energies

MDPI

Article

ANN-Based Stop Criteria for a Genetic Algorithm Applied to Air Impingement Design

Ander Sánchez-Chica [1], Ekaitz Zulueta [1,*], Daniel Teso-Fz-Betoño [1], Pablo Martínez-Filgueira [2] and Unai Fernandez-Gamiz [3]

1 System Engineering & Automation Control Department, University of the Basque Country, UPV/EHU, Nieves Cano 12, 01016 Vitoria-Gasteiz, Spain; ander.sanchez@ehu.eus (A.S.-C.); daniel.teso@ehu.eus (D.T.-F.-B.)
2 CS Centro Stirling S. Coop., Avenida Álava 3, 20550 Aretxabaleta, Spain; pmartinez@centrostirling.com
3 Nuclear Engineering & Fluid Mechanics Department, University of the Basque Country, UPV/EHU, Nieves Cano 12, 01016 Vitoria-Gasteiz, Spain; unai.fernandez@ehu.eus
* Correspondence: ekaitz.zulueta@ehu.eus

Received: 18 October 2019; Accepted: 16 December 2019; Published: 19 December 2019

Abstract: Artificial Neural Networks (ANNs) have proven to be a powerful tool in many fields of knowledge. At the same time, evolutionary algorithms show a very efficient technique in optimization tasks. Historically, ANNs are used in the training process of supervising networks by decreasing the error between the output and the target. However, we propose another approach in order to improve these two techniques together. The ANN is trained with the points obtained during an optimization process by a genetic algorithm and a flower pollination algorithm. The performance of this ANN is used as a stop criterion for the optimization process. This new configuration aims to reduce the number of iterations executed by the genetic optimizer when learning the cost function by an ANN. As a first step, this approach is tested with eight benchmark functions. As a second step, the authors apply it to an air jet impingement design process, optimizing the surface temperature and the fan efficiency. Finally, a comparison between the results of a regular optimization and the results obtained in the present study is presented.

Keywords: air jet impingement; artificial neural network; genetic algorithms; cooling enhancement; multi-objective optimization

1. Introduction

In industrial applications, thermal management is a critical task, requiring optimal design of cooling or heating systems. Among heat transfer methods, jet impingement is a technique with excellent performance for applications where high heat fluxes are required. The working fluid, which can be air, passes through one or more nozzles, increasing its speed and impinging on the surface of the element to be cooled or heated. In a previous work by Martinez-Filgueira et al. [1], the particularities of this kind of design were studied extensively.

The air impingement system requires an appropriate air supply, which is commonly provided by a blower. The influence of this component on the heat exchange system makes the selection or the design of a proper blower a major task. The blower is usually characterized by different variables such as the specific diameter (Ds) and the specific speed (Ns); for more details, see Wright et al. [2]. One of the main properties of the blower that influences the air jet impingement application is the efficiency, which can be expressed in function of Ns. In other cases, efficiency is directly related to the flow; see the study of Ingole et al. [3].

To make a correct characterization of heat exchange, it is necessary to define the variables that impact directly on the cooling performance. These variables can be classified into two different groups. In the first class, thermal and fluid properties such as the Nusselt number (Nu) [3], which defines the ratio of convective to conductive heat transfer, and the Reynolds number (Re) [4] which is the ratio between viscous and inertial forces, can be found. The second class is composed of geometrical variables such as the diameter (D), the height of the plate (z) and diameter ratio (z/D) [5], and the space between the nozzle center and diameter ratio (s/D) [6].

Jain et al. [7] explained that nature-inspired meta-heuristic optimizations algorithms are a particular type of method of artificial intelligence that imitate animals and plants behavior or physical phenomena. In that work a summarize of the different nature-based algorithms that can be found in the literature from the beginning of this field of investigation in 1975 when genetic algorithms (GA) were developed. In Jain's work, more than 20 different algorithms developed before 2015 are mentioned, such as the flower pollination algorithm (FPA) [8] or Dragonfly algorithm (DA) [9]. Other algorithms inspired by nature can also be found in the literature, such us Polar Bear optimization (PBO), see the study of Polap et al. [10]. In general, these algorithms can be described as a meta heuristic equation that modify the values of a variable normally called "agent" or "individuals". These variables change their value across the search space in order to find an optimal solution. Nowadays the researchers try to implement new heuristic functions, inspired by nature in order to found more efficient and powerful optimization algorithms.

PBO is based on polar bear's seal hunting behavior. This algorithm has two different search approaches. The global area moves between ice floes (global search) and the hunting of the seals (local search). The global search is applied to the top 10% of the population. The new position of each bear is calculated based on the distance towards the best bear. Moreover, the global search is applied to all the bears in the population. The new location is determinate by an excerpt from a modified tryfolium equation. The algorithm also implements a dynamic population approach controlled by reproduction and extinction process. Both processes have a probability of 25%. The reproduction consists of the average of the best bear and another one from the top rated 10% among all bears.

DA is also based in an animal behavior. In this case it is based on the dragon fly swarm movement and distribution in the space. The algorithm implements three mechanism that emulates the dragonfly swarm behavior: Separation, Alignment and Cohesion. In addition, in order to secure the subsistence of the swarm, the distance to the food and from the predators is implement. These five main factors can be defined as follows:

The separation: the distance that dragonflies maintains between each other in order to avoid collisions. For each individual, the separation is calculated by the sum of the distance between the individual and the neighborhoods.

Alignment: individuals adjust their velocity in order to match it with the velocity of their neighborhoods. The alignment is defined as the average speed of the surrounding neighborhoods for each individual.

Cohesion: Dragonfly's tend to advance towards the center of the mass of the swarm. Cohesion can be calculated with the difference between the induvial position from the center of mass of the surrounding neighborhoods.

The food source is defined as the best solution found. The enemy, on the other hand, as the worst solution. The attraction factor towards the food is the distance between an individual and the food source. At the same time, the distraction outwards an enemy is the distance between the individual and the enemy.

Unlike the two previous algorithms, FPA is not based on an animal behavior. It is inspired by plant reproduction process. The main idea of the algorithm is to use the pollination principles as an optimization tool. Like other meta-heuristic algorithms, FPA performs a different process for global and local searches. Each induvial only perform one process per iteration. The process to be performed by each individual is selected randomly.

In a global search, the individual actualizes its position based on the distance between its position and the best individual multiplied by a random number generated by the Levy distribution. Local search actualizes in function of the distance between another two random individuals from the population. The distance is multiplied by a random number.

GAs were developed several years ago. They are commonly use in optimization tasks. Beasley et al. [11] explain that they are composed of populations that use heuristic and stochastic mechanics to solve problems such as search and optimization. GAs are adaptive methods, which can solve real-world and engineering problems. Many examples of GA applications in optimization problems can be found in the literature, such as crude oil operations [12], energy management optimization in electric vehicles by Wieczorek et al. [13], optimization of a building's thermal design by Ferdyn-Grygiereket et al. [14] and optimization of a solar chimney power plant's collector roof by Gholamalizadeh et al. [15].

GAs emulate natural behavior, and for each particle that forms the population, a "fitness" or "cost" score is assigned depending on the problem and according to a defined function. For this emulation purpose, GAs' coding distinguishes some functions such as "Reproduction", where two individuals are selected for breeding. New particles called "children" are created in the "offspring" process. In some cases, the oldest particles suffer a "die out". Every iteration of the algorithm that involves these functions is called "generation". The GA architecture and working process are largely discussed in the related literature [16,17] and present different types and approaches. The correct selection and implementation of the different GA variants depend on the application. Due to the new children generation process, GAs is a good choice in applications where the variety of the population is critical.

