International Journal of Innovative Research and Scientific Studies, 8(2) 2025, pages: 4605-4616

IJIRSS
International Journal of Innovative Research & Scientific Studies

ISSN: 2617-6548

URL: www.ijirss.com

Artificial intelligence tools for modeling and optimizing solid-state fermentation processes

Hugo Beatriz-Cuellar[1], Isabel de la Luz Membrillo-Venegas[1], Jesus de la Cruz-Alejo[1*], Edwin Christian Becerra-Alvarez[2]

[1]*Tecnologico Nacional de México, Tecnológico de Estudios Superiores de Ecatepec, Ecatepec de Morelos, Mexico.*
[2]*Centro Universitario de Ciencias Exactas e Ingenierías, Universidad de Guadalajara, Departamento de Electro-Fotonica, Mexico.*

Corresponding author: Jesus de la Cruz-Alejo (*Email: jdelacruz@tese.edu.mx*)

Abstract

Solid-state fermentation (SSF) is a process used to produce enzymes and secondary metabolites; however, its low efficiency limits its application, as it does not cost-effectively meet market demand. This article proposes modeling the operation and determining the optimal parameters of SSF processes through the application of artificial intelligence systems. To this end, programming algorithms were designed in MATLAB software to implement an artificial neural network (ANN), a genetic algorithm (GA), particle swarm optimization (PSO), and the artificial bee colony (ABC) algorithm. To verify the proposed method, the production of proteases used in the cheese industry was modeled and optimized. The results show that the optimal process parameters were correctly identified, the modeling precision and accuracy were increased ($R^2 > 0.90$), and the process resources required can be reduced. These findings suggest that the use of artificial intelligence systems in SSF processes is an effective tool to maximize their production.

Keywords: Artificial bee colony algorithm, Artificial neural network, Genetic algorithm, Particle swarm optimization, Solid-state fermentation.

DOI: 10.53894/ijirss.v8i2.6404
Funding: This study was supported by Consejo Mexiquense de Ciencia y Tecnología, through the scholarship awarded to Hugo Beatriz-Cuellar (folio: EESP2024-0037).
History: Received: 5 March 2025 / **Revised:** 10 April 2025 / **Accepted:** 18 April 2025 / **Published:** 23 April 2025

Competing Interests: The authors declare that they have no competing interests.
Authors' Contributions: All authors contributed equally to the conception and design of the study. All authors have read and agreed to the published version of the manuscript.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.
Publisher: Innovative Research Publishing

1. Introduction

The industrial application of SSF processes, driven by market needs such as population growth, product availability, associated costs, product quality and other factors, makes it essential to improve their efficiency, reduce costs and increase their competitiveness [1]. Typically, in a SSF process using solid substrates to produce enzymes and secondary metabolites [2, 3] low efficiency and productivity are present, which limits its application as it does not cost-effectively meet market

requirements. Nowadays, several methodologies have been proposed to optimize a SSF process, such as controlling the culture conditions (incubation temperature, humidity, aeration, etc.), the appropriate selection of the microorganism and the substrate, and the application of experimental design techniques to determine the optimal culture conditions [4-6]. Furthermore, SSF can be analyzed and modeled using mathematical, kinetic, and thermodynamic approaches. However, this represents a complex task due to the heterogeneous characteristics of the fermentation process [7-9]. Currently, response surface methodology (RSM) is the most popular technique for determining optimal parameter values that minimize or maximize the response of a fermentation process [10-12]. However, this methodology cannot model nonlinear or very complex systems; it can only model one process response at a time, and, as the number of independent variables increases, the modeling accuracy decreases [13]. On the other hand, artificial intelligence techniques are systems inspired by diverse situations in nature, which allow for improved precision in the analysis, design, and control of processes [14, 15]. In this sense, an ANN is a computational model composed of multiple processing units (artificial neurons) inspired by the nervous system of living beings and that can be used to model and control processes. Additionally, there are various optimization and search techniques, such as GA, which is based on the principles of genetics and natural selection; PSO, which is inspired by the social and collective behavior of organisms in a colony; and ABC, which is based on the way bees forage and communicate information with each other [16]. In the literature, ANN, GA, PSO and ABC have been used in several applications with excellent results, such as finger gesture recognition [17] estimation or forecasting of electrical energy production through solar panels installed in a given area [18] intrusion detection to ensure the security, integrity and reliability of the Internet of Things [19] design of polymers with high thermal stability at high temperatures, resistance to high electric fields and high dielectric strength [20] design and optimization of the geometry of horizontal-axis wind turbine blades [21] estimation of medium and long-term energy consumption for planning and taking appropriate measures [22] extraction of maximum power from a photovoltaic system by tracking the maximum power point [23] among others. Therefore, this study presents the modeling and optimization of SSF processes through the design and implementation of artificial intelligence systems, as a basis for manipulating process variables toward optimal parameters that allow for increased efficiency and/or productivity. This is because these systems have shown good results in various applications and, consequently, can reduce resource use in the fermentation process. To achieve this objective, an ANN was used to model the relationship between the independent and dependent variables of the process. In addition, GA, PSO, and ABC were used to determine the optimal parameters that maximize the production of the fermentation process. To verify the performance of the artificial intelligence systems used, an SSF process was modeled and optimized. In this regard, the optimal parameters for protease production in *Rhizomucor miehei* were determined. In addition, the corresponding programming algorithms were designed in C++ to implement the artificial intelligence techniques in software, depending on the system conditions. The structure of the article is as follows: Section 2 describes the characteristics of the system (ANN, GA, PSO, and ABC), Section 3 presents the results obtained, Section 4 includes the discussion, and the last section presents the conclusions.

2. System Description

Figure 1 shows a block diagram of an SSF process, modeled with an ANN and optimized using GA, PSO, or ABC. A prerequisite for the application of advanced control algorithms is the use of well-characterized process models. This is because a mathematical model must represent both the static and dynamic behavior of a process and can be used for the design of the different design stages (simulation, controller design, etc.). To model the fermentation process, the first step is to define the number of independent and dependent variables, as well as their respective maximum and minimum values. The second step is to perform a design of experiments (such as the Box-Behnken design (BBD), central composite design (CCD), etc.), which is used to study the effects of the independent variables on the dependent variables. The third step is to design the ANN structure, which determines the relationship between the independent and dependent variables to correctly simulate and predict process behavior. In addition, the model must be validated using different criteria (e.g., mean square error (MSE), coefficient of determination (R^2), etc.), which demonstrate the model's accuracy. To optimize the fermentation process, the first step is to define the optimization technique (GA, PSO, or ABC), which depends on the complexity of the problem, the runtime, the computational resources, among other factors. The second step is to define the variable or variables to be optimized. The third step involves combining ANN and the optimization technique to obtain the optimal process parameters.

