [37], which generates a random walk. The walk length is defined by a heavy-tailed probability distribution. The random walk is generated as follows:

$$s_i = \frac{\mathbf{u}}{|\mathbf{v}|^{1/\beta}} \tag{17}$$

where $\mathbf{u} = \{u_1, \ldots, u_d\}$ and $\mathbf{v} = \{v_1, \ldots, v_d\}$ are vectors of dimensionality d, and the value $\beta = 3/2$. The elements of the vectors **u** and **v** are computed using normal distributions as below:

$$\begin{aligned} u &\sim N(0, \sigma_u^2), & v &\sim N(0, \sigma_v^2), \\ \sigma_u &= \left(\frac{\Gamma(1+\beta)\cdot sin(\pi\cdot\beta/2)}{\Gamma((1+\beta)/2)\cdot\beta\cdot 2^{(\beta-1)/2}}\right)^{1/\beta}, & \sigma_v &= 1, \end{aligned} \tag{18}$$

where $\Gamma(\cdot)$ represents the Gamma distribution.

(B) Local pollination. The local pollination is described in Equation (19), where $\mathbf{f}_j^k$ and $\mathbf{f}_h^k$ are pollens from two different flowers of the same plant species.

$$\mathbf{f}_i^{k+1} = \mathbf{f}_i^k + \varepsilon(\mathbf{f}_i^k - \mathbf{f}_h^k) \tag{19}$$

where ε is a scalar random value within [0, 1].

(C) Global pollination. The original position $\mathbf{f}_i^k$ is replaced with the new position $\mathbf{f}_i^{k+1}$ according to the following equation:

$$\mathbf{f}_i^{k+1} = \mathbf{f}_i^k + s_i(\mathbf{g}^* - \mathbf{f}_i^k) \tag{20}$$

where s_i is a random scalar within [0, 1] while $\mathbf{g}^*$ is the best candidate solution seen so-far.

3. 2D-IIR Filter Design Process as an Optimization Problem

This section formulates the process of the 2D-IIR filter design as an optimization problem. In this paper, the zero-phase IIR filter design is considered. A zero-phase filter is a particular example of linear phase filters where the phase slope is zero. This fact suggests that the impulse response is always even ($h(n) = h(-n)$) [8]. The structure of a 2D-IIR filter is presented in Equation (21) [15,16,38–40].

$$H(z_1, z_2) = H_0 \cdot \frac{\sum\limits_{i=0}^{K}\sum\limits_{j=0}^{K} a_{ij} z_1^i z_2^i}{\prod\limits_{k=1}^{K}(1 + b_k z_1 + c_k z_2 + d_k z_1 z_2)} \tag{21}$$

where a_{ij}, b_k, c_k, d_k are the filter coefficients, H_0 is an intensity constant while K symbolizes the filter order. In case of K = 2, the model of Equation (21) corresponds to a 2nd order filter whereas a higher value (K > 3) specifies a high order filter. In Equation (21), it is usually configured $a_{00} = 1$ (by normalizing a_{ij} to the value of a_{00}). z_1 and z_2 represent complex variables in the z-transform domain. However, for convenience, the Fourier frequency domain is used, where z_1 and z_2 are defined as follows:

$$z_1 = e^{j\omega_1}, z_2 = e^{j\omega_2} \tag{22}$$

where j is the imaginary constant and $\omega_1, \omega_2 \in [0, \pi]$ are frequency variables. With the substitution of Equation (22) in Equation (21), $H(z_1, z_2)$ changes to $M(\omega_1, \omega_2)$.

The objective of filter design is to approximate the response of a determined 2D-IIR filter structure to the desired behavior $M_d(\omega_1, \omega_2)$ provided by the user requirements. Therefore, the design process consists of finding the optimal parameters $M(\omega_1, \omega_2)$ that present the best possible resemblance with the desired response $M_d(\omega_1, \omega_2)$. Both responses $M(\omega_1, \omega_2)$ and $M_d(\omega_1, \omega_2)$ are discretely computed at $N_1 \times N_2$ points.

To guide the search process, an objective function J that evaluates the similarity between the candidate filter $M(\omega_1, \omega_2)$ and the desired response $M_d(\omega_1, \omega_2)$ must be defined. This similarity can be defined as the difference that exists between both responses, such as follows:

$$J(a_{ij}, b_k, c_k, d_k, H_0) = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} \left[\left| M(\omega_1, \omega_2) \right| - M_d(\omega_1, \omega_2) \right]^2 \tag{23}$$

where $\omega_1 = (\pi/N_1) \cdot n_1$, $\omega_2 = (\pi/N_2) \cdot n_2$. For the sake of clarity, the Equation (23) can be rewritten as:

$$J = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} \left[\left| M\left\{ \frac{\pi n_1}{N_1}, \frac{\pi n_2}{N_2} \right\} \right| - M_d\left\{ \frac{\pi n_1}{N_1}, \frac{\pi n_2}{N_2} \right\} \right]^2 \tag{24}$$

Figure 1 illustrates the filter design process. In the learning schema, the difference between the computed ($y(t)$) and desired ($d(t)$) filter response is used to guide the evolution process.

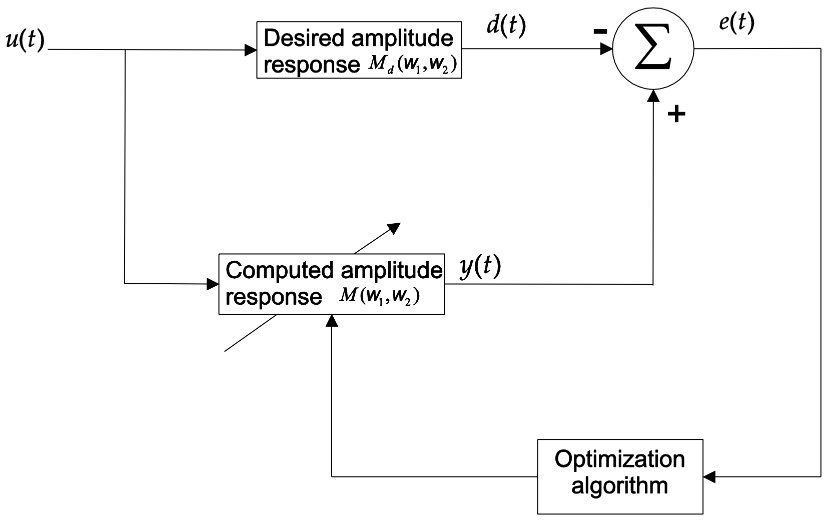
$u(t)$
Desired amplitude response $M_d(w_1,w_2)$
$d(t)$
-
Σ
$e(t)$
+
Computed amplitude response $M(w_1,w_2)$
$y(t)$
Optimization algorithm

Figure 1. Scheme of two-dimensional Infinite Impulse Response (2D-IIR) filter design based on the desired amplitude response.

One main difficulty of 2D IIR filters is the tendency to produce unstable transfer functions. Therefore, in the evaluation of Equation (24), it is necessary to verify if the poles of the transfer function are located within the unitary circle of the z-plane. To validate this state, several conditions must be fulfilled, which, for Equation (25) can be expressed as follows [15,16,38]:

$$|b_k + c_k| - 1 < d_k < 1 - |b_k - c_k|, \; k = 1, 2, \ldots, K \tag{25}$$

Under such circumstances, the filter design process can be translated into an optimization problem with constraints.