Artificial Neural Networks (ANNs) are a type of machine learning that aims to emulate the behavior of the human brain. Ali et al. explain in [18] that ANNs are a combination of simpler computation elements called "neurons". ANNs are divided into three main types of layer: Input, Hidden, and Output. Different types of ANN can be distinguished depending on their internal connection and the number of hidden layers, such as single-layer perceptron, multi-layer perceptron or competitive networks.

These tools allow researchers to solve different types of problems, such as numerical regression, pattern recognition, clustering, and image processing. These problems appear in a large amount of engineering and real-life applications such as nanophotonic particle simulation and design by Peurifoy et al. [19], modeling of photovoltaic modules by Manuel Lopez-Guede et al. [20] or horizontal-axis wind turbine control by Saenz-Aguirre et al. [21]. Arena et al. [22] presented a combination of game theory and GA for the optimization of the Parrando paradox probability region.

The current work aims to develop a neural network trained alongside a genetic algorithm optimization process and to use its training performance as a stop condition. In addition, the new optimization approach is tested with various benchmark functions. This test is repeated with a Flower pollination algorithm in order to compare the performance of both algorithms.

The final goal is to apply this technique to an air jet impingement cooling system design by optimizing the surface junction temperature and hydraulic efficiency.

2. Methods

2.1. Air Impingement Design

A novel methodology to calculate the junction temperature of the cooled surface and the machining time of the plate was presented in the work of Martínez-Filgueira et al. [1]. In the current work, the same criteria for the temperature calculation are used. However, the machining time is discarded as a design criterion in the present study, and it has been replaced by the hydraulic efficiency. In both cases, the variables needed for the calculation are the nozzle diameter (D) and nozzle-to-nozzle spacing and diameter ratio (s/D).

In this application, an industrial centrifugal fan is used as a source of flow. Such fans operate according to their characteristic curve, which indicates the relation between the pressure and the flow rate provided by the fan [23]. Figure 1 illustrates an example of this type of curve.

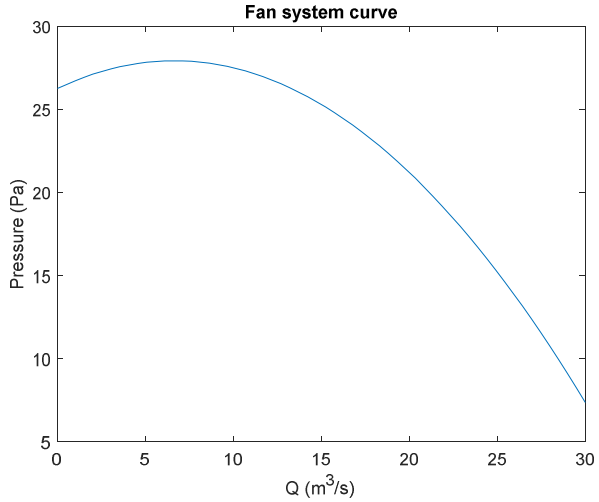


Figure 1. Fan system curve adapted from [2].

Alongside the previous figure, industrial fans are also characterized by efficiency curves. Even though the fan can work at all the points contained in the fan curve, the efficiency changes depending on the point of operation. Figure 2 shows how the maximum efficiency that is achieved when the air flow supplied by the fan is higher than half the maximum.

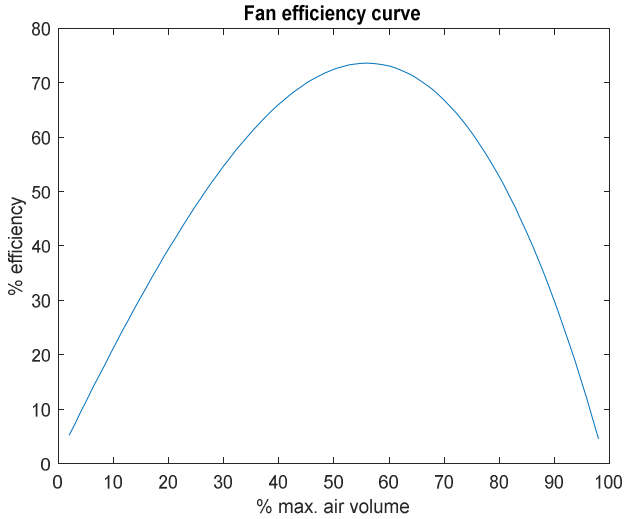


Figure 2. A typical fan static efficiency curve. Adapted from [2].

The plot presented in Figure 2 is assumed in the current design as an estimation for the efficiency. In order to implement the efficiency calculation on our air jet impingement design algorithm, a regression of the data presented in Figure 2 was performed. In order to obtain a mathematical expression, the figure adapted from [2] was processed to obtain data points from the curve. A numerical regression was performed with the obtained points. Finally, the following expression, in terms of efficiency (η) and relative air volume (Φ), was obtained:

$$\begin{aligned}\eta = \; & 8.0346 \times 10^{0}\Phi^{6} - 3.0005 \times 10^{1}\Phi^{5} + 4.1693 \times 10^{1}\Phi^{4} - 2.3701 \times 10^{1}\Phi^{3} \\ & + 9.6981 \times 10^{-1}\Phi^{2} + 3.0118 \times 10^{0}\Phi\end{aligned} \quad (1)$$

The relative air volume is the fraction of the maximum flow that is achieved at the operation point. In our case, the maximum flow provided by the fan is 61 m^3/h. So, Φ can be calculated as:

$$\Phi = \frac{flow\ at\ operation\ point}{\text{maximum flow}} \tag{2}$$

The design criteria and restrictions are summarized in Table 1. In a previous study [1], the size of the plate was fixed to match the dimension of an industrial insulated gate bipolar transistor (IGBT). In the present work, we aim to cool a hypothetical surface with the same properties as that device, but considerably more prominent. Therefore, the research team decided a new plate size in order to increase the search space for the algorithm. The z/D ratio was set to be equal to 3 according to the results of Xing et al. [24] and Attala et al. [25]. The minimum s/D ratio was also set by the research team in order to set a minimum nozzle-to-nozzle spacing. Finally, the nozzle diameter and the maximum s/D ratio were selected by the research team's criteria. This change in the specifications allows for finding more solutions that are possible and checking the performance of the new algorithm in a better way.

Table 1. Design constraints. Plate size and z/D ratio are fixed. The diameter and s/D ratio are the variables of the optimization process. This table presents their limit values.

Geometrical Restraint	Value
Plate width	0.20 m
Plate length	0.15 m
Maximum Nozzle Diameter	10 cm
Minimum Nozzle Diameter	0.05 cm
Maximum s/D ratio	200
Minimum s/D ratio	2
z/D	3

2.2. Fitness Function

A multi-objective optimization is proposed in this work. One way to solve this kind of problem is to reduce it to a single optimization problem [1]. Minimizing the temperature (T) of the surface and maximizing η are the main goals. So, in order to implement a single function where the two variables need to be minimized, Equation (3) is presented where the loss of efficiency $(1 - \eta)$ is added as a minimization criterion:

$$Cost = w_T \times T + w_\eta \times (1 - \eta) \times\ 100 \tag{3}$$

where W_T and W_η are two coefficients that weight the temperature and the loss of efficiency. The value of these coefficients is determined by the users, as is explained in the study of Lin et al. [26]. In this work, both coefficients are set to be equal to 0.5. The loss of efficiency is multiplied by a factor of 100 in order to set this value in the same order as the temperature.