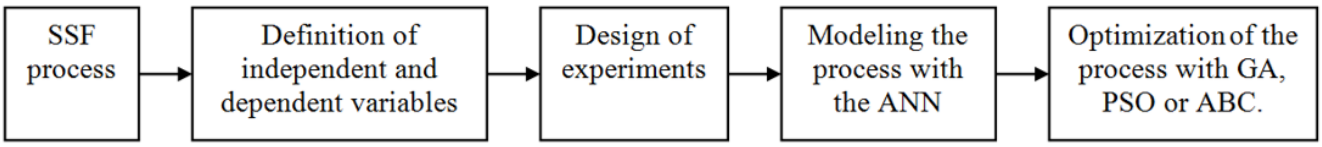

Figure 1.
Block diagram of the optimization of a fermentation process using artificial intelligence systems.

2.1. Design of the Artificial Neural Network

A prerequisite for the application of advanced control algorithms is the use of well-adapted process models. In this sense, an ANN is a system composed of several layers, which, in turn, are made up of artificial neurons. Figure 2 shows the structure of the ANN used to model the SSF process mentioned above. The ANN consists of an input layer (L_1) that receives data on the independent variables, a first hidden layer (L_2) with ten neurons, a second hidden layer (L_3) with ten neurons and an output layer (L_4) with five neurons that provides the responses of the fermentation process. In this case, X1,…,X5 and Y1,…,Y5 represent the independent and dependent variables of the process, respectively.

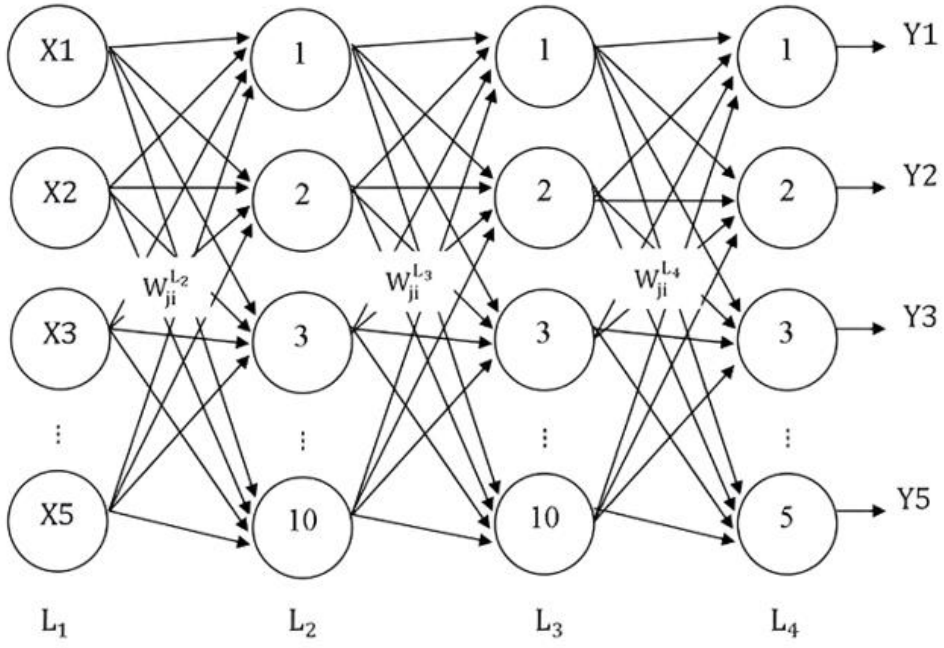

Figure 2.
Structure of the ANN for modeling protease production.

To obtain the relationship between the input and output variables, the synaptic weights ($W_{ji}^{(L)}$) and thresholds (θ_j) All the neurons in the network must be adjusted. To do this, the backpropagation algorithm was used, which uses a set of training samples, that is, the values of the input variables and their respective desired outputs. Training samples are obtained through an experimental design (obtained offline) and must be presented to the network until the error between the output and desired signals is within an acceptable range [24, 25]. The first stage of the backpropagation algorithm consists of propagating the value of the input variables layer by layer until the network response is obtained. For this purpose, (1) and (2) are used to determine the input and output of the j-th neuron in L_2, (3) and (4) to determine the input and output of the j-th neuron in L_3, and (5) and (6) to determine the input and output of the j-th neuron in L_4.

$$I_j^{(L_2)} = \sum_{i=1}^{m_1} \left(w_{ji}^{(L_2)} * X_i\right) - \theta_j \qquad m_1: independent\ variables \quad (1)$$

$$O_j^{(L_2)} = 1/\left(1 + e^{-I_j^{(L_2)}}\right) \quad (2)$$

$$I_j^{(L_3)} = \sum_{i=1}^{m_2} \left(w_{ji}^{(L_3)} * O_i^{(L_2)}\right) - \theta_j \qquad m_2: neurons\ in\ L_2 \quad (3)$$

$$O_j^{(L_3)} = \frac{1}{\left(1 + e^{-I_j^{(L_3)}}\right)} \quad (4)$$

$$I_j^{(L_4)} = \sum_{i=1}^{m_3} \left(w_{ji}^{(L_4)} * O_i^{(L_3)}\right) - \theta_j \qquad m_3: neurons\ in\ L_3 \quad (5)$$

$$O_j^{(L_4)} = \frac{1}{\left(1 + e^{-I_j^{(L_4)}}\right)} \quad (6)$$

The second stage of the backpropagation algorithm consists of propagating the error from the output layer to the input layer, passing through the intermediate layers of the network. For this, (7) is used, which quantifies the difference between the response of the neurons in the output layer and the desired output (d_j).

$$E = \frac{1}{2}\sum_{j=1}^{m_4} \left(d_j - O_j^{(L_4)}\right)^2 \qquad m_4: nuerons\ in\ L_4 \quad (7)$$

Subsequently, the gradient descent algorithm is applied to Equation 7, based on the synaptic weights and the network thresholds, obtaining (8), (10) and (12) to adjust the synaptic weights, and (9), (11), and (13) to adjust the thresholds.