Comparison Setting

In order to conduct the comparison, it is necessary to define the experiment and its main characteristics. In the test, the filter design problem has been selected according to [16,38]. The idea with this selection is to conserve compatibility with the results produced by such studies. In the experiment, the 2D-IIR filter structure is formulated as follows:

$$H(z_1, z_2) = H_0 \cdot \frac{a_{00} + a_{01}z_2 + a_{02}z_2^2 + a_{10}z_1 + a_{20}z_1^2 + a_{11}z_1z_2 + a_{12}z_1z_2^2 + a_{21}z_1^2z_2 + a_{22}z_1^2z_2^2}{(1 + b_1z_1 + c_1z_2 + d_1z_1z_2)(1 + b_2z_1 + c_2z_2 + d_2z_1z_2)} \tag{26}$$

For the sake of computational ease, the variables z_1 and z_2 are replaced by $f_{xy} = \cos(x\omega_1 + y\omega_2)$ and $g_{xy} = \sin(x\omega_1 + y\omega_2)$, $[x, y] = [0, 1, 2]$. This simplification has been proposed by Swagatam and Konar in [15]. The objective is to make the implementation in digital filters easier. Therefore, Equation (26) is rewritten as:

$$\begin{aligned} M(\omega_1, \omega_2) = H_0\Big[& \tfrac{a_{00}+a_{01}f_{01}+a_{02}f_{02}+a_{10}f_{10}+a_{20}f_{20}+a_{11}f_{11}+a_{12}f_{12}+a_{21}f_{21}+a_{22}f_{22}}{V} \dots \\ & \dots - j\tfrac{a_{00}+a_{01}g_{01}+a_{02}g_{02}+a_{10}g_{10}+a_{20}g_{20}+a_{11}g_{11}+a_{12}g_{12}+a_{21}g_{21}+a_{22}g_{22}}{V}\Big] \end{aligned} \tag{27}$$

where V is defined by:

$$\begin{aligned} V = \ & [(1 + b_1f_{10} + c_1f_{01} + d_1f_{11}) - j(b_1g_{10} + c_1g_{01} + d_1g_{11}] \quad \cdot \\ & [(1 + b_1f_{10} + c_1f_{01} + d_1f_{11}) - j(b_1g_{10} + c_1g_{01} + d_1g_{11}] \end{aligned} \tag{28}$$

Under such conditions, the Equation (27) can be compacted as follows:

$$M(\omega_1, \omega_2) = H_0 \cdot \frac{N_R - jN_I}{(D_{1R} - jD_{1R})(D_{2R} - jD_{2I})} \tag{29}$$

where the elements of Equation (29) are defined as:

$$\begin{aligned} N_R &= a_{00} + a_{01}f_{01} + a_{02}f_{02} + a_{10}f_{10} + a_{20}f_{20} + a_{11}f_{11} + a_{12}f_{12} + a_{21}f_{21} + a_{22}f_{22} \\ N_I &= a_{00} + a_{01}g_{01} + a_{02}g_{02} + a_{10}g_{10} + a_{20}g_{20} + a_{11}g_{11} + a_{12}g_{12} + a_{21}g_{21} + a_{22}g_{22} \\ D_{1R} &= 1 + b_1f_{10} + c_1f_{01} + d_1f_{11} \\ D_{1I} &= 1 + b_1g_{10} + c_1g_{01} + d_1g_{11} \\ D_{2R} &= 1 + b_2f_{10} + c_2f_{01} + d_2f_{11} \\ D_{2I} &= 1 + b_2g_{10} + c_2g_{01} + d_2g_{11} \end{aligned} \tag{30}$$

Considering Equations (29) and (30), the magnitude $M(\omega_1, \omega_2)$ of the filter is computed as:

$$\big|M(\omega_1, \omega_2)\big| = H_0 \sqrt{\frac{N_R^2 - N_I^2}{(D_{1R}^2 - D_{1I}^2)(D_{2R}^2 - D_{2I}^2)}} \tag{31}$$

In the test, the desired response $M_d(\omega_1, \omega_2)$ is defined as a circular symmetric low-pass response [10,15,16,38–40], which is given by:

$$M_d(\omega_1, \omega_2) = \begin{cases} 1 & \text{if } \sqrt{\omega_1^2 + \omega_2^2} \le 0.08\pi \\ 0.5 & \text{if } 0.08\pi \le \sqrt{\omega_1^2 + \omega_2^2} \le 0.12\pi \\ 0 & \text{otherwise} \end{cases} \tag{32}$$

For approximating both responses $M(\omega_1, \omega_2)$ and $M_d(\omega_1, \omega_2)$, it is used a matrix of $N_1 \times N_2$ points, where N_1 and N_2 are set to 50.

In order to ensure the stability of the filter for Equation (26), the candidate solution must fulfill the following four constraints:

$$\begin{aligned} -(1+d_k) < (b_k + c_k) < (1+d_k) \\ -(1-d_k) < (b_k + c_k) < (1-d_k) \\ (1+d_k) > 0 \\ (1-d_k) > 0 \end{aligned} \tag{33}$$

Under these conditions, the filter design process can be interpreted as a constrained optimization problem. This optimization task can be formulated as follows:

$$\begin{aligned} \min J(a_{ij}, b_k, c_k, d_k, H_0) = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} \left[\left| M\left\{\frac{\pi n_1}{50}, \frac{\pi n_2}{50}\right\}\right| - M_d\left\{\frac{\pi n_1}{50}, \frac{\pi n_2}{50}\right\}\right]^2 \\ \text{subject to} - (1+d_k) < (b_k + c_k) < (1+d_k) \\ -(1-d_k) < (b_k + c_k) < (1-d_k) \\ (1+d_k) > 0 \quad k = 1, 2, \ldots, K \\ (1-d_k) > 0, \end{aligned} \tag{34}$$

In order to use an EC algorithm for the problem formulated in Equation (34), it is necessary to represent each candidate solution as one location in the multi-dimensional search space. A candidate solution to Equation (34), considering a 2nd order filter $K = 2$, is represented by the filter parameters concentered in the 15-dimensional vector $\mathbf{x}_i$:

$$\mathbf{x}_i = [a_{01}, a_{02}, a_{10}, a_{11}, a_{12}, a_{20}, a_{21}, a_{22}, b_1, c_1, d_1, b_2, c_2, d_2, H_0]^T \tag{35}$$

Under such conditions, the optimal solution is represented by the solution $\mathbf{x}_i$ whose parameters minimize Equation (34), while they simultaneously hold the requirements imposed by its respective constraints. The optimal solution corresponds to the 2D-IIR filter whose response $M(\omega_1, \omega_2)$ is the most similar to $M_d(\omega_1, \omega_2)$.

For constraint handling, all EC methods incorporate a penalty factor [41] whose value corresponds to the violated constraints. The main purpose is to include a tendency term in the original cost function to penalize constraint violations for determining only feasible solutions

4. Experimental Results

In this section, the comparison results are presented. In the experiments, we have used the 2D-IIR filter design problem formulated in Section 3. In the tests, it is compared the algorithms: Particle Swarm Optimization (PSO) [29], the Artificial Bee Colony (ABC) algorithm [30], the Differential Evolution (DE) method [31], the Harmony Search (HS) strategy [32], the Gravitational Search Algorithm (GSA) [33] and the Flower Pollination Algorithm (FPA) [34]. These methods are considered as the most popular EC algorithms currently in use [11]. In comparison, the parameters of all algorithms are set as follows:

1. PSO, parameters $c_1 = 2$, $c_2 = 2$ and weights factors were set $w_{\max} = 0.9$, and $w_{\min} = 0.4$ [42].
2. ABC, the parameters are given in [43], *limit = 90.*
3. HS, corresponding to [44] the parameters were set *HMCR* = 0.9, and *PAR* = 0.3.
4. DE, as is reported in [31] the parameters were set *CR* = 0.9 and $F = 0.5$.
5. GSA, $G_0 = 100$ and $\alpha = 20$, according to [33].
6. FPA, the probabilistic global pollination factor, was set $p = 0.8$ [34].