2.3. Optimization Algorithms

The main task of this work is to find a new stop criterion in order to reduce the computational cost of optimizing the air impingement problem. For this optimization task, the research team propose two different algorithms a GA and a Flower pollination algorithm.

In the present work, the solutions found by the algorithm in each iteration works as the inputs of the neural network, so it is necessary to maintain a high level of diversity throughout the optimization process. Due to the diversity requirement, exploration is predominant over exploitation during the process.

2.3.1. Genetic Algorithm

In Table 2 the algorithm parameters are presented. The number of variables was set to be equal to the number of variables needed to solve the air impingement problem (D and s/D). The rest of the values presented in the table were decided after a preliminary test, where different combinations were analyzed.

Table 2. Presentation of genetic algorithm parameters.

GA Parameter	Value
Number of variables	2
Population	500
Maximum Generations	500
Elite count	10
Crossover factor	0.5
Tournament number	5

The crossover factor indicates the fraction of particles dedicated to the crossover process. The number of particles destined for the mutation process is the remainder of the total population after the elite count and crossover processes.

An initialization of the variables is necessary to start the optimization process. After the population is initialized randomly, each particle is evaluated by the cost function and the optimization process begins. First, the elite count process is executed. It consists of the selection of k particles with the best cost. The elite count variable indicates the value of k, which is 10 in the present study.

The second process is known as crossover. It consists of a cross of two particles called "parents" to obtain a new particle, the "child". This process can be divided into two steps. The first one is the selection, where a few particles from the population are chosen to be the next parents. Different methodologies can be used to decide which particles will be selected, such as roulette wheel, rank, tournament, or Boltzmann. In a work by Nish [27], a review of these different methods was presented. In the algorithm developed in the present study, the tournament method developed by Abbas [28] is implemented. This method consists of making a random selection of n particles from the population. The value of n is set up by the tournament number variable. Finally, between these n particles, the one with the best cost is selected. Since the selection of the particles is random, these techniques maintain the diversity of the population.

The second step in the crossover process consists of mixing two parents selected by the tournament method and generating a child particle. Equation (4) describes how this child is obtained, where α is a random number between 0 and 1.

$$Child = parent_1 \times \alpha + parent_2 \times (1 - \alpha) \quad (4)$$

The number of children produced in every generation is decided by the crossover factor. In the current case, 250 particles are generated in each generation by this method.

Finally, mutation is the last process that occurs in the GA. The mutation changes the particle randomly, intending to increase the exploration of the algorithm. There are different manners of applying a mutation process. In our case, due to the diversity requirement for the population, the mutation adds new random particles to the population. In each generation, 240 completely new particles are added to the population. After the generation of all the new particles, the former ones are erased and substituted.

2.3.2. Flower Pollination Algorithm

The flower pollination algorithm is implemented following the process described in [8] by Abdel-Basset et al. in Table 3, the main parameters of FPA algorithm are presented. The values are set

to be equal to the proposed by Abdel-Basset. The number of iterations and the number of flowers are set to be equal to the values for GA presented in Table 3.

Table 3. Flower pollination parameters summary.

FPA Parameter	Value
Number of iterations	500
Number of flowers	500
Number of variables	2
p	0.5
ε	0.1
λ	0.5
a	1

2.4. Test Functions

Our algorithm was tested by benchmark functions. A list of these different functions is presented in Table 4. All the information exposed in this table, the function expressions and how to implement them in computers, can be found in [29,30]. These functions can be optimized, considering a significant number of dimensions; see the works of Mirjalili [31] and Evers et al. [32]. In our case, the algorithm only optimized two dimensions in order to mirror the air impingement design problem.

Table 4. Summary of benchmark functions used in the algorithm testing process. The limit of the search space and minimum point information is presented.

Function Tag	Function Name	Search Space	Global Minimum Point	Minimum Value
f1	Sphere	[−100,100]	[0,...,0]	0
f2	Schwefel	[–500,500]	[420.9687,...,420.9687]	0
f3	Schwefel_P2	[–10,10]	[0,...,0]	0
f4	Rosenbrock	[–5,10]	[1,...,1]	0
f5	Shubert	[–10,10]	Various	–186.7309
f6	Rothyp	[–65.536,65.536]	[0,...,0]	0
f7	Rastrigin	[–5.12,5.12]	[0,...,0]	0
f8	Ackley	[–32.768,32.768]	[0,...,0]	0

In order to check the performance of our algorithm, every function was optimized twenty times. The mean best solutions and their standard deviations were calculated. The results of the test can be seen in Section 3.

2.5. Neural Network Model

After the first test with the benchmark functions, a new functionality was added. In every generation, the value of the particles generated by the mutation process is stored. This continues until the accumulation of data is significant enough to begin the training of the neural network.

The current model is built by supervised training. The cost associated with each particle is used as the target of the training. In Table 5, the model structure and training parameters can be seen.

The number of inputs and outputs is defined by the air jet impingement problem. The other values presented in the table were obtained after a preliminary test, where different values and configurations were tested.

Table 5. Neural network model parameters. This configuration is applied without variation to all the benchmark functions.

ANN Design Parameters	Value
Number of inputs	2
Number of outputs	1
Number of hidden layers	2
1st Hidden layer size	15
1st Hidden layer activation function	Sigmoid
2nd Hidden layer size	15
2nd Hidden layer activation function	Radial Basis
Output layer size	1
Output layer function	Linear
Training Function	Levenberg–Marquardt
Performance	Mean squared error
Epochs	1000
Training ratio	85%
Validation Ratio	10%
Test Ratio	5%

2.6. New Stop Condition

The main objective of this paper is using the neural model explained in the previous section as stop criteria for the optimization process. As the maximum number of generations is 500, our algorithm will perform a training of the network after 50 generations, dividing the process into five steps. After the training, the performance is checked. If it is lower than 2, the optimization ends. In order to check the feasibility of the technique, the test explained in Section 2.3 was executed with the addition of these new criteria. Figure 3 shows a flow diagram of the proposed optimization algorithm.

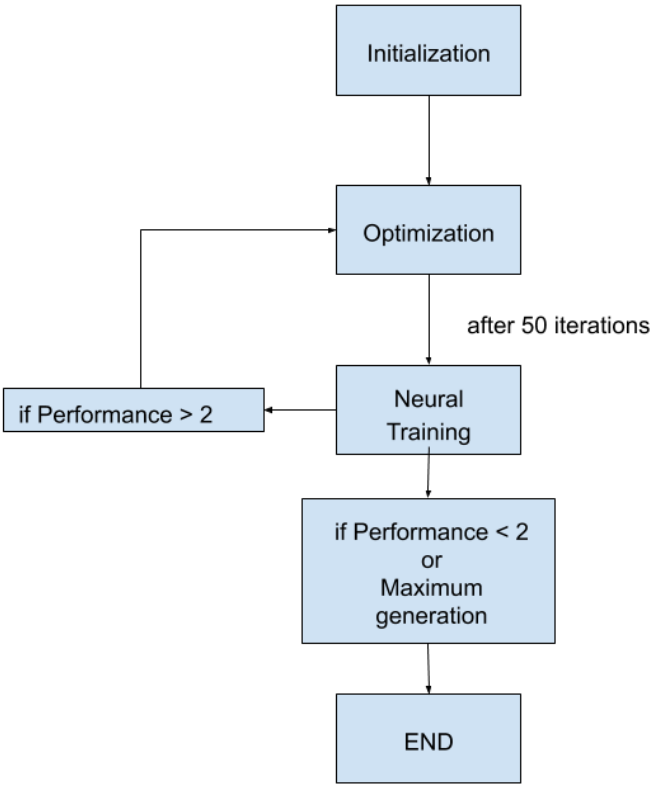


Figure 3. Optimization algorithm scheme.