$$w_{ji}^{(L_4)} = w_{ji}^{(L_4)} + \eta * \left(\left(d_j - O_j^{(L_4)}\right) * g'\left(I_j^{(L_4)}\right)\right) * O_j^{(L_3)} \qquad \eta: learning\ rate \quad (8)$$

$$\theta_j^{(L_4)} = \theta_j^{(L_4)} + \eta * \left(\left(d_j - O_j^{(L_4)}\right) * g'\left(I_j^{(L_4)}\right)\right) * (-1) \quad (9)$$

$$W_{ji}^{(L_3)} = W_{ji}^{(L_3)} + \eta\left(\sum_{k=1}^{m_4} \left(\left(d_k - O_k^{(L_4)}\right) * g'\left(I_k^{(L_4)}\right) * W_{kj}^{(L_4)}\right)\right) * g'\left(I_j^{(L_3)}\right) * O_j^{(L_2)} \quad (10)$$

$$\theta_j^{(L_3)} = \theta_j^{(L_3)} + \eta\left(\sum_{k=1}^{m_4}\left(\left(d_k - O_k^{(L_4)}\right) * g'\left(I_k^{(L_4)}\right) * W_{kj}^{(L_4)}\right)\right) * g'\left(I_j^{(L_3)}\right) * (-1) \quad (11)$$

$$W_{ji}^{(L_2)} = W_{ji}^{(L_2)} + \eta\left(\sum_{k=1}^{m_3}\left(\left(\sum_{k=1}^{m_4}\left(d_k - O_k^{(L_4)}\right) * g'\left(I_k^{(L_4)}\right) * W_{kj}^{(L_4)}\right) * g'\left(I_j^{(L_3)}\right)\right) * W_{kj}^{(L_3)} * g'\left(I_j^{(L_2)}\right)\right) * X_i \quad (12)$$

$$\theta_j^{(L_2)} = \theta_j^{(L_2)} + \eta\left(\sum_{k=1}^{m_3}\left(\left(\sum_{k=1}^{m_4}\left(d_k - O_k^{(L_4)}\right) * g'\left(I_k^{(L_4)}\right) * W_{kj}^{(L_4)}\right) * g'\left(I_j^{(L_3)}\right)\right) * W_{kj}^{(L_3)} * g'\left(I_j^{(L_2)}\right)\right) * (-1) \quad (13)$$

To evaluate the fit of the model obtained using the ANN, the R^2 were used. Finally, Appendix A shows the structure of the programming algorithm in C++ language to implement the ANN in software.

2.2. Genetic Algorithm (GA) Design

Optimization algorithms are generally used to find the minimum point of a function; however, to obtain the maximum point, it is sufficient to multiply the objective function value by (-1). This was done in the optimization techniques employed in this work. The GA consists of the stages of evaluation, selection, pairing, crossing and mutation [26, 27]. The independent variables of the fermentation process must be set according to the structure of a chromosome. $Chrom_i = [X_1 \;\; X_2 \;\; \dots \;\; X_m]$, where m indicates the number of independent variables, and whose value is randomly generated to cover the entire experimental region. Each chromosome represents a possible solution to the problem. Then, a population of chromosomes, $Pop\_chrom = [Chrom_1, Chrom_2, \dots, Chrom_N]$, is generated, where *N* indicates the number of chromosomes that make up the population. The evaluation stage determines the process response based on the independent variables of a chromosome using ANN. The selection stage selects the most suitable chromosomes from the population, which are those that generate the lowest values of the objective function. Generally, the most suitable chromosomes make up 50% of the population, while the rest are eliminated. The matching stage determines a set of chromosomes called parents, denoted as Chrom_dad and Chrom_mom. To do this, the roulette method is used, which consists of ordering the chromosomes from lowest to highest according to the response of the objective function. Then, the probability and cumulative probability of each of the best-fit chromosomes are determined using (14) and (15), respectively. Both parameters depend on the position p of the best-fit chromosomes. To select a parent chromosome, a random number is generated and the best-fit chromosome whose Cumu_prob value is greater than the random number is selected.

$$Pb_i = \frac{N_{ch} - p + 1}{\sum_{p=1}^{N_{ch}} p} \qquad N_{ch}: number\ of\ most\ suitable\ chromosomes \quad (14)$$

$$cumu\_prob = \sum_{i=1}^{N_{ch}} Pb_i \quad (15)$$

The crossover stage generates a set of chromosomes called descendants, represented by Chrom_desc, which are obtained by combining the independent variables that make up Chrom_dad and Chrom_mom, producing two Chrom_desc. To achieve this, the following is defined: a crossover point (position of a variable on both chromosomes), a new value for said crossover point (*C1* and *C2*) is determined using (16) and (17), and the values of the variables of the chromosomes that are located to the right of the crossover point are exchanged. In this way, the new population is composed of the most suitable chromosomes and the descendant chromosomes.

$$variable_{new1} = variable_{C1} - \beta(variable_{C2} - variable_{C1}) \qquad \beta: Random\ number\ [0,1] \quad (16)$$

$$variable_{new2} = variable_{C2} + \beta(variable_{C2} - variable_{C1}) \quad (17)$$

The mutation stage modifies the value of the independent variables of some chromosomes in the new population. The number of mutations is determined by (18). To perform the mutations, the independent variables of the affected chromosomes are randomly selected. In this way, the steps described above are defined as a generation of the GA and are repeated until the optimal process parameters are reached. Finally, Appendix A shows the structure of the programming algorithm in C++ language to implement the GA in software.

$$Mutat = (0.20) * (N) * (m) \quad (18)$$

2.3. Particle Swarm Optimization (PSO)

PSO uses a set of particles (*SP*), which must explore the experimental region until one of them finds the global optimal position (minimum point). To do this, each particle, during its movement, learns both from the others (social behavior) and from its own experience (cognitive behavior). To start the search process, the number of independent process variables (D) and their respective minimum and maximum values are defined. Each particle i has two vectors associated with it: the first describes its position $X_i = X_{i1}, X_{i2}, \dots, X_{iD}$, while the second describes its velocity $V_i = V_{i1}, V_{i2}, \dots, V_{iD}$ [28, 29]. For this purpose, the values of the components of X_i and V_i are initialized randomly to cover the experimental region. In each iteration, the ANN is used to evaluate the objective function based on the independent variables of a particle i. With the results obtained, the best solution found by each particle ($Pbest_i^{t+1}$) must be selected. To do this, the current response is compared with the previous response of a particle i, as indicated in (19). Also, the best position found by the swarm ($Gbest^t$) is determined using (20).

$$Pbest_i^{t+1} = \begin{cases} Pbest_i^t & si \quad f(X_i) > Pbest_i^t \\ X_i & si \quad f(X_i) \le Pbest_i^t \end{cases} \qquad t: iteration\ number \quad (19)$$

$$Gbest^t = min\{Pbest_1^t, Pbest_2^t, \dots, Pbest_{NS}^t\} \quad (20)$$

The change of the velocity of a particle i depends on the following parameters: $Gbest^t$, $Pbest_i^t$, V_i^t, two social parameters (r_1 and r_2), and two random numbers (c_1 and c_2), as given in (21). Meanwhile, the change of position of a particle i depends on the variables X_i^t and V_i^t, as shown in (22). This procedure represents an iteration and should be repeated until the optimal process parameters are obtained. Finally, Appendix A shows the structure of the programming algorithm in C++ language to implement PSO in software.