All selected techniques have been set with a population of 25 elements, while the stop criteria is configured to 3000 iterations. In contrast with most of the related works that usually employ less than 500 iterations [16,43,45], in this manuscript, the maximum iterations number is set as 3000 in order to perform a convergence analysis, which is described in Section 4.2. In this work, the produced filters have not been implemented in the hardware of fixed arithmetic. For this reason, an analysis of the quantization error has not been considered. All conclusions have been obtained through simulated experiments.

4.1. Accuracy Comparison

In the first experiment, the design of a 2nd order filter from Equations (32) and (34) is considered. Table 1 reports the J values (Equation (34)) generated by the experimentation. They evaluate the differences between the ideal filter response and those produced by the EC techniques. A lower value of J involves a better filter performance. Table 1 reports minimal ($J_{\min}$), maximal ($J_{\max}$) and mean ($\bar{J}$) values of J along with their standard deviations (σ) obtained for each algorithm during 250 executions. The first three indexes consider the accuracy of each solution, whereas the last one assesses its robustness. The results show that the ABC and FPA algorithms present the best performance in terms of accuracy and robustness. On the other hand, Table 1 also indicates that the ABC algorithm maintains the lowest standard deviation, which means that it can find a similar solution with a very small variation. The rest of the methods obtain different performance levels. In Table 1, the highlighted values represent the best values.

Table 1. Minimum ($J_{\min}$) and maximum ($J_{\max}$) values of the objective function, mean ($\bar{J}$) and standard deviation (σ) for the results obtained by each algorithm over 250 executions.

	PSO	ABC	DE	HS	GSA	FPA
$J_{\min}$	2.6971	3.0655	2.6550	3.4276	3.2668	**2.3030**
$J_{\max}$	16.1331	4.0599	15.1968	8.5494	16.9647	**3.3276**
$\bar{J}$	6.5976	3.5173	5.2582	4.5836	6.3707	**2.5838**
σ	3.4537	**0.1736**	2.0377	0.5925	4.1108	0.2759

Table 2 reports the filter parameters of the best filter obtained by each algorithm PSO, ABC, DE, HS, GSA and FPA after 250 individual runs, where each execution contemplates 3000 iterations. They correspond to the parameters that produce minimal J value ($J_{\min}$). In Table 2, the highlighted values represent the best-found parameters among all techniques.

Table 2. Coefficients for 2D-IIR filter design obtained by Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Differential Evolution (DE), Harmony Search (HS), Gravitational Search Algorithm (GSA) and Flower Pollination Algorithm (FPA) algorithms.

2D-IIR Filter Coefficients	PSO	ABC	DE	HS	GSA	FPA
$\mathbf{a_{01}}$	0.1182	0.3217	−0.4785	−0.7521	0.2978	**−0.0388**
$\mathbf{a_{02}}$	0.0149	0.3109	−0.9983	−0.2398	−0.0654	**−0.6939**
$\mathbf{a_{10}}$	0.6393	−0.5132	−1.0000	−0.3936	−0.6860	**−0.5417**
$\mathbf{a_{11}}$	0.2938	−0.9392	−0.8839	0.1290	0.6559	**0.0738**
$\mathbf{a_{12}}$	0.3514	−0.1289	0.9422	−0.3705	−0.8044	**−0.4146**
$\mathbf{a_{20}}$	0.3459	0.7601	0.7540	−0.9324	−0.1292	**−0.2994**
$\mathbf{a_{21}}$	0.9404	−0.0376	−0.8596	0.1805	0.3098	**−0.8339**
$\mathbf{a_{22}}$	−0.9137	0.8856	−0.4131	0.2330	0.2307	**0.6927**
$\mathbf{b_1}$	−0.2290	0.3588	−0.9353	−0.3858	−0.2743	**−0.2251**
$\mathbf{b_2}$	−0.9081	−0.9238	0.5207	−0.6663	−0.8617	**−0.9241**
$\mathbf{c_1}$	−0.8491	−1.0000	−0.9046	−0.3996	−0.8175	**−0.1543**
$\mathbf{c_2}$	−0.7313	−0.6050	−0.8622	−0.5995	−0.5501	**−0.9438**
$\mathbf{d_1}$	0.1360	−0.4827	0.8458	−0.2974	0.1367	**−0.8157**
$\mathbf{d_2}$	0.6545	0.5521	−0.9204	0.2159	0.4369	**0.8906**
$\mathbf{H_0}$	0.0004	0.0019	0.0009	0.0046	0.0013	**0.0038**

Figure 2 graphically shows the filter responses produced by each algorithm. Each graph (b–g) presents the filter response that is produced according to the parameters exhibited in Table 2. After an analysis of Figure 2, it is clear that algorithms PSO and HS produce the worst performance since prominent ripples are appreciated in their responses. GSA and DE present lower ripples than PSO and

HS. Under such conditions, they maintain better performance. Finally, ABC and FPA obtain the best filter responses. In both cases, ABC keeps very small ripples, whereas FPA practically eliminates them.