Therefore, this new stop condition brings the following improvements: The optimization algorithm is stopped when the neural network learns the functions. The number of iterations is reduced. If each iteration has a high computational cost, these stop criteria improve the optimization time.

3. Results

3.1. Optimization with Genetic Algorithm

The results of the first test are illustrated in Table 6. Here, the genetic algorithm optimizes eight different equations without any stop condition. The algorithm ends at the moment of reaching the maximum number of generations.

Table 6. Genetic algorithm (GA) optimization test summary. The first column shows the mean values of the best results obtained after 20 executions. The second column shows the standard deviations of these results. Finally, the last two columns show the best and worst results obtained.

Tag	Mean Best Value	Standard Deviation	Best	Worst
f1	2.81×10^{-173}	0	0	5.62×10^{-172}
f2	2.60×10^{-5}	2.46×10^{-6}	2.55×10^{-5}	3.64×10^{-5}
f3	6.59×10^{-28}	2.95×10^{-27}	6.85×10^{-159}	1.32×10^{-26}
f4	1.40×10^{-6}	4.37×10^{-6}	0	1.97×10^{-5}
f5	-1.87×10^{2}	5.14×10^{-5}	-1.87×10^{2}	-1.87×10^{2}
f6	6.47×10^{-253}	0	0	1.29×10^{-251}
f7	3.23×10^{-9}	1.22×10^{-8}	0	5.44×10^{-8}
f8	8.88×10^{-16}	0	8.88×10^{-16}	8.88×10^{-16}

On the other hand, the same test was re-executed with the addition of the stop criteria by the neural network. Table 7 shows the results of the test with the addition of the information about the neural network training. The first four columns contain the same information as Table 6. The last two columns represent the best performance of the neural training and the median of the training performance after 20 executions.

Table 7. Results summary of the optimization test for the GA with the Artificial Neural Network (ANN)-based stop criteria.

Tag	Mean Best Value	Standard Deviation	Best	Worst	Best Network Performance	Median Network Performance
f1	1.08×10^{-30}	1.33×10^{-30}	0	3.93×10^{-30}	0.32	0.68
f2	2.55×10^{-5}	0	2.55×10^{-5}	2.55×10^{-5}	1.33×10^{3}	6.21×10^{3}
f3	1.36×10^{-16}	2.11×10^{-16}	2.69×10^{-32}	7.03×10^{-16}	7.66×10^{-4}	0.01
f4	8.20×10^{-06}	1.11×10^{-5}	0	2.99×10^{-5}	0.87	545.67
f5	-1.87×10^{2}	9.98×10^{-5}	-1.87×10^{2}	-1.87×10^{2}	513.38	667.44
f6	7.89×10^{-31}	1.17×10^{-30}	0	3.43×10^{-30}	0.12	0.68
f7	0	0	0	0	41.65	62.29
f8	2.66×10^{-15}	1.87×10^{-15}	8.88×10^{-16}	4.44×10^{-15}	0.31	0.32

3.2. Optimization with Flower Pollination

The eight functions presented in Table 4 are optimized with FPA algorithm. The result of the optimization test is presented in Table 8.

Table 8. Flower pollination algorithm test results. The first column shows the mean values of the best results obtained after 20 executions. The second column shows the standard deviations of these results. Finally, the last two columns show the best and worst results obtained.

Tag	Mean Best Value	Standard Deviation	Best	Worst
f1	7.95×10^{-12}	4.71×10^{-12}	5.93×10^{-13}	2.00×10^{-11}
f2	5.55×10^{-2}	1.15×10^{-1}	1.18×10^{-3}	3.89×10^{-1}
f3	5.84×10^{-7}	3.38×10^{-7}	1.16×10^{-7}	1.17×10^{-6}
f4	1.89×10^{-7}	2.12×10^{-7}	1.35×10^{-8}	7.94×10^{-7}
f5	-1.87×10^{2}	7.17×10^{-4}	-1.87×10^{2}	-1.87×10^{2}
f6	4.49×10^{-12}	6.10×10^{-12}	2.73×10^{-14}	2.67×10^{-11}
f7	9.58×10^{-2}	8.47×10^{-2}	4.75×10^{-3}	3.30×10^{-1}
f8	7.11×10^{-4}	5.29×10^{-4}	2.96×10^{-5}	2.45×10^{-3}

The result obtained for the FPA algorithm are worse than the previous results obtained in GA test. However, the result is good enough to consider FPA algorithm as a candidate for our work. In order to check the behavior of FPA algorithm with the proposed stop criteria, the test explained in Section 2.

Table 9 shows the results obtained in optimization test with the proposed stop criteria. The values of the training performance are worse than the obtained in the GA test. The low diversity of the flowers in the FPA algorithms and the low variation of the flowers between iterations generates bad training data.

Table 9. Results summary of the optimization test for the flower pollination (FPA) algorithm with the ANN-based stop criteria.

Tag	Mean Best Value	Standard Deviation	Best	Worst	Best Network Performance	Median Network Performance
f1	1.63×10^{7}	5.56×10^{4}	1.62×10^{7}	1.64×10^{7}	1.63×10^{7}	5.56×10^{4}
f2	2.43×10^{7}	7.43×10^{5}	2.35×10^{7}	2.62×10^{7}	2.43×10^{7}	7.43×10^{5}
f3	1.65×10^{7}	9.60×10^{4}	1.64×10^{7}	1.67×10^{7}	1.65×10^{7}	9.60×10^{4}
f4	2.46×10^{7}	5.58×10^{5}	2.39×10^{7}	2.55×10^{7}	2.46×10^{7}	5.58×10^{5}
f5	3.26×10^{7}	9.19×10^{5}	3.07×10^{7}	3.41×10^{7}	3.26×10^{7}	9.19×10^{5}
f6	1.64×10^{7}	7.04×10^{4}	1.63×10^{7}	1.65×10^{7}	1.64×10^{7}	7.04×10^{4}
f7	2.65×10^{7}	6.04×10^{5}	2.57×10^{7}	2.76×10^{7}	2.65×10^{7}	6.04×10^{5}
f8	2.28×10^{7}	6.08×10^{5}	2.20×10^{7}	2.37×10^{7}	2.28×10^{7}	6.08×10^{5}

The FPA algorithm, as is explained in [8], only update the flowers values if a new flower with a better solution is found. This is a good feature for the optimization process. However, a dynamic population is required for the approach presented in this work.

Other algorithms presented in the literature and the introduction have a similar problem. The research team implement a GA with modifications in order to obtain a better data training.

3.3. Computational Cost Reduction with GA

In order to show how the computational cost is reduced by the new stop criteria, Table 10 is presented. The stop criteria have no effect on functions f2 f5 and f7; this correlates with the data in the previous table. The training performance did not reach the minimum value in any run of the algorithm, showing that the stop criteria do not stop the optimization process. For the other five functions, a 50% reduction in iterations can be appreciated in the worst case. It generally is enough with only 150 iterations, which means three trainings of the network, to learn the function.

Table 10. Summary of the generation reached by the algorithm with the new stop criteria.