$$V_i^{t+1} = W_{iner} * V_i^t + c_1 * r_1 * (Pbest_i^t - X_i^t) + c_2 * r_2 * (Gbest^t - X_i^t) \quad (21)$$

$$X_i^{t+1} = X_i^t + V_i^t \quad (22)$$

2.4. Artificial Bee Colony Algorithm (ABC)

The ABC algorithm determines the optimal parameters of a process using three types of bees: employee bees, observer bees, and scout bees [30, 31]. To initiate the search process, the number of independent process variables (D) and their respective levels, both maximum (x^{max}) and minimum (x^{min}) are defined. Scout bees generate a set of food sources (FS) whose position $X_i = x_{i,1}, \dots, x_{i,D}$ represents a solution to the optimization problem. The value of the j-th dimension of X_i is determined by (23).

$$x_{i,j} = x_j^{min} + rand(0,1)\left(x_j^{max} - x_j^{min}\right) \quad i = 1,2, \dots, FS \quad (23)$$

Each bee employed must analyze the position of a food source X_i to generate a new food source position $V_i = v_{i,1}, v_{i,2}, \dots, v_{i,D}$, by modifying only one parameter of X_i. To do this, a food source X_k is randomly chosen, different from the analyzed food source. In addition, a dimension of X_i must be randomly chosen, using (24) to generate a new value of the j-th dimension of V_i.

$$V_{i,j} = x_{i,j} + \emptyset_{i,j}\left(x_{i,j} - x_{k,j}\right) \quad \emptyset_{i,j} is\ a\ random\ number\ [-1, 1] \quad (24)$$

Next, ANN is used to determine the objective function (f_i) response of V_i and X_i. If the quality of V_i is better than that of X_i, then V_i will replace X_i in the population. To do this, the fitness value of food sources is determined using (25).

$$fit_i = \begin{cases} \frac{1}{1+f_i} & si\ f_i \geq 0 \\ 1 + |f_i| & si\ f_i < 0 \end{cases} \quad (25)$$

Scout bees use range selection to select a food source from among the other bees employed. To do this, the probability of the food sources is determined by (26). The probabilities are then ordered from highest to lowest and compared to a random number between 0 and 1. If the random number is greater than p_i, then a new food source is generated using (24). Furthermore, if the quality of V_i is better than X_i, then V_i replaces X_i in the population.

$$p_i = \frac{fit_i}{\sum_{i=1}^{SN} fit_i} \quad (26)$$

If, after performing the above process several times, a food source does not improve, it will be eliminated. In this case, a scout bee generates a new food source using (23). To determine whether a food source should be eliminated, a counter records the number of times a food source is visited by both employed and observer bees. If the counter reaches its limit, the food source is eliminated. Finally, Appendix A shows the structure of the programming algorithm in the C++ language to implement ABC in software.

2.5. Desirability Function

The SSF process analyzed has more than one response; therefore, the desirability function, which correlates multiple responses into a single value, was used [32]. With it, several responses can be optimized, either by minimizing some or maximizing others. To do this, the desirability of each response ($d_1, \dots, d_n$) is determined and a set of weights ($w_1, \dots, w_n$) is defined, as shown in (27). The desirability value of a response is obtained by (28). The value of the weights depends on the importance of each response in the process and must be between 0 (inadequate response) and 1 (ideal response).

$$DE = w_1 d_1 + w_2 d_2 + \cdots + w_5 d_5 \quad (27)$$

$$d_n = \begin{cases} \frac{y_i - y_{min}}{y_{max} - y_{min}} & Maximize\ response \\ \vdots & \\ \frac{y_{max} - y_i}{y_{max} - y_{min}} & Minimize\ response \end{cases} \quad (28)$$

3. Results

To validate the functionality of the artificial intelligence tools used, a process was analyzed using parametric tests, employing the RSM and the proposed method. In this sense, the production of proteases for the cheese industry was carried out using the microorganism *Rhizomucor miehei*, and wheat bran, wheat flour, soybeans and corn were used as substrates [33]. In this process, the responses analyzed were milk coagulation activity (Y1), specific coagulation activity (Y2), proteolytic activity (Y3), specific proteolytic activity (Y4) and the ratio between milk coagulation activity and proteolytic activity (Y5). However, Y2 and Y5, which are the most important responses of the fermentation process, were optimized. For this process, the microorganism was grown on potato dextrose agar (PDA) slant plates and stored at 4°C for use. To prepare the inoculum, *Rhizomucor miehei* was inoculated into 90-mm Petri dishes with 20 mL of PDA and incubated at 37°C for 5 days. The inoculum was then obtained by adding 30 mL of distilled water and scraping the surface of the PDA, all under sterile conditions. For the preparation of the fermentation medium, corn and soybeans were ground and sieved to obtain a

particle size of 1 mm. A mineral solution composed of 0.007 g/L of $ZnSO_4$ $7H_2O$, 0.007 g/L of $MgSO_4$ $7H_2O$, 0.07 g/L of $CuSO_4$ $7H_2O$ and 0.09 g/L of $FeSO_4$ was prepared. To adjust the humidity of the substrates, ten milliliters of this solution diluted in one liter of distilled water was used to complete the mineral solution. Then, 20 grams of wet substrate were placed in 250 mL Erlenmeyer flasks, which were autoclaved for 20 min at 121 °C. To inoculate the medium, a spore suspension (10^6 spores/mL) was used. The modeling and optimization of the fermentation process were carried out with the RSM in Minitab 17 software, which used a second-order polynomial to model the process and a CCD to analyze it, considering five independent variables: fermentation time (X1), temperature (X2), pH (X3), moisture content (X4) and nitrogen concentration (X5). Table 1 shows the characteristics of the CCD independent variables. To optimize the process and consequently obtain the maximum value of Y2 and Y5, Y1 production must be maximized and Y2 production minimized. Using RSM, R^2 values of 0.8536 and 0.5783 were obtained for Y3 and Y5, respectively. Additionally, it was determined that the optimal parameters are X1 = 81.21 h, X2 = 41.11 °C, X3 = 6.31, X4 = 80% and X5 = 1.33%, through which Y1=2258.13 soxhlet units/mL, Y2=5.11 mg/mL, Y3=441.90 soxhlet units/mg, Y4=1.14 protease units/mg and Y5=388.66 can be obtained.

Table 1.
Characteristics of the independent variables used in the CCD.