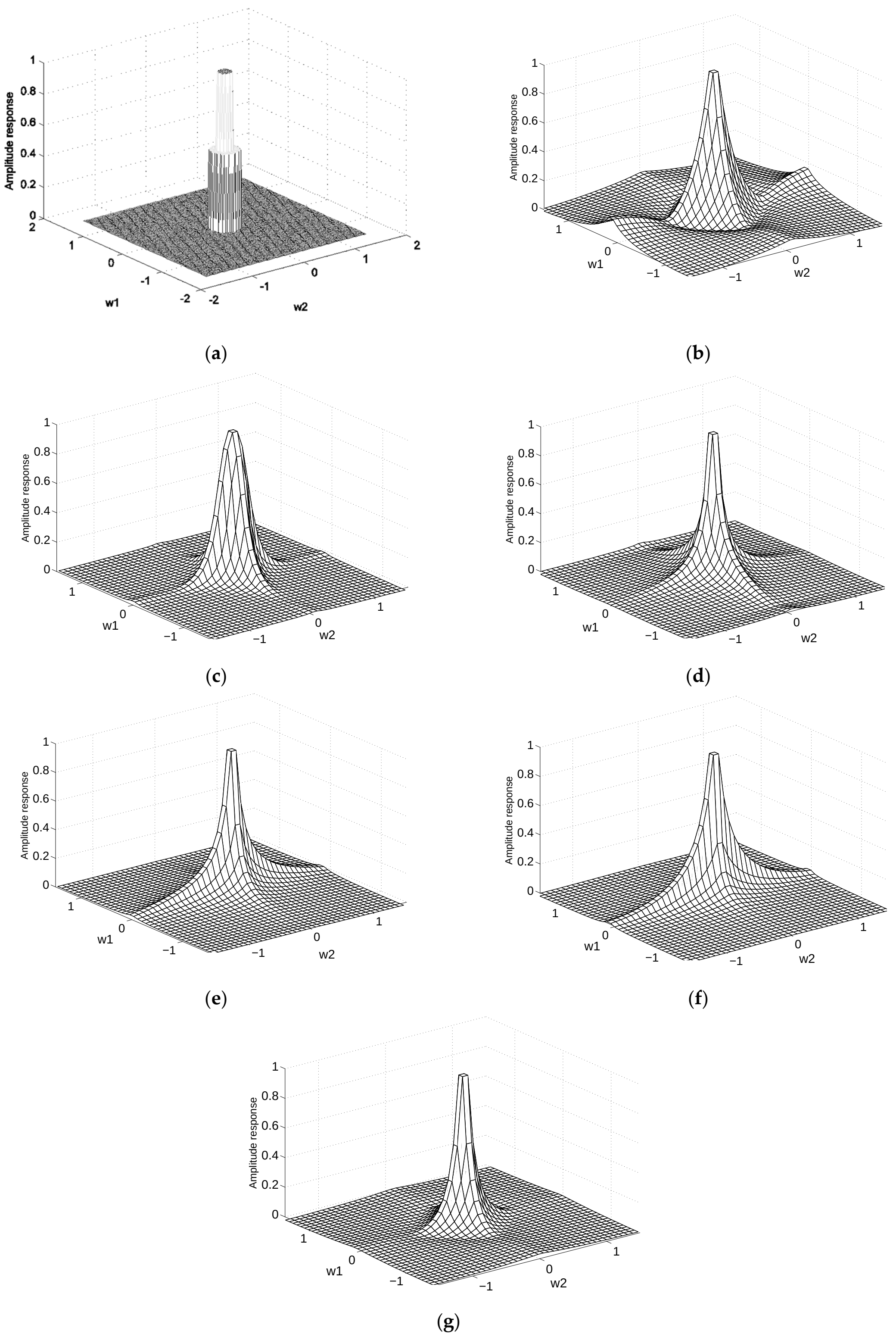

Figure 2. (**a**) Desired amplitude response, (**b**) Amplitude response calculated by PSO, (**c**) Amplitude response calculated by ABC, (**d**) Amplitude response calculated by DE, (**e**) Amplitude response calculated by HS, (**f**) Amplitude response calculated by GSA, (**g**) Amplitude response calculated by FPA.

4.2. Convergence Analysis

The comparative study of the final $\bar{J}$ values cannot effectively represent the searching capabilities of the EC techniques. Hence, in Figure 3, a convergence experiment on the seven methods is carried out. The main idea with this test is evaluating the velocity with which a compared technique approaches the optimum solution. In the test, the performance of each employed technique is examined over the minimization of the filter design. The graphs of Figure 3 represent the median final result considering 250 executions, where each execution contemplates 3000 iterations. In the Figure 3, it is clear that PSO and GSA converge faster than the other algorithms, but prematurely.

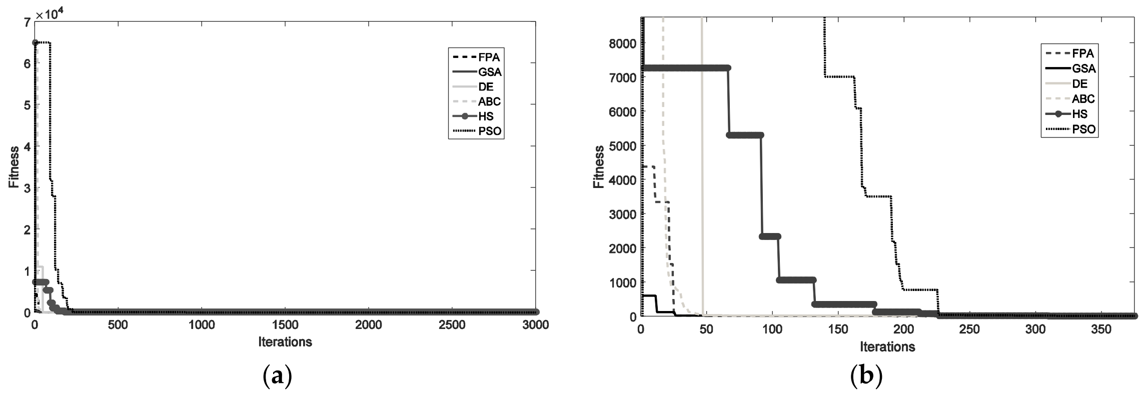

Figure 3. Evolution of the objective function values for FPA, GSA, DE, HS, ABC, and PSO. (**a**) Graphs over 3000 iterations and (**b**) zoom over 400 iterations.

4.3. Computational Cost

In this experiment, the objective is to measure the number of function evaluations (*NFE*) and the computing time (*CT*) spent by each method required to calculate the parameters of a certain filter. All experiments consider the filter design formulated by Equations (32) and (34), which correspond to the parameters exhibited in Table 2.

The index *NFE* refers to the number of required function evaluations from which improvement is not detected in the objective function $\bar{J}$ (Equation (34)). CT represents the elapsed time in seconds consumed by each algorithm corresponding to the *NFE*. The test has been executed in Matlab running on a Computer Nitro, Core i7. Table 3 shows the averaged measurements as they are obtained from 20 independent executions. A close inspection of Table 3 demonstrates that PSO and HS are the fastest methods. Nevertheless, this apparent velocity corresponds to premature convergence. ABC, DE and GSA are the slowest algorithms. On the other hand, FPA represents the scheme which obtains the best trade-off between velocity and accuracy.

Table 3. Averaged measurements corresponding to the number of function evaluations (*NFE*) and the computing time (*CT*) spent by each method obtained from 20 independent executions.

Index	PSO	ABC	DE	HS	GSA	FPA
NFE	48,543.12	58,321.09	57,322.24	49,322.74	56,654.57	51,342.18
CT	8.25 s	12.54 s	13.02 s	9.10 s	11.82 s	10.12 s

4.4. Comparison with Different Bandwidth Sizes

In order to extend the comparison analysis, the performance of each method in the design of a low pass filter with different bandwidth sizes is evaluated. For this experiment, two particular cases are considered. In the first case, the objective is to design a filter with an ideal response defined by:

$$M_d^1(\omega_1,\omega_2) = \begin{cases} 1 & \text{if } \sqrt{\omega_1^2+\omega_2^2} \le 0.16\pi \\ 0.5 & \text{if } 0.16\pi \le \sqrt{\omega_1^2+\omega_2^2} \le 0.24\pi \\ 0 & \text{otherwise} \end{cases} \tag{36}$$

The second case considers the design of a filter with an ideal response defined as follows:

$$M_d^2(\omega_1,\omega_2) = \begin{cases} 1 & \text{if } \sqrt{\omega_1^2+\omega_2^2} \le 0.32\pi \\ 0.5 & \text{if } 0.32\pi \le \sqrt{\omega_1^2+\omega_2^2} \le 0.48\pi \\ 0 & \text{otherwise} \end{cases} \tag{37}$$

Assuming J^1 and J^2 as the objective functions corresponding to each ideal response M_d^1 and M_d^2, Table 4 presents comparison results. They evaluate the differences between the ideal filter response and those produced by the designed filters through EC techniques for different bandwidth sizes. The Table 4 reports minimal (J^p_{min}), maximal (J^p_{max}) and mean ($\bar{J}^p$) values of J along with their standard deviations (σ_p) obtained for each algorithm for every case p = 1.2. The obtained values are calculated from 250 different executions of each method. A close inspection of Table 4 shows that the FPA scheme produces the best performance for both cases J^1 and J^2.

Table 4. Minimum (J^p_{min}) and maximum (J^p_{max}) values of the objective function, mean ($\bar{J}^p$) and standard deviation (σ_p) for the results obtained by every case p = 1.2.