Tag	Generation Reached Mode	Best Result	Worst Result
f1	100	100	250
f2	500	500	500
f3	100	100	250
f4	150	100	250
f5	500	500	500
f6	100	100	250
f7	500	500	500
f8	100	100	250

In addition, Figure 4 shows how many times the algorithm stops at a certain iteration for each function. In most of the cases, the algorithm stops after executing 100 iterations. Finally, a graphical comparison between the model obtained after the optimization process and the real function is attached in Figures A1 and A2 in Appendix A.

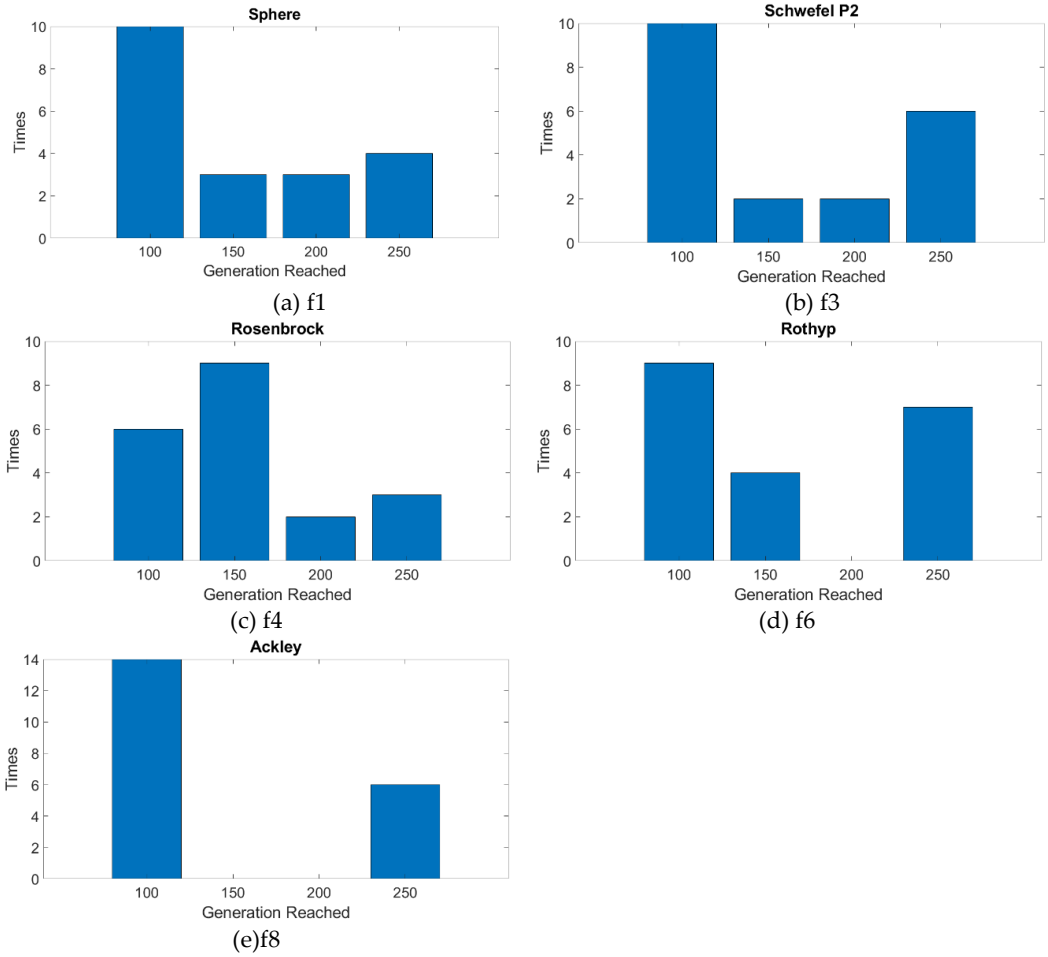


Figure 4. Absolute frequency of the generation reached by the algorithm for each function. F2 f5 and f7 are not shown because at every run, the algorithm reached the last iteration.

3.4. Air Impingement Design

The air impingement design problem is solved twice. First time with the GA without the proposed stop criteria. Then the new stop condition is added, and a new test is performed. The results obtained after ten runs of each method are summarized in Table 11.

Table 11. Summary of the two answers obtained in the present study.

Parameters	Genetic Algorithm	Genetic Algorithm with Stop Criteria
Mean best Cost	33.81	33.68
Standard deviation of best Cost	0.08	0.01
Mean temperature	31.66 °C	31.42
Std of temperature	0.17 °C	0.02 °C
Mean η	0.64	0.64
Std of η	2.9×10^{-4}	1.39×10^{-5}
Mean Network Performance	-	68.58
Generation Reached Mode	-	3.59

In the first case, the best configuration of nozzles is:

- Nozzle diameter: 0.249 cm
- Nozzle-to-nozzle spacing and diameter ratio: 13.96

Moreover, with the second method, the best solution found is:

- Nozzle diameter: 0.255 cm
- Nozzle-to-nozzle spacing and diameter ratio: 13.99

3.5. Air Impingement Design Graphical Analysis

A comparison between the cost surface and the obtained model is presented in Figure 5. In addition, the efficiency and temperature curves are shown.

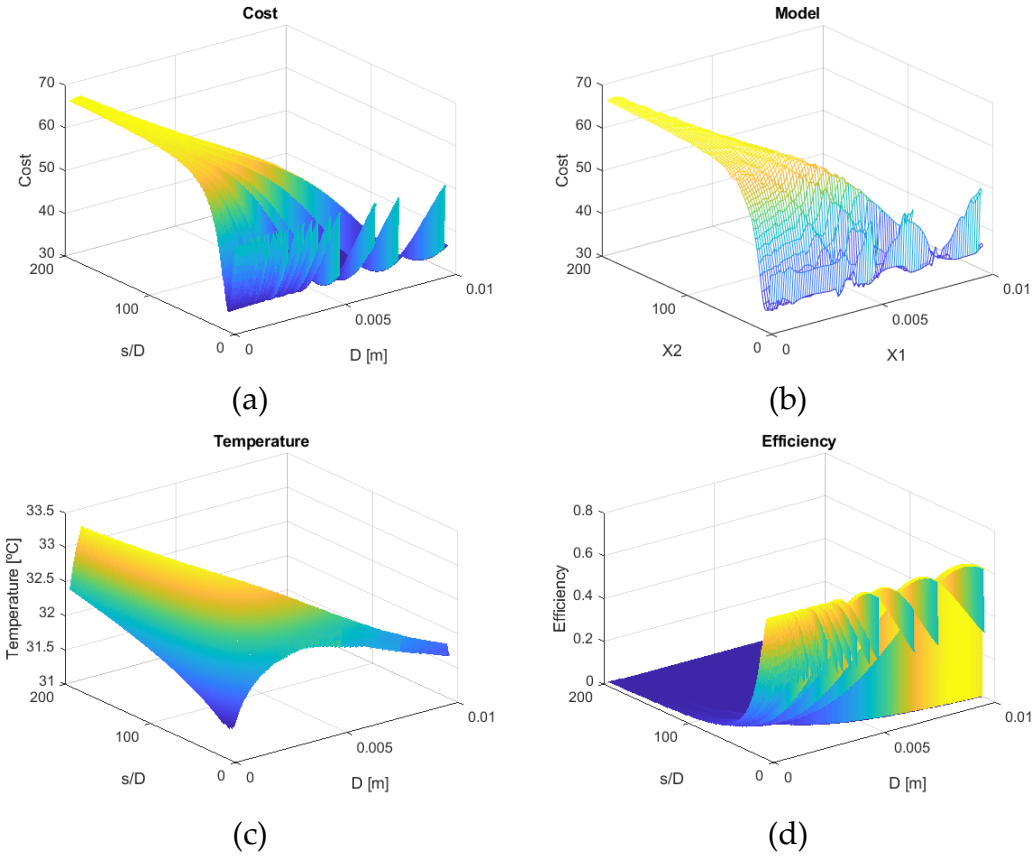


Figure 5. Comparison between the air impingement cost surface and the result obtained with the model. (**a**) Real cost function. (**b**) Model cost function. (**c**) Temperature surface. (**d**) Real efficiency surface.