Variable	Units	Coded level				
		-α	**-1**	**0**	**+1**	**+α**
X1	Hour	24	48	72	96	120
X2	°C	30	35	40	45	50
X3	-	3	4	5	6	7
X4	% v/w	40	50	60	70	80
X5	% v/w	0.5	1	1.5	2	2.5

For the proposed method, the process was modeled using an ANN composed of five inputs (X1, X2, X3, X4, and X5), two hidden layers with ten neurons each, and one output with five neurons (Y1, Y2, Y3, Y4, and Y5). With this process configuration, R^2 values of 0.98894 and 0.97761 were obtained for Y3 and Y5, respectively. This indicates greater precision than the classical method described above. Table 2 shows the results of the CCD, the RSM, and the ANN. MATLAB™ software was used for the software implementation of the ANN, GA, PSO, and ABC, as well as for obtaining the response surfaces and contour plots. For the optimization techniques, 50 chromosomes and 30 generations were used for the GA, 50 particles and 30 iterations for the PSO, and 50 bees and 30 cycles for the ABC. Table 3 shows the optimal parameters obtained from the optimization techniques used and from RSM. As can be seen, the value of Y1 obtained from the RSM is outside the range of the experimental results of the process, which may indicate an incorrect result. Furthermore, an increase in the final values of Y2, Y3, and Y4 is predicted. Another very important aspect is that a reduction in the use of process resources can be achieved, since lower values of X1, X2, X3, X4 and X5 are required compared to RSM. Finally, Figures 3(a), 3(b), 3(c) and 3(d) show the variation of Y3 as a function of X1, X2, X3, X4 and X5, using response surface plots and contour plots. These figures allow to visualization of the areas of interest where the Y3 values are high, low, or constant. Furthermore, Figure 3(e) shows the number of iterations required for the convergence of the optimization techniques, and as can be seen, these systems quickly determine the maximum throughput of the process.

Table 2.
Experimental results of BBD, RSM and ANN results in TH-PR production.

X1	X2	X3	X4	X5	Y3			Y5		
					Experimental	RSM	ANN	Experimental	RSM	ANN
48	35	4	50	1	59.61	10.792	58.234	64.52	19.284	64.674
96	35	4	50	1	30.84	52.7	23.521	139.98	87.252	141.14
48	45	4	50	1	122.21	78.14	122.79	158.37	105.22	158.77
96	45	4	50	1	24.46	50.496	28.236	66.36	126.63	64.738
48	35	6	50	1	109.29	224.9	112.19	130.24	152.82	130.28
96	35	6	50	1	372.44	295.51	365.84	257.45	181.52	257.19
48	45	6	50	1	342.5	294.24	345.36	261.15	175.56	263.25
96	45	6	50	1	221.48	295.3	215.36	154.44	157.7	154.84
48	35	4	70	1	92.48	69.072	91.274	145.8	73.52	145.53
96	35	4	70	1	133.05	145.06	132.95	185.2	183.82	184.73
48	45	4	70	1	162.74	174.42	163.29	196.67	171.66	197.26
96	45	4	70	1	144.1	180.86	143.7	205.06	235.4	204.94
48	35	6	70	1	192.11	175.98	194.21	163.18	201.46	161.14
96	35	6	70	1	195.76	280.67	199.95	344.83	272.5	339.23
48	45	6	70	1	248.02	283.32	247.56	303.13	236.4	301.23
96	45	6	70	1	374.4	318.46	371.54	279.02	260.88	277.8
48	35	4	50	2	15.34	11.154	21.146	133.33	70.784	132.25
96	35	4	50	2	31.6	98.374	32.429	72.46	114.75	71.977
48	45	4	50	2	20.49	106.6	25.751	52.79	134.52	50.293
96	45	4	50	2	172.55	124.27	170.74	189.56	131.93	188.26
48	35	6	50	2	203.32	219.66	198.46	247.45	230.32	247.63
96	35	6	50	2	292.99	335.58	293.24	233.04	235.02	232.7
48	45	6	50	2	344	317.11	348.91	286.45	230.86	287.13
96	45	6	50	2	282.92	363.48	284.33	164.65	189	163.6
48	35	4	70	2	8.3	53.434	8.0563	19.49	75.82	20.57
96	35	4	70	2	138.39	174.73	141.75	146.08	162.12	146.29
48	45	4	70	2	177.47	186.88	176.99	179.3	151.76	180.15
96	45	4	70	2	244.55	238.63	241.43	215.83	191.5	214.86
48	35	6	70	2	143.56	154.74	144.3	396.83	229.76	392.15
96	35	6	70	2	256.01	304.74	254.19	221.82	276.8	221.65
48	45	6	70	2	251.64	290.19	255.02	154.22	242.5	151.24
96	45	6	70	2	275.32	370.64	269.42	243.26	242.98	253.15
24	40	5	60	1.5	0	13.073	2.2729	0	87.672	6.1551
120	40	5	60	1.5	95.31	135.43	94.786	219.61	156.12	220.5
72	30	5	60	1.5	9.25	19.799	5.0484	53.48	119.84	54.684
72	50	5	60	1.5	110.7	153.04	113.25	214.35	171.96	214.26
72	40	3	60	1.5	31.75	74.565	27.151	150.38	100	150.99
72	40	7	60	1.5	410.84	420.68	409.65	209.15	285.01	208.39
72	40	5	40	1.5	315.68	375.7	320.01	158.87	199.2	159.04
72	40	5	80	1.5	448.28	441.14	447.56	323.88	307.41	324.63
72	40	5	60	0.5	113.37	238.85	117.76	102.04	205.3	103.36
72	40	5	60	2.5	363.96	291.39	378.45	318.18	238.9	317.62
72	40	5	60	1.5	211.13	296.82	279.08	311	241.7	255.71
72	40	5	60	1.5	294.25	296.82	279.08	212.43	241.7	255.71
72	40	5	60	1.5	273.96	296.82	279.08	275.2	241.7	255.71
72	40	5	60	1.5	324.93	296.82	279.08	209.23	241.7	255.71
72	40	5	60	1.5	286.42	296.82	279.08	273.23	241.7	255.71

Table 3.
Results obtained from RSM and artificial intelligence tools.

Method	X1	X2	X3	X4	X5	Y1	Y2	Y3	Y4	Y5
RSM	81.21	41.11	6.31	80.00	1.33	2258.13	5.11	441.900	1.14	388.66
GA	74.740	38.909	4.2581	80.00	1.232	1541.3	5.6891	448.200	1.988	342.25
PSO	74.594	38.891	4.263	80.000	1.2276	1541.2	5.6899	448.190	1.9883	338.37
ABC	74.583	38.896	4.260	80.000	1.228	1541.2	5.6901	448.190	1.988	339.14