	PSO	ABC	DE	HS	GSA	FPA
J^1_{min}	15.94	13.01	20.93	23.12	22.67	**12.68**
J^2_{min}	36.54	35.72	18.90	24.33	58.46	**17.52**
J^1_{max}	73.68	16.85	35.84	47.58	87.91	**15.24**
J^2_{max}	155.63	51.91	31.39	41.48	45.15	**26.35**
$\bar{J}^1$	25.40	14.94	25.21	31.19	75.87	**13.95**
$\bar{J}^2$	68.62	44.45	24.65	28.94	49.94	**22.75**
σ_1	14.17	**0.99**	3.14	6.90	17.60	1.54
σ_2	28.81	4.85	3.44	3.75	**3.13**	3.86

4.5. Filter Performance Characteristics

In this subsection, the performance of the filter characteristics is analyzed. The objective is to evaluate the responses of each of the best filters produced by every algorithm PSO, ABC, DE, HS, GSA and FPA (reported in Table 2).

The responses are assessed considering the following indexes: (a) error in the passband, (b) magnitude ripple, (c) stop-band attenuation and (d) the magnitude of poles.

(a) Pass-band error. It evaluates the error in frequency within the passband according to the following formulation:

$$E_{RP}(\omega) = \left|M(e^{j\omega}) - M_d(e^{j\omega})\right| \text{ for } \omega \in [0,\omega_p], \tag{38}$$

where ω_p corresponds to the passband frequency.

(b) Ripple magnitude in the passband. It considers the ripple size considering the model below:

$$E_{MR}(\omega) = \left|M(e^{j\omega})\right| - 1 \text{ for } \omega \in [0,\omega_p]. \tag{39}$$

(c) The attenuation in the stop-band. It describes the response debilitation within the stop-band according to:

$$AT(\omega) = 20\log_{10}\left(\left|M(e^{j\omega})\right|\right) \text{ for } \omega \in [\omega_s, \pi], \tag{40}$$

where ω_s represents the stop-band frequency.

(d) The maximal pole size (*MPs*). This index evaluates the maximal magnitude of poles present in $M(e^{j\omega})$.

The error produced by the frequency response in the passband $E_{RP}(\omega)$ corresponds to the mismatch between the transfer function of the synthesized filter $M(e^{j\omega})$ and the desired transfer function $M_d(e^{j\omega})$, which models the ideal filter response. Under these conditions, a passband error as small as possible characterizes a better filter response. On the other hand, the ripple magnitude in passband $E_{MR}(\omega)$ has to maintain a small enough value to guarantee lower distortions in the passband behavior. Conversely, the attenuation $E_{MR}(\omega)$ should be large with the objective to eliminate undesired components of high frequency. Table 5 presents the performance comparison of the filter characteristics. The results correspond to the responses of each of the best filters produced by each algorithm PSO, ABC, DE, HS, GSA and FPA (reported in Table 2). On close inspection of Table 5, it is evident that the FPA algorithm archives the best performance in error terms in passband, magnitude ripple, stop-band attenuation and magnitude of poles.

Table 5. Performance comparison of the filter characteristics. The results correspond to the responses of each of the best filters produced by every algorithm PSO, ABC, DE, HS, GSA and FPA (reported in Table 2).

Index	PSO	ABC	DE	HS	GSA	FPA
$E_{RP}(\omega)$	0.0881	0.0097	0.0631	0.0287	0.0714	**0.0051**
$E_{MR}(\omega)$	0.1421	0.0918	0.1977	0.1021	0.1391	**0.0272**
$E_{MR}(\omega)$	25.87	34.21	26.42	35.21	29.28	**42.18**
MPs	0.8721	0.9180	0.9182	0.8916	0.9019	**0.8460**

4.6. Statistical Non-Parametrical Analysis

To statistically verify the results provided by the six techniques, a nonparametric statistical significance analysis is implemented. This statistical analysis is known as Wilcoxon's rank-sum test [26,28,46,47]. To make use of this analysis, 35 different executions for each algorithm were needed to prove that there is a significant statistical difference between the $\bar{J}$ values produced by each technique. The Wilcoxon's rank-sum analysis provides estimating result variations between two related techniques. The study is conducted considering a 5% significance value (0.05) over the $\bar{J}$ data. Being the ABC and FPA, the techniques that better perform (according to Table 1), we proceed to compare them with the rest of the methods. Table 6 summarizes the p-values performed by Wilcoxon's analysis for the pair-wise comparison among FPA vs. PSO, FPA vs. ABC, FPA vs. DE, FPA vs. HS, FPA vs. GSA. Table 7 reports the results among ABC vs. PSO, ABC vs. DE, ABC vs. HS, ABC vs. FPA. In the case of PSO, DE, HS and GSA, the non-parametric analysis was not conducted since the results obtained by these techniques are worse than ABC and FPA. In the Wilcoxon study, it is contemplated as the null hypothesis that there is no important difference between the two techniques. On the other hand, the alternative hypothesis is that the difference between the two approaches is important. To simplify the study of Tables 6 and 7, the symbols ▲, ▼, and ► are chosen. ▲ symbolizes that the proposed approach achieves significantly better results than the compared algorithm on a specified function. ▼ expresses that the proposed approach achieves worse results than the compared technique, and ► indicates that the Wilcoxon analysis is not able to distinguish between the results of the proposed approach and the tested technique. From Tables 6 and 7, it is visible that all p-values are less than 0.05, which means that the FPA performs better than the rest of the algorithms.

Table 6. *p*-values of the Wilcoxon test comparing FPA vs. PSO, ABC, DE, HS and GSA.

FPA vs.	PSO	ABC	DE	HS	GSA
Wilcoxon result	1.4118×10^{12}▲	1.8196×10^{12}▲	9.2235×10^{13}▲	6.5455×10^{13}▲	7.1329×10^{13}▲

Table 7. *p*-values of the Wilcoxon test comparing ABC vs. PSO, DE, HS, GSA and FPA.

ABC vs.	PSO	DE	HS	GSA	FPA
Wilcoxon result	4.0756×10^{11}▲	3.4569×10^{9}▲	1.5366×10^{12}▲	2.6216×10^{6}▲	1.8196×10^{12}▲

Since we have run several Wilcoxon's rank-sum tests, the probability of committing an error type 1 (false positive) increases and we can reject the null hypothesis even when it is true. To avoid this problem, a correction for the *p*-values of the Wilcoxon test is adjusted by using the Bonferroni correction method [48,49]. Combining the Wilcoxon test with the Bonferroni method, we can show that there exists a significant statistical difference between FPA and each of the selected algorithms. The results of Bonferroni correction for FPA and ABC are shown in Tables 8 and 9, respectively. The significance value of Bonferroni correction was set to $p = 0.001428$ for the case of FPA comparison, and for the ABC comparison the value of Bonferroni has been set to $p = 0.000084034$. It is important to point out that the absolute values of the correction factors in Bonferroni are not used for comparative proposes. In the Bonferroni test, if the new significance value is marked with 1, it means that this element confirms the result of the Wilcoxon analysis for this particular case.

Table 8. Bonferroni correction for the case of FPA comparison.

FPA vs	PSO	ABC	DE	HS	GSA
***p* <= Bonferroni**	1	1	1	1	1

Table 9. Bonferroni correction for the case of ABC comparison.

ABC vs	PSO	DE	HS	GSA	FPA
***p* <= Bonferroni**	1	1	1	1	1

4.7. Comparison Study in Images

Figures 4 and 5 present the visual performance of the designed filters when they are applied to eliminate Gaussian noise for two different images.