4. Discussion

4.1. Testing the Algorithm

First, the eight functions are labeled in two different groups due to the results that appear in the experiments. The first group contains smooth functions (f1, f3, f4, f6, and f8), while the second group includes sharp functions (f2, f5, and f7). Table 6 shows the difference in the results between these two groups of functions. The functions with a smoother shape produce a mean result closer to the minimum. They also have a better best result compared with the sharp functions. This effect agrees with the standard deviation of the test; the sharp functions have the highest deviation due to the many local minima present in them.

The effect of the new stop criteria can be observed in Table 7. The difference between the mean best value achieved by the algorithm with the new stop criteria and the theoretical minimum one is larger than that obtained with the GA without stop criteria. Nonetheless, the solution achieved is good enough for our purposes, and besides, these stop criteria reduce the number of iterations run by the algorithm. Looking at the performance reached in each function, a significant difference is observed between the smooth and sharp functions.

The difference in the behavior of the algorithm between the smooth and sharp functions can also be appreciated in the graphical comparison. The smooth functions seem to be very similar in both cases. However, in sharp functions, a big difference between the plots can be appreciated, especially in the local minimum points. This difference happens because our algorithm does not have enough data to learn the complex shape function.

4.2. Applying the Algorithm

As can be appreciated in Figure 5a, the cost function of the air impingement presents a smooth shape, similar to the functions analyzed in the previous section. Because of this, the algorithm presents an excellent way to optimize air jet impingement design. The comparison between the results of both algorithms is presented in Table 7. The addition of the stop criteria shows a similar result to the standard algorithm. The mode of the number of iterations needed to optimize the function is 50. The neural network is capable of learning the cost function after the first 50 iterations of the optimization algorithm. Besides, because the algorithm converges quickly, the solution obtained is good enough. It is also similar to the solution obtained by the GA without the stop condition. Figure 5b and Table 11 show the accuracy of the model. The accuracy is adequate to approximate the cost function.

Finally, the solution obtained satisfies the design criteria. It makes the fan works at a high efficiency point and reduces the temperature to a low point in the curve.

5. Conclusions

A genetic algorithm optimization is developed in the current work. The points found by the algorithm are used in the training of a neural network model that predicts the cost function. The performance of the training of this ANN is used as a stop criterion for the GA.

The new approach is tested with a Flower pollination algorithm. The lack of diversity and the static population of FPA algorithm shows as a problem for the training process. In order to achieve good training data, the optimization algorithm must have dynamic population between iteration and a high exploration ability.

The new criterion is shown as a good way to stop optimization algorithms that have a smooth cost function, which is the case of the air impingement design. At the same time, the algorithm is capable of learning the cost function. Therefore, this new approach provides two advantages in one algorithm. Due to the requirement of diversity in the population, the number of particles needed to develop this new approach is higher than that of a classical GA. Besides, a better result is obtained due to the fact that the population was split into two subpopulations, one focused on the search task and

the other on the optimization. For future work with algorithms that aim to achieve more goals than only an optimization, the division of the population would be crucial.

The normalization of the data before the neural network training improves the results and allows the learning of different functions with only one algorithm. However, as the targets are not normalized, functions that have a large range are more difficult to model. In future works, this would be an excellent addition to the algorithm.

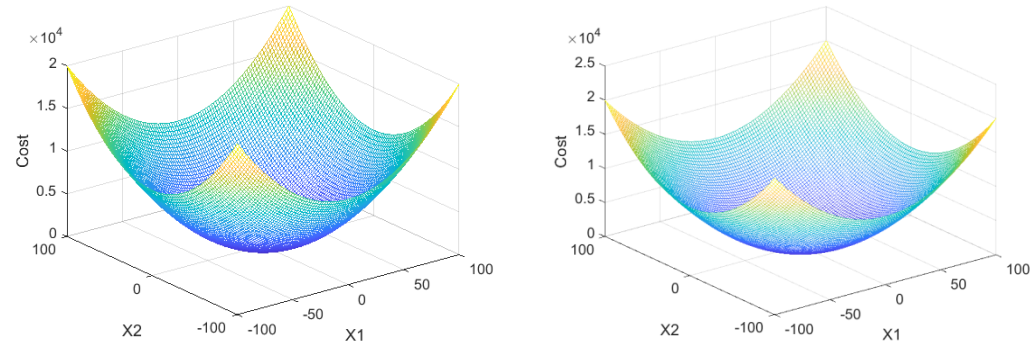
Finally the air impingement design is improved. With this new optimization process, the efficiency of the blower is taken into account, making it work at an operating point close to the optimum. In addition, along with optimization, the algorithm proves to be able to learn the cost function. However, the cooled surface in this article is larger than the existing ones in industrial equipment; therefore, in future work, it would be interesting to obtain similar results by working with smaller surfaces.

Author Contributions: P.M.-F. and U.F.-G. designed the thermohydraulic calculation procedure and test case; E.Z. and A.S.-C. developed the genetic algorithm and ran the computations. D.T.-F.-B. supervised the neural network implementation and performance. All authors were responsible for carrying out the post-processing of the resulting data and writing the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: The Regional Development Agency of the Basque Country (SPRI) is gratefully acknowledged for economic support through the research project "Refrigeración de dispositivos de alto flujo térmico mediante impacto de chorro" (AIRJET), KK-2018/00109, Programa ELKARTEK.

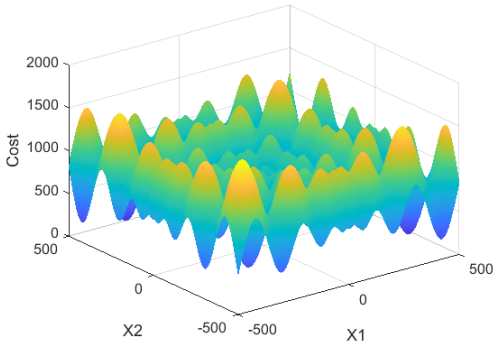
Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