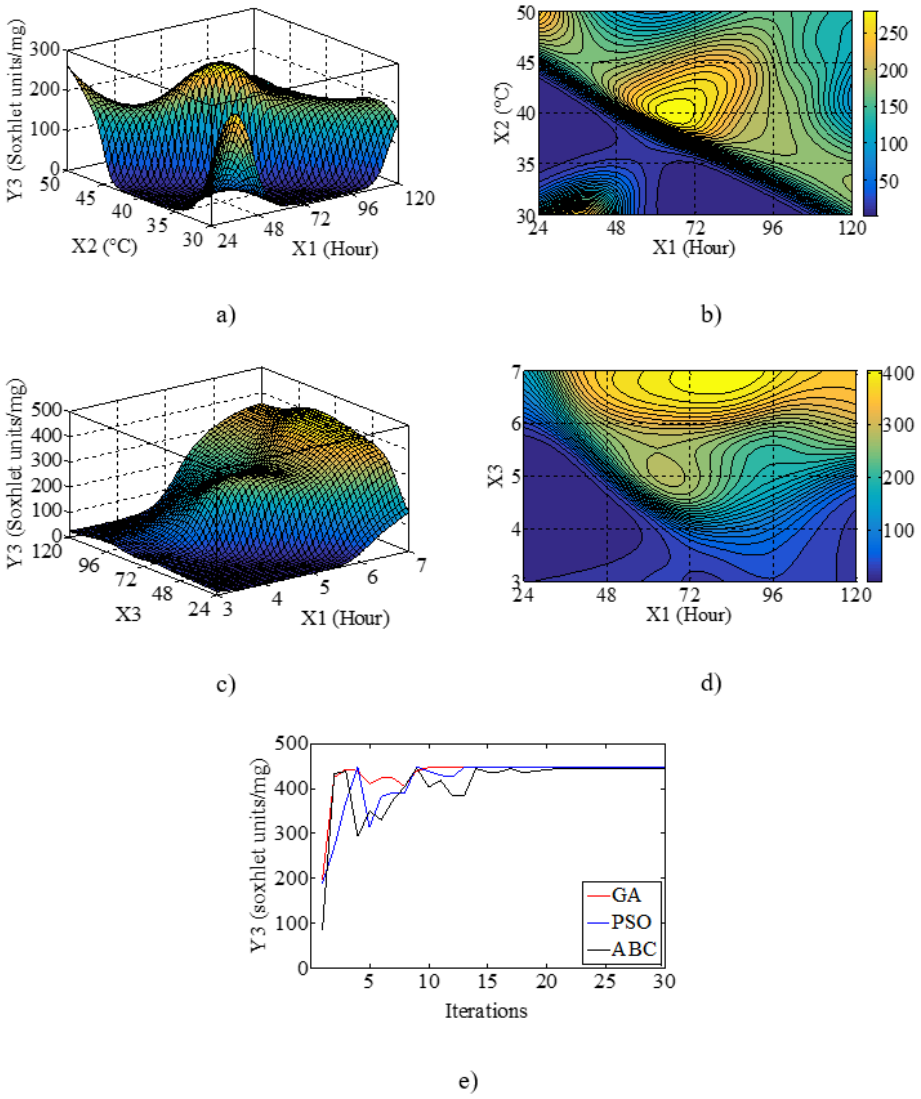

Figure 3.
Results obtained from protease optimization.

4. Discussion

The proposed method has demonstrated that artificial intelligence can manipulate variables in systems that encompass a wide variety of processes, which require the integration of components and information processing during their design. In this regard, it was observed that its application in SSF processes demonstrates that an ANN can model systems with multiple inputs and outputs with a high degree of accuracy, based on the results of an experimental design. To achieve this, it is only necessary to adjust the number of layers and neurons in the network. Furthermore, GA, PSO, and ABC correctly determined the optimal parameter values for the independent process variables, regardless of the characteristics of the response surface. This allows the values of the independent variables to be appropriately adjusted to increase production. In this case, only the number of iterations and the number of system elements (chromosomes, particles, or bees) need to be defined to determine the optimal process values. Another important point is that the implementation of these types of systems, that is, theoretical and experimental modeling, hardware simulation, and verification, only requires methods and tools developed with software or hardware compatible with the C++ language. Due to the characteristics of ANN, GA, PSO, and ABC, it is possible to model and optimize highly complex systems (both linear and nonlinear), regardless of the number of independent or dependent variables, without compromising model accuracy. Furthermore, system cost is reduced, as there is no need to purchase additional software, as all elements can be implemented using MATLAB software.

5. Conclusions

In this work, the analysis and characterization of artificial intelligence systems for modeling and optimizing SSF processes were presented, focusing specifically on optimal values for process parameters, using ANN, GA, PSO, and ABC. In this context, the ANN was able to model fermentation processes with a high degree of accuracy, regardless of the number of input variables, the number of output variables, or the shape of the response surface. Furthermore, GA, PSO, and ABC successfully determined the optimal values for the process parameters and manipulated them until the desired values were achieved. By adjusting these variables, an approach is presented for managing and controlling the most important process parameters. Furthermore, to implement these artificial intelligence algorithms, only data from an experimental design is

required, without the need for complex mathematical equations, and C++ software is sufficient. However, the study may have limitations, since if the microorganism and substrate are not properly selected, the fermentation process yield will be low. This is because these elements are the most important. Furthermore, if the experimental tests (runs of the design of experiments) are not performed correctly, regardless of the methods used, the modeling and optimization will be flawed.

References

[1] C. R. Chilakamarry *et al.*, "Advances in solid-state fermentation for bioconversion of agricultural wastes to value-added products: Opportunities and challenges," *Bioresource Technology*, vol. 343, p. 126065, 2022. https://doi.org/10.1016/j.biortech.2021.126065

[2] L. Yafetto, "Application of solid-state fermentation by microbial biotechnology for bioprocessing of agro-industrial wastes from 1970 to 2020: A review and bibliometric analysis," *Heliyon*, vol. 8, no. 3, p. e09173, 2022. https://doi.org/10.1016/j.heliyon.2022.e09173

[3] H. Chen, *Modern solid state fermentation*. Dordrecht, Netherlands: Springer 2013.

[4] S. C. Araujo *et al.*, "Optimization of lipase production by Penicillium roqueforti ATCC 10110 through solid-state fermentation using agro-industrial residue based on a univariate analysis," *Preparative biochemistry & biotechnology*, vol. 52, no. 3, pp. 325-330, 2022. https://doi.org/10.1080/10826068.2021.1944203

[5] U. Perwitasari *et al.*, "Cacao pod husk for citric acid production under solid state fermentation using response surface method," *Biomass Conversion and Biorefinery*, pp. 1-9, 2021. https://doi.org/10.1007/s13399-021-01690-9

[6] F. V. Do Nascimento, A. M. de Castro, A. R. Secchi, and M. A. Z. Coelho, "Insights into media supplementation in solid-state fermentation of soybean hulls by Yarrowia lipolytica: Impact on lipase production in tray and insulated packed-bed bioreactors," *Biochemical Engineering Journal*, vol. 166, p. 107866, 2021. https://doi.org/10.1016/j.bej.2020.107866

[7] I. Caro Pina, C. Marzo Gago, A. B. Díaz Sánchez, and A. M. Blandino Garrido, "Modelling and optimization of simultaneous saccharification and fermentation of agro-food residues," *Journal of Environmental Chemical Engineering*, vol. 12, no. 1, p. 111862, 2024. https://doi.org/10.1016/j.jece.2023.111862

[8] Y. Ge, L. Li, and L. Yun, "Modeling and economic optimization of cellulosic biofuel supply chain considering multiple conversion pathways," *Applied Energy*, vol. 281, p. 116059, 2021. https://doi.org/10.1016/j.apenergy.2020.116059