Figure 4 shows the noise removing the results of the filters whose corresponding parameters are exhibited in Table 2. They have been designed through the use of PSO, ABC, DE, HS, GSA and FPA considering the approximation to an ideal low pass behavior. The visual results reflex the capacity of each filter to remove noise. Figure 4a presents the original image "Lenna". Figure 4b shows the image corrupted with Gaussian noise of mean = 0 and variance of 0.005. The set of images from Figure 4c–h exhibits the performance of each method PSO, ABC, DE, HS, GSA and FPA in the noise removing process. According to the Figure 4, it is clear that the filters produced by the FPA and ABC algorithms present the best performance. On the other hand, the filters generated by the algorithms PSO, DE, HS and GSA obtain the worst filtering effect. They are not able to remove appropriately, also producing distortions in the image due to the frequency ripples.

Figure 5 shows the noise removing the results of the filters with a wider bandwidth size. They correspond to the filters produced by the methods PSO, ABC, DE, HS, GSA and FPA, considering as ideal response that defined by M_d^1. Figure 5a presents the original image "Cameraman". Figure 5b shows the image corrupted with Gaussian noise of mean = 0 and variance of 0.005. The set of images from Figure 5c–h exhibits the performance of each method PSO, ABC, DE, HS, GSA and FPA in the noise removal process. Also, in this case, the filters produced by the FPA and ABC algorithms present the best noise removal result. Contrarily, the filters obtained by the algorithms PSO, DE, HS and GSA produce a bad filtering effect.

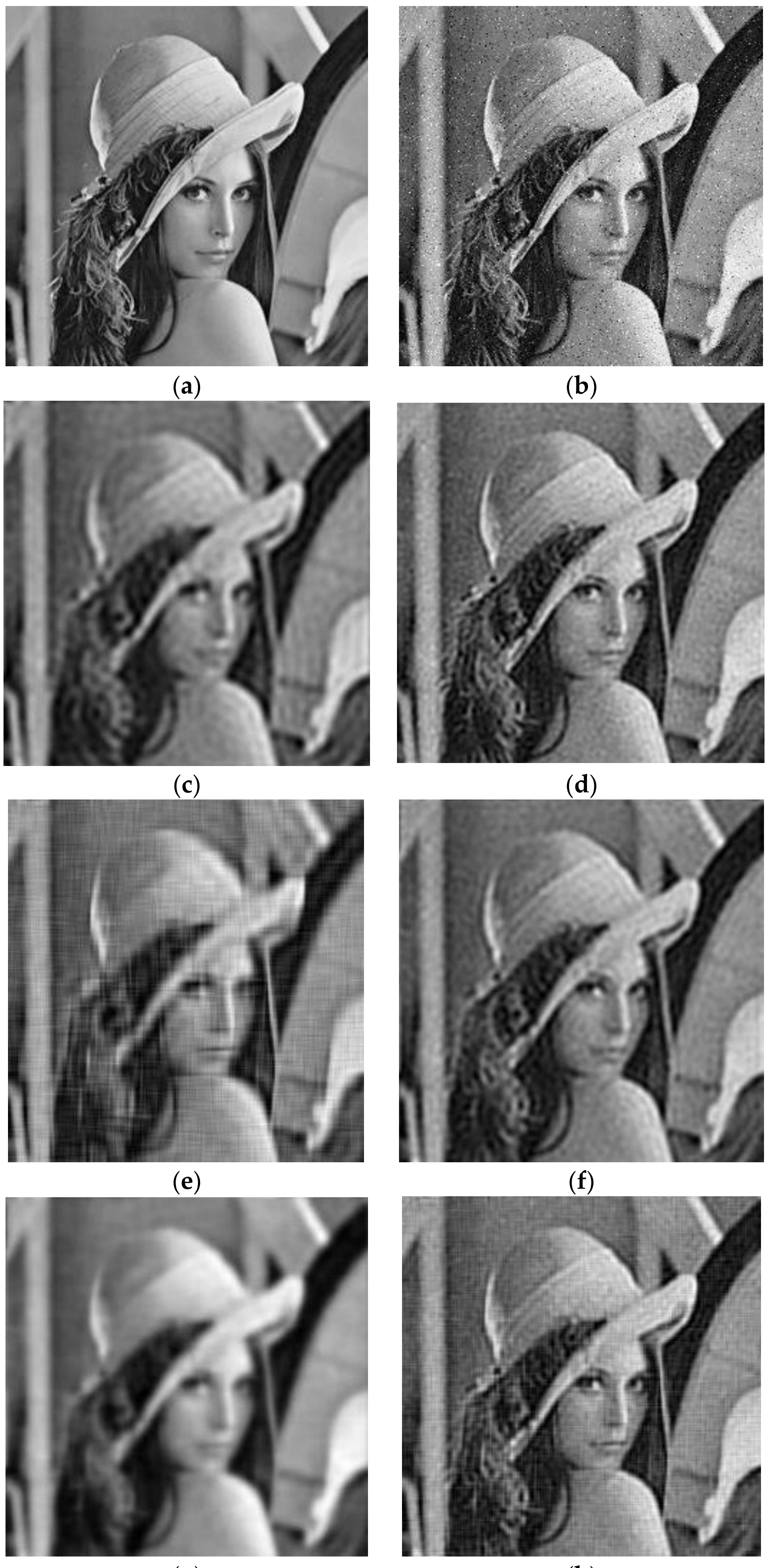

Figure 4. Visual comparison of the different designed filters. (**a**) Original image, (**b**) corrupted image, (**c**) filtering with PSO, (**d**) filtering with ABC, (**e**) filtering with DE, (**f**) filtering with HS, (**g**) filtering with GSA and (**h**) filtering with FPA.