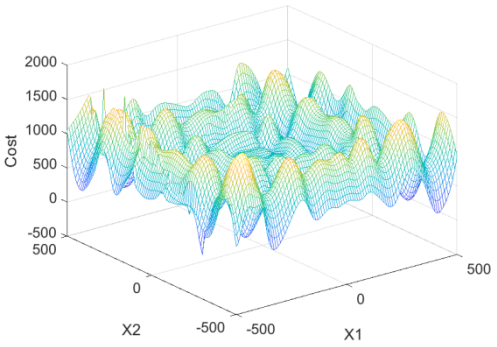


(a) Sphere function

(b) Sphere model

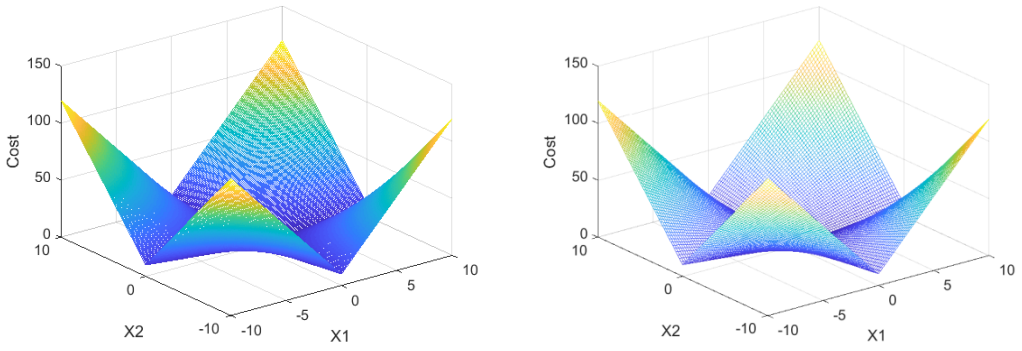


(c) Schwefel function

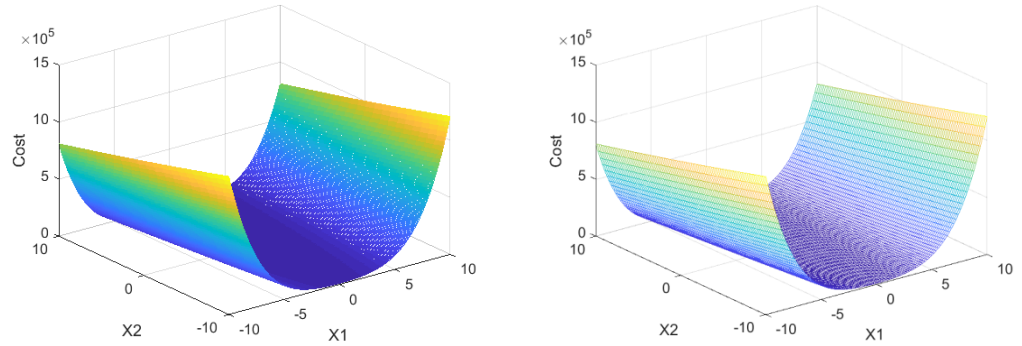


(d) Schwefel model

Figure A1. *Cont.*

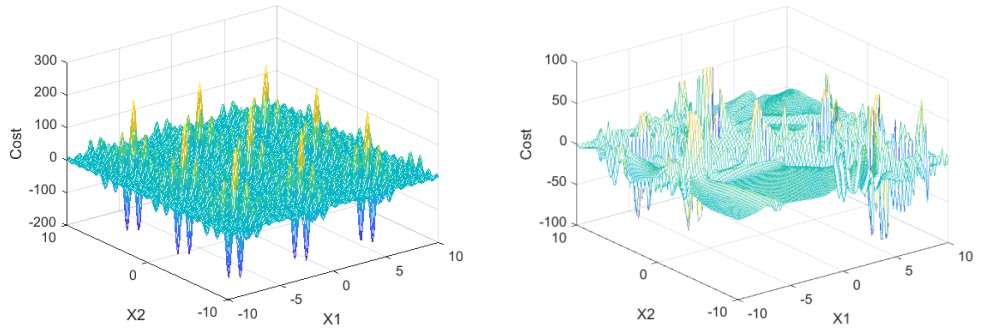


(e) Schwefel_P2 function (f) Schwefel_P2 model

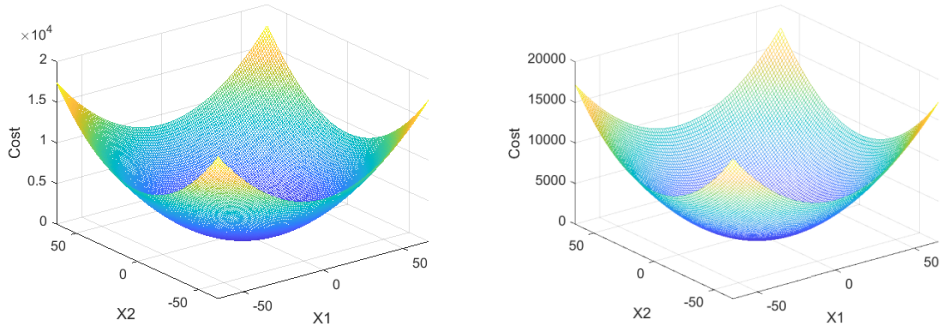


(g) Rosenbrock function (h) Rosenbrock model

Figure A1. Graphical representation of the first four functions (f1 to f4). The left column shows the real shape, and the right column shows the shape obtained with the model.

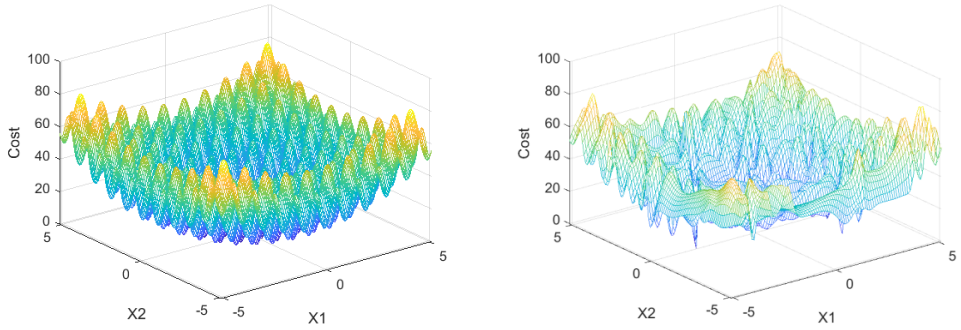


(a) Shubert function (b) Shubert model

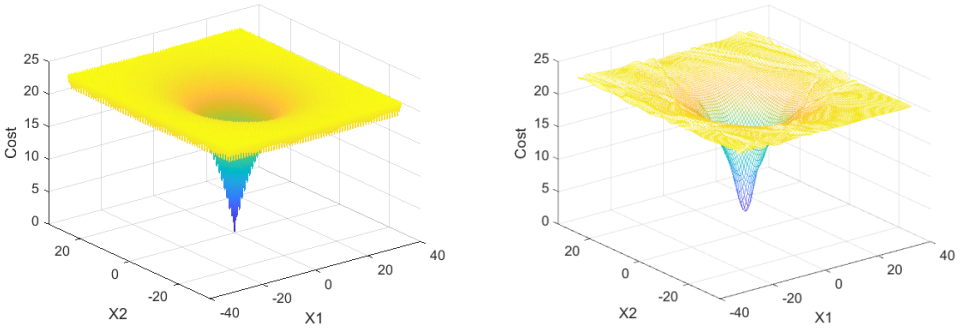


(c) Rothyp function (d) Rothyp model

Figure A2. *Cont.*



(e) Rastrigin function (f) Rastrigin model



(g) Ackley function (h) Ackley model

Figure A2. Graphical representation of the last four processes (f5 to f8). The left column shows the real shape, and the right column shows the shape obtained with the model.