[9] W. Arthur, D. Diedericks, G. Coetzee, E. Van Rensburg, and J. F. Görgens, "Kinetic modelling of cellulase recycling in paper sludge to ethanol fermentation," *Journal of Environmental Chemical Engineering*, vol. 9, no. 5, p. 105981, 2021. https://doi.org/10.1016/j.jece.2021.105981

[10] P. Leite *et al.*, "Recent advances in production of lignocellulolytic enzymes by solid-state fermentation of agro-industrial wastes," *Current Opinion in Green and Sustainable Chemistry*, vol. 27, p. 100407, 2021. https://doi.org/10.1016/j.cogsc.2020.100407

[11] P. Leite, I. Belo, and J. M. Salgado, "Co-management of agro-industrial wastes by solid-state fermentation for the production of bioactive compounds," *Industrial Crops and Products*, vol. 172, p. 113990, 2021. https://doi.org/10.1016/j.indcrop.2021.113990

[12] Ç. Gönen, E. Ü. Deveci, and N. Akter Önal, "Evaluation of biomass pretreatment to optimize process factors for different organic acids via Box–Behnken RSM method," *Journal of Material Cycles and Waste Management*, vol. 23, no. 5, pp. 2016-2027, 2021. https://doi.org/10.1007/s10163-021-01276-7

[13] L. H. S. De Menezes *et al.*, "Artificial neural network hybridized with a genetic algorithm for optimization of lipase production from Penicillium roqueforti ATCC 10110 in solid-state fermentation," *Biocatalysis and Agricultural Biotechnology*, vol. 31, p. 101885, 2021. https://doi.org/10.1016/j.bcab.2020.101885

[14] O. A. Omitaomu and H. Niu, "Artificial intelligence techniques in smart grid: A survey," *Smart Cities*, vol. 4, no. 2, pp. 548-568, 2021. https://doi.org/10.3390/smartcities4020029

[15] N. Ghaffar Nia, E. Kaplanoglu, and A. Nasab, "Evaluation of artificial intelligence techniques in disease diagnosis and prediction," *Discover Artificial Intelligence*, vol. 3, no. 1, p. 5, 2023. https://doi.org/10.1007/s44163-023-00049-5

[16] M. G. Abdolrasol *et al.*, "Artificial neural networks based optimization techniques: A review," *Electronics*, vol. 10, no. 21, p. 2689, 2021. https://doi.org/10.3390/electronics10212689

[17] K. H. Lee, J. Y. Min, and S. Byun, "Electromyogram-based classification of hand and finger gestures using artificial neural networks," *Sensors*, vol. 22, no. 1, p. 225, 2021. https://doi.org/10.3390/s22010225

[18] J. M. Barrera, A. Reina, A. Maté, and J. C. Trujillo, "Solar energy prediction model based on artificial neural networks and open data," *Sustainability*, vol. 12, no. 17, p. 6915, 2020. https://doi.org/10.3390/su12176915

[19] J. Liu, D. Yang, M. Lian, and M. Li, "Research on intrusion detection based on particle swarm optimization in IoT," *IEEE Access*, vol. 9, pp. 38254-38268, 2021. https://doi.org/10.1109/ACCESS.2021.3063671

[20] C. Kim, R. Batra, L. Chen, H. Tran, and R. Ramprasad, "Polymer design using genetic algorithm and machine learning," *Computational Materials Science*, vol. 186, p. 110067, 2021. https://doi.org/10.1016/j.commatsci.2020.110067

[21] A. Pourrajabian, M. Dehghan, and S. Rahgozar, "Genetic algorithms for the design and optimization of horizontal axis wind turbine (HAWT) blades: A continuous approach or a binary one?," *Sustainable Energy Technologies and Assessments*, vol. 44, p. 101022, 2021. https://doi.org/10.1016/j.seta.2021.101022

[22] D. Özdemir, S. Dörterler, and D. Aydın, "A new modified artificial bee colony algorithm for energy demand forecasting problem," *Neural Computing and Applications*, vol. 34, no. 20, pp. 17455-17471, 2022. https://doi.org/10.1007/s00521-022-07675-7

[23] C. González-Castaño, C. Restrepo, S. Kouro, and J. Rodriguez, "MPPT algorithm based on artificial bee colony for PV system," *Ieee Access*, vol. 9, pp. 43121-43133, 2021. https://doi.org/10.1109/ACCESS.2021.3066281

[24] G. R. Yang and X.-J. Wang, "Artificial neural networks for neuroscientists: A primer," *Neuron*, vol. 107, no. 6, pp. 1048-1070, 2020. https://doi.org/10.1016/j.neuron.2020.09.005

[25] I. N. Da Silva *et al.*, *Artificial neural network architectures and training processes*. Springer. https://doi.org/10.1007/978-3-319-43162-8, 2017.

[26] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools and Applications*, vol. 80, pp. 8091-8126, 2021. https://doi.org/10.1007/s11042-020-10139-6

[27] R. L. Haupt and S. E. Haupt, *Practical genetic algorithms*. New Jersey: Wiley-Interscience, 2004.

[28] A. G. Gad, "Particle swarm optimization algorithm and its applications: A systematic review," *Archives of Computational Methods in Engineering*, vol. 29, no. 5, pp. 2531-2561, 2022. https://doi.org/10.1007/s11831-021-09694-4

[29] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh, and S. Mirjalili, "Particle swarm optimization: A comprehensive survey," *Ieee Access*, vol. 10, pp. 10031-10061, 2022. https://doi.org/10.1109/ACCESS.2022.3142859

[30] B. Akay, D. Karaboga, B. Gorkemli, and E. Kaya, "A survey on the artificial bee colony algorithm variants for binary, integer and mixed integer programming problems," *Applied Soft Computing*, vol. 106, p. 107351, 2021. https://doi.org/10.1016/j.asoc.2021.107351

[31] R. Durgut and M. E. Aydin, "Adaptive binary artificial bee colony algorithm," *Applied Soft Computing*, vol. 101, p. 107054, 2021. https://doi.org/10.1016/j.asoc.2020.107054

[32] J. P. Park, J. H. Jo, and Y. E. Nahm, "A desirability function-based multi-characteristic robust design optimization technique," *Journal of Society of Korea Industrial and Systems Engineering*, vol. 46, no. 4, pp. 199–208, 2023. https://doi.org/10.11627/jksie.2023.46.4.199

[33] H. A. Aljammas, S. Yazji, and A. Azizieh, "Optimization of protease production from Rhizomucor miehei Rm4 isolate under solid-state fermentation," *Journal of Genetic Engineering and Biotechnology*, vol. 20, no. 1, p. 82, 2022. https://doi.org/10.1186/s43141-022-00358-9

Appendix A. C++ programming algorithms for the implementation of artificial intelligence tools.