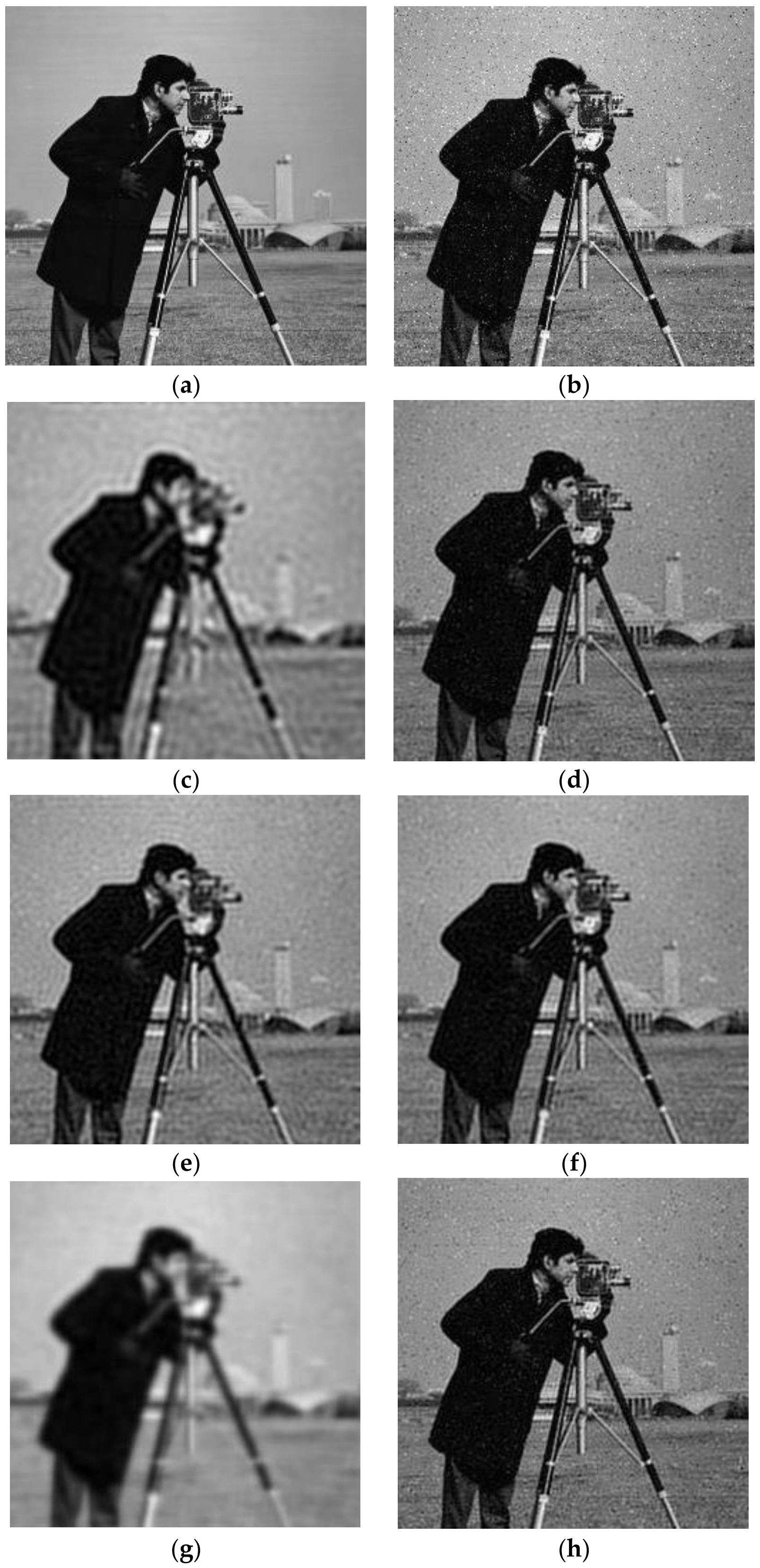

Figure 5. Visual comparison of the different designed filters with a wider bandwidth. (**a**) Original image, (**b**) corrupted image, (**c**) filtering with PSO, (**d**) filtering with ABC, (**e**) filtering with DE, (**f**) filtering with HS, (**g**) filtering with GSA and (**h**) filtering with FPA.

5. Discussion

The performance of EC methods is based on the balance achieved by two search procedures, namely, exploration and exploitation. In the exploration procedure, the search mechanism of EC algorithms allows producing large movements within the search space, increasing the possibility of finding potential zones. As a counterpart, the exploitation procedure increases the population diversity within the discovered zones by the exploration stage, producing better and more accurate solutions. Additionally, to know the advantages and limitations of EC methods, their correct evaluation is an important task in the computational intelligence community. In general, EC algorithms are complex pieces of software with several operators and stochastic branches. Under such circumstances, appropriate statistical methods must be considered for obtaining significant conclusions.

In this paper, a comparison study among several EC methods is conducted for solving the 2D-IIR filter design optimization problem. The experimental study compares commonly used EC methods for solving complex optimization methods. From the results, it is quite evident that the GSA algorithm is capable of producing better results than PSO, DE and HS. However, the FPA overcomes the GSA method due to the valuable combination of *Lévy* flights in its global pollination structure and the local pollination procedure. It can be demonstrated that for most of the experiments, FPA can be used as an alternative method for solving 2D-IIR filters.

The performance results obtained by FPA outperform the PSO, DE, HS, and GSA concerning the average and standard deviation of the values after 250 individual executions. On the other hand, the ABC overcomes the FPA in standard deviation terms, which means that ABC can converge to almost the same solution with low variation, which indicates that FPA and ABC are good alternatives for this specific application. In the case of PSO, DE, HS and GSA the fitness values obtained by these techniques were oscillatory, showing they were not able to perform the 2D-IIR filter design well.

To validate the performance results based on the minimization process, a non-parametric test known as Wilcoxon's rank-sum was conducted. In this study, the significance level of 0.05 has been chosen, to prove if there is a statistical difference among the fitness values achieved by FPA and its competitors. Additionally, to avoid type 1 error, the Bonferroni correction has been computed to avoid false-positive results. From the statistical results, the significant performance of FPA is achieved by the balance between the global and local pollination operators.

6. Conclusions

Recently, the design of two-dimensional Infinite Impulse Response (2D-IIR) has attracted attention in several areas of engineering due to its wide range of applications. Most of the related literature concerning the optimal design of 2D-IIR structure is based on traditional optimization methods that are prone to fail in the presence of multiple optima. Under such circumstances, the optimal design of 2D-IIR filters is considered as a complex optimization problem involving multimodal surfaces.

On the other hand, Evolutionary Computation (EC) algorithms are considered as stochastic global search methods that can operate under complex multimodal error surfaces. This paper reports a comparative study considering several EC methodologies when they are applied to the complex application of 2D-IIR filter design. In comparison, special attention is paid to the most popular EC algorithms currently in use such as Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Differential Evolution (DE), Harmony Search (HS), Gravitational Search Algorithm (GSA) and Flower Pollination Algorithm (FPA).

From the experimental results, it can be shown that the FPA method outperforms the rest of the EC techniques in terms of accuracy, consistency and robustness. Additionally, the remarkable performance of the FPA is statistically compared against the performance of its competitors considering a non-parametric test and a type 1 error correction based on a Bonferroni correction to avoid false-positive solutions. From the statistical tests, it can be corroborated that the FPA can find the optimal structure for the 2D-IIR filter design optimization problem.

Author Contributions: Formal analysis, J.G.; Investigation, O.A.; Methodology, E.H.H.; Software, K.H.; Writing—original draft, E.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare that there is no conflict of interests regarding the publication of this paper.