References

1. Martinez-Filgueira, P.; Zulueta, E.; Sanchez-Chica, A.; Fernandez-Gamiz, U.; Soriano, J. Multi-Objective Particle Swarm Based Optimization of an Air Jet Impingement System. *Energies* **2019**, *12*, 1627. [CrossRef]
2. Wright, T.; Gerhart, P. Fluid Machinery: Application, Selection, and Design. In *Fluid Machinery: Application, Selection, and Design*, 2nd ed.; CRC Press: Boca Raton, FL, USA, 2009; p. 473.
3. Ingole, S.B.; Sundaram, K.K. Cold Zone Exploration using Position of Maximum Nusselt Number for Inclined Air Jet Cooling. *Arch. Mech. Eng.* **2017**, *64*, 533–549. [CrossRef]
4. Zhu, H.; Tang, T.; Zhao, H.; Gao, Y. Control of vortex-induced vibration of a circular cylinder using a pair of air jets at low Reynolds number. *Phys. Fluids* **2019**, *31*, 043603. [CrossRef]
5. Obot, N.T.; Trabold, T.A. Impingement Heat-Transfer within Arrays of Circular Jets: Part 1—Effects of Minimum, Intermediate, and Complete Cross-Flow for Small and Large Spacings. *J. Heat Transf. ASME* **1987**, *109*, 872–879. [CrossRef]
6. Gardon, R.; Cobonpue, J. Heat Transfer Between a Flat Plate and Jets of Air Impinging on It. *Int. Dev. Heat Transf. ASME* **1962**, 454–460. [CrossRef]
7. Jain, M.; Singh, V.; Rani, A. A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm Evol. Comput.* **2019**, *44*, 148–175. [CrossRef]
8. Abdel-Basset, M.; Shawky, L.A. Flower pollination algorithm: A comprehensive review. *Artif. Intell. Rev.* **2019**, *52*, 2533–2557. [CrossRef]
9. Mirjalili, S. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Comput. Appl.* **2016**, *27*, 1053–1073. [CrossRef]
10. Polap, D.; Wozniak, M. Polar Bear Optimization Algorithm: Meta-Heuristic with Fast Population Movement and Dynamic Birth and Death Mechanism. *Symmetry* **2017**, *9*, 203. [CrossRef]
11. Beasley, D.; Bull, D.R.; Martin, R.R. An overview of genetic algorithms: Part 1, fundamentals. *Univ. Comput* **1993**, *15*, 56–59.
12. Hou, Y.; Wu, N.; Zhou, M.; Li, Z. Pareto-Optimization for Scheduling of Crude Oil Operations in Refinery via Genetic Algorithm. *IEEE Trans. Syst. Man Cybern. Syst.* **2017**, *47*, 517–530. [CrossRef]

13. Wieczorek, M.; Lewandowski, M. A mathematical representation of an energy management strategy for hybrid energy storage system in electric vehicle and real time optimization using a genetic algorithm. *Appl. Energy* **2017**, *192*, 222–233. [CrossRef]
14. Ferdyn-Grygierek, J.; Grygierek, K. Multi-Variable Optimization of Building Thermal Design Using Genetic Algorithms. *Energies* **2017**, *10*, 1570. [CrossRef]
15. Gholamalizadeh, E.; Kim, M. Multi-Objective Optimization of a Solar Chimney Power Plant with Inclined Collector Roof Using Genetic Algorithm. *Energies* **2016**, *9*, 971. [CrossRef]
16. Keshanchi, B.; Souri, A.; Navimipour, N.J. An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: Formal verification, simulation, and statistical testing. *J. Syst. Softw.* **2017**, *124*, 1–21. [CrossRef]
17. Chen, W.; Panahi, M.; Pourghasemi, H.R. Performance evaluation of GIS-based new ensemble data mining techniques of adaptive neuro-fuzzy inference system (ANFIS) with genetic algorithm (GA), differential evolution (DE), and particle swarm optimization (PSO) for landslide spatial modelling. *CATENA* **2017**, *157*, 310–324. [CrossRef]
18. Ali, I.; Alharbi, O.M.L.; Alothman, Z.A.; Badjah, A.Y.; Alwarthan, A.; Basheer, A.A. Artificial neural network modelling of amido black dye sorption on iron composite nano material: Kinetics and thermodynamics studies. *J. Mol. Liq.* **2018**, *250*, 1–8. [CrossRef]
19. Peurifoy, J.; Shen, Y.; Jing, L.; Yang, Y.; Cano-Renteria, F.; DeLacy, B.G.; Joannopoulos, J.D.; Tegmark, M.; Soljacic, M. Nanophotonic particle simulation and inverse design using artificial neural networks. *Sci. Adv.* **2018**, *4*, eaar4206. [CrossRef]
20. Manuel Lopez-Guede, J.; Antonio Ramos-Hernanz, J.; Zulueta, E.; Fernandez-Gamiz, U.; Oterino, F. Systematic modeling of photovoltaic modules based on artificial neural networks. *Int. J. Hydrogen Energy* **2016**, *41*, 12672–12687. [CrossRef]
21. Saenz-Aguirre, A.; Zulueta, E.; Fernandez-Gamiz, U.; Lozano, J.; Manuel Lopez-Guede, J. Artificial Neural Network Based Reinforcement Learning for Wind Turbine Yaw Control. *Energies* **2019**, *12*, 436. [CrossRef]
22. Arena, P.; Fazzino, S.; Fortuna, L.; Maniscalco, P. Game theory and non-linear dynamics: the Parrondo Paradox case study. *Chaos Solitons Fractals* **2003**, *17*, 545–555. [CrossRef]
23. Liu, M.; Liu, G.; Joo, I.; Song, L.; Wang, G. Development of In Situ Fan Curve Measurement for VAV AHU Systems. *J. Sol. Energy Eng.* **2005**, *127*, 287–293. [CrossRef]
24. Xing, Y.; Spring, S.; Weigand, B. Experimental and numerical investigation of heat transfer characteristics of inline and staggered arrays of impinging jets. *J. Heat Transf.* **2010**, *132*, 092201. [CrossRef]
25. Attalla, M.A.M. *Experimental Investigation of Heat Transfer Characteristics from Arrays of Free Impinging Circular Jets and Hole Channels*; Otto von Guericke University Library: Magdeburg, Germany, 2005.
26. Lin, C.; Gao, F.; Bai, Y. An intelligent sampling approach for metamodel-based multi-objective optimization with guidance of the adaptive weighted-sum method. *Struct. Multidiscip. Optim.* **2018**, *57*, 1047–1060. [CrossRef]
27. Nisha, S. Review of Selection Methods in Genetic Algorithms. *Int. J. Eng. Comput. Sci.* **2017**, *6*, 22261–22263. Available online: https://www.ijecs.in/index.php/ijecs/article/download/2562/2368/ (accessed on 12 September 2019).
28. Abbas, Q.; Ahmad, J.; Jabeen, H. Tournament selection mechanism based random vector selection in differential evolution algorithm. *Int. J. Adv. Appl. Sci.* **2017**, *4*, 147–158. [CrossRef]
29. Optimization Test Functions and Datasets. Available online: https://www.sfu.ca/~{}ssurjano/optimization.html (accessed on 2 September 2019).
30. Molga, M.; Smutnicki, C. Test Functions for Optimization Needs. Available online: https://www.vafaeijahan.com/en/wp-content/uploads/2012/02/Test-functions-for-optimization-needs.pdf (accessed on 25 Novemeber 2019).
31. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
32. Evers, G.I.; Ghalia, M.B. Regrouping particle swarm optimization: A new global optimization algorithm with improved performance consistency across benchmarks. In Proceedings of the 2009 IEEE International Conference on Systems, Man and Cybernetics (IEEE SMC 2009), Hyatt Regency Riverwalk, San Antonio, TX, USA, 11–14 October 2009; pp. 3901–3908.