References

1. Srivastava, V.K.; Ray, G.C. Design of 2D-multiple notch filter and its application in reducing blocking artifact from DCT coded image. In Proceedings of the 22nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Chicago, IL, USA, 23–28 July 2000; pp. 2829–2833.
2. Rashedi, E.; Zarezadeh, A. Noise filtering in ultrasound images using gravitational search algorithm. In Proceedings of the 2014 Iranian Conference on Intelligent Systems (ICIS), Bam, Iran, 4–6 February 2014; pp. 1–4.
3. Sgro, J.A.; Emerson, R.G.; Pedley, T.A. Real-time reconstruction of evoked potentials using a new two-dimensional filter method. *Electroencephalogr. Clin. Neurophysiol. Potentials Sect.* **1985**, *62*, 372–380. [CrossRef]
4. Razee, A.M.; Dziyauddin, R.A.; Azmi, M.H.; Sadon, S.K. Comparative Performance Analysis of IIR and FIR Filters for 5G Networks. In Proceedings of the 2018 2nd International Conference on Telematics and Future Generation Networks (TAFGEN), Kuching, Malaysia, 24–26 July 2018; pp. 143–148.
5. Pal, R. Comparison of the design of FIR and IIR filters for a given specification and removal of phase distortion from IIR filters. In Proceedings of the 2017 International Conference on Advances in Computing, Communication and Control (ICAC3), Mumbai, India, 1–2 December 2017; pp. 1–3.
6. Dumitrescu, B. Optimization of two-dimensional IIR filters with nonseparable and separable denominator. *IEEE Trans. Signal. Process.* **2005**, *53*, 1768–1777. [CrossRef]
7. Das, S.; Konar, A. Two-dimensional iir filter design with modern search heuristics: A comparative study. *Int. J. Comput. Intell. Appl.* **2006**, *06*, 329–355. [CrossRef]
8. Whalley, R. Two-dimensional digital filters. *Appl. Math. Model.* **1990**, *14*, 304–311. [CrossRef]
9. Lu, W.-S.; Andreas, A. *Two-Dimensional Digital Filters*; CRC Press: New York, NY, USA, 1992.
10. Dhabal, S.; Venkateswaran, P. Two-Dimensional IIR filter design using simulated annealing based particle swarm optimization. *J. Optim.* **2014**, *2014*, 239721. [CrossRef]
11. Ampazis, N.; Perantonis, S.J. An efficient constrained learning algorithm for stable 2D IIR filter factorization. *Adv. Artif. Neural Syst.* **2013**, *2013*, 1–7. [CrossRef]
12. Jin, Y.; Branke, J. Evolutionary optimization in uncertain environments—A survey. *IEEE Trans. Evol. Comput.* **2005**, *9*, 303–317. [CrossRef]
13. Kukrer, O. Analysis of the dynamics of a memoryless nonlinear gradient IIR adaptive notch filter. *Signal. Process.* **2011**, *91*, 2379–2394. [CrossRef]
14. Nanda, S.J.; Panda, G. A survey on nature inspired metaheuristic algorithms for partitional clustering. *Swarm Evol. Comput.* **2014**, *16*, 1–18. [CrossRef]
15. Das, S.; Konar, A. A swarm intelligence approach to the synthesis of two-dimensional IIR filters. *Eng. Appl. Artif. Intell.* **2007**, *20*, 1086–1096. [CrossRef]
16. Mastorakis, N.E.; Gonos, I.F.; Swamy, M.N.S. Design of two-dimensional recursive filters using genetic algorithms. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **2003**, *50*, 634–639. [CrossRef]
17. Karaboga, N. A new design method based on artificial bee colony algorithm for digital IIR filters. *J. Franklin Inst.* **2009**, *346*, 328–348. [CrossRef]
18. Tan, K.C.; Chiam, S.C.; Mamun, A.A.; Goh, C.K. Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization. *Eur. J. Oper. Res.* **2009**, *197*, 701–713. [CrossRef]
19. Alba, E.; Dorronsoro, B. The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. *IEEE Trans. Evol. Comput.* **2005**, *9*, 126–142. [CrossRef]
20. Paenke, I.; Jin, Y.; Branke, J. Balancing population and individual-level adaptation in changing environments. *Adapt. Behav.* **2009**, *17*, 153–174. [CrossRef]
21. Cuevas, E.; Echavarría, A.; Ramírez-Ortegón, M.A. An optimization algorithm inspired by the States of matter that improves the balance between exploration and exploitation. *Appl. Intell.* **2014**, *40*, 256–272. [CrossRef]
22. Reddy, S.S.; Rathnam, C.S. Optimal power flow using glowworm swarm optimization. *Int. J. Electr. Power Energy Syst.* **2016**, *80*, 128–139. [CrossRef]

23. Reddy, S.S. Optimal power flow using hybrid differential evolution and harmony search algorithm. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 1077–1091.
24. Reddy, S.S.; Bijwe, P.R.; Abhyankar, A.R. Faster evolutionary algorithm based optimal power flow using incremental variables. *Int. J. Electr. Power Energy Syst.* **2014**, *54*, 198–210. [CrossRef]
25. Caraffini, F.; Kononova, A.V.; Corne, D. Infeasibility and structural bias in differential evolution. *Inf. Sci.* **2019**, *496*, 161–179. [CrossRef]
26. Garcia, S.; Molina, D.; Lozano, M.; Herrera, F. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 special session on real parameter optimization. *J. Heuristics* **2009**, *15*, 617–644. [CrossRef]
27. Elbeltagi, E.; Hegazy, T.; Grierson, D. Comparison among five evolutionary-based optimization algorithms. *Adv. Eng. Inform.* **2005**, *19*, 43–53. [CrossRef]
28. Shilane, D.; Martikainen, J.; Dudoit, S.; Ovaska, S.J. A general framework for statistical performance comparison of evolutionary computation algorithms. *Inf. Sci.* **2008**, *178*, 2870–2879. [CrossRef]
29. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995.
30. Karaboga, D. An idea based on honey bee swarm for numerical optimization. *Tech. Rep. TR06 Erciyes Univ.* **2005**, *TR06*, 10.
31. Storn, R.; Price, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
32. Geem, Z.W. A new heuristic optimization algorithm: Harmony search. *Simulation* **2001**. [CrossRef]
33. Rashedi, E.; Nezamabadi-pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
34. Yang, X.S. Flower pollination algorithm for global optimization. *Lect. Notes Comput. Sci. (Incl. Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinform.)* **2012**, *7445*, 240–249.
35. Lin, Y.-L.; Chang, W.-D.; Hsieh, J.-G. A particle swarm optimization approach to nonlinear rational filter modeling. *Expert Syst. Appl.* **2008**, *34*, 1194–1199. [CrossRef]
36. Rashedi, E.; Rashedi, E.; Nezamabadi-pour, H. A comprehensive survey on gravitational search algorithm. *Swarm Evol. Comput.* **2018**, *41*, 141–158. [CrossRef]
37. Barthelemy, P.; Bertolotti, J.; Wiersma, D.S. A *Lévy* flight for light. *Nature* **2008**, *453*, 495–498. [CrossRef] [PubMed]
38. Mladenov, V.M.; Mastorakis, N.E. Design of two-dimensional recursive filters by using neural networks. *IEEE Trans. Neural Netw.* **2001**, *12*, 585–590. [CrossRef] [PubMed]
39. Tsai, J.-T.; Ho, W.-H.; Chou, J.-H. Design of two-dimensional IIR digital structure-specified filters by using an improved genetic algorithm. *Expert Syst. Appl.* **2009**, *36*, 6928–6934. [CrossRef]
40. Sarangi, S.K.; Panda, R.; Dash, M. Design of 1-D and 2-D recursive filters using crossover bacterial foraging and cuckoo search techniques. *Eng. Appl. Artif. Intell.* **2014**, *34*, 109–121. [CrossRef]
41. Cuevas, E.; Cienfuegos, M. A new algorithm inspired in the behavior of the social-spider for constrained optimization. *Expert Syst. Appl.* **2014**, *41*, 412–425. [CrossRef]
42. Sun, J.; Fang, W.; Xu, W. A quantum-behaved particle swarm optimization with diversity-guided mutation for the design of two-dimensional IIR digital filters. *IEEE Trans. Circuits Syst. II Express Briefs* **2010**, *57*, 141–145. [CrossRef]
43. Kockanat, S.; Karaboga, N.; Koza, T. Image denoising with 2-D FIR filter by using artificial bee colony algorithm. In Proceedings of the 2012 International Symposium on INnovations in Intelligent SysTems and Applications, Trabzon, Turkey, 2–4 July 2012.
44. Geem, Z.W. Global optimization using harmony search: Theoretical foundations and applications. *Stud. Comput. Intell.* **2009**, *203*, 57–73.
45. Das, S.; Dey, D. Design of two-dimensional IIR filters using an improved DE algorithm. In Proceedings of the First international conference on Pattern Recognition and Machine Intelligence, Kolkata, India, 20–22 December 2005; pp. 369–375.
46. Wilcoxon, F. Individual comparisons by ranking methods. *Biom. Bull.* **1945**, *1*, 80–83. [CrossRef]
47. Cuevas, E.; Gálvez, J.; Hinojosa, S.; Avalos, O.; Zaldívar, D.; Pérez-Cisneros, M. A comparison of evolutionary computation techniques for IIR model identification. *J. Appl. Math.* **2014**, *2014*. [CrossRef]

48. Hochberg, Y. A sharper Bonferroni procedure for multiple tests of significance. *Biometrika* **1988**, *75*, 800–802. [CrossRef]
49. Armstrong, R.A. When to use the Bonferroni correction. *Ophthalmic Physiol. Opt.* **2014**, *34*, 502–508. [CrossRef] [PubMed]