General structure of the algorithm in C++ to implement the ANN in MATLAB™

```
% Initialize the value of the synaptic weights randomly
epochs = 100000; samples = 47; eta=0.075;  %eta = η
for k1=1:1: epochs
for k2=1:1: samples  % samples are CCD data.
% Determine the input and output of the neurons in L2 (Equations 1 and 2)
I2j1=(w1ji10*X0)+(w1ji11*X1)+ … +(w1ji15*X5);  Y2j1=1/(1+exp(-I1j1));
% Determine the input and output of the neurons in L3 (Equations 3 and 4)
I3j1=(w2ji10*X0)+(w2ji11*Y1j1)+ … +(w2ji110*Y1ji10);  Y3j1=1/(1+exp(-I2j1));
% Determine the input and output of the neurons in L4 (Equations 5 and 6)
I4j1=(w3ji10*X0)+(w3ji11*Y2j1)+ … +(w3ji110*Y2ji10);  Y4j1=1/(1+exp(-I3j1));
%Update the synaptic weights of L4 (Equation 8)
error31=(dj1-Y3j1)*(Y3j1*(1-Y3j1));  w3ji11=w3ji11+(eta*error31*Y2j1);
%Update the synaptic weights of L3 (Equation 9)
error21=((error31*w3ji11)+…+(error35*w3ji51))*(Y2j1*(1-Y2j1));  w2ji11=w2ji11+(eta*error21*Y1j1);
%Update the synaptic weights of L2 (Equation 10)
error11=((error21*w2ji11)+…+(error210*w2ji101))*(Y1j1*(1-Y1j1));    w1ji11=w1ji11+(eta*error11*X1);
end
end
```

General structure of the algorithm in C++ to implement PSO in MATLAB™

```
SP = 40; D=5; c1i=0.5; c1f=2.5;  c2i=2.5;  c2f=0.5; t=0;  tmax=iteration;  W_inermax=0.9; W_inermin=0.4;
iteration=30; Pbest_i=zeros(SP,1);  Z1=zeros(SP,1);  X_i=zeros(SP,D);  V_i=zeros(SP,D);
for k1=1:1:SP
%Randomly generate X_i and V_i. Use ANN to evaluate X_i; its response is denoted as YANN.
fila1 = fila1 + 1;  Pbest_i(fila1,columna1)=X_i(fila1,columna1);     Z1(fila1,columna1)=YANN;
end
for k2=1:1:SP  % Determine Gbest
fila1 = fila1 + 1;  ZZ1=Z1(fila1,columna1);
if ZZ1 <= Gbest1
Gbest1 = ZZ1;   Gbest_i = X_i(fila1,:);
end
end
for k3=1:1: iteracion  % Update particle speed and position
for k4=1:1:(SP*D)
fila1 = fila1 + 1;     r1=rand;     r2=rand;     W_iner=((W_inermax-W_inermin)*((T-t)/T))+W_inermin;
c1 = ((c1f - c1i)*(t/T)) + c1i;  c2 = ((c2f - c2i)*(t/T)) + c2i;
compo_inercia=((W_iner)*(V_i(fila1,col1)));   compo_cogniti=((c1*r1)*((Pbest_i(fila1,col1))-(X_i(fila1,col1))));
      compo_social=(c2*r2)*((Gbest_i(fila2,col1))-(X_i(fila1,col1)));
velocity=compo_inercia+compo_cogniti+compo_social;  % ecuacion 18
      V_i(fila1,col1) = velocity;    X_i = X_i(fila1,col1) + V_i(fila1,col1);
 end
for k5=1:1:(SP)  % Determine the value of Pbest_i of each particle
% Zx_i and Zpbest are ANN responses based on the values of X_i and Pbest_i, respectively
Z1(fila1,1)=Zx_i;
if Zx_i <= Zpbest
Pbest_i(fila1,:) = X_i(fila1,:);
End
```

```
 end
   for k6=1:1:SP  % Determine Gbest
      fila1 = fila1 + 1;  ZZ1=Z1(fila1,1);
      if ZZ1 <= Gbest1
         Gbest1 = ZZ1;   Gbest_i = X_i(fila1,:);
      end
   end
end
```

General structure of the algorithm in C++ to implement GA in MATLAB™

```
Generation=30; N=50; m=5;  n=0;  Pb_acu=0;  Pop_chrom=zeros(N,m);  Z1=zeros(N,1);  Opt_chrom=25;
Pop_chrom_opti=zeros(Opt_chrom,m);  Z1_opti=zeros(N,1);  Probability=zeros(Opt_chrom,2);
Parents=zeros(Opt_chrom,1);
for k1=1:1:Npop  % Randomly generate the values of the chromosome variables
   fila1 = fila1 + 1;  x1=rand;  Vdesnor = ((VmaxE1-VminE1)*(x1)) + VminE1;  Pop_chrom(fila_val1,columna1) =
Vdesnor;
end
for k2=1:1:Generation
   for k3=1:1:N  % Evaluate each chromosome of Pop_chrom
      % Use ANN to evaluate X_i; its response is denoted as YANN.
      Z1(fila_val1,columna1)=z1;
   end
   costORDER = sort(Z1,'descend');  % Order the chromosomes according to the ANN response
   for k12=1:1:Opt_chrom  % Save the fittest chromosomes
      fila1 = fila1 + 1;  Pop_chrom_opti(fila1,:)=Pop_chrom(fila1,:);  Z1_opti(fila1,columna1)=Z1(fila1,columna1);
   end
   for k13=1:1:Nkeep  % Determine the probability of range
      n=n+1;  Pb=(Nkeep-n+1)/(Nkeep*(Nkeep+1)/(2));  Pb_acu=Pb_acu+Pb;  Probability(n,columna1)=Pb;
      Probability(n,columna2)=Pb_acu;
   end
   for k14=1:1:Opt_chrom  % Determine the parent chromosomes
      fila1 = fila1 + 1;  Parent_val = rand;
      for k15=1:1:Opt_chrom
         fila2 = fila2 + 1;  Prob_parent=Pn_acum(fila2,columna2);
         if Parent_val < Prob_parent
            cont=cont + 1;
            if cont == 1
               Parents(fila1,columna1)=fila2;
            end
         end
      end
      fila_val2=0; cont=0;
   end
   pos_cruce=(m - 1)*(rand) + 1;  pos_cruce=round(pos_cruce);  beta=rand;     % Select the crossover point of a
chromosome
   if pos_cruce == 1  %Parents(fila_val1,1);
      mother=Parents(fila_val1,1);  father=Parents(fila_val2,1);
      parent_new1=mother-((beta)*(mother-father));  parent_new2=father+((beta)*(mother-father));
   end
   var_mutated=round((0.20)*(N-1)*(m));  valmax_col=m;  valmin_col=0.5;  valmax_row=N;  valmin_row=2;
      for k21=1:1:var_mutated
         muta_row=round(((valmax_row-valmin_row)*(rand)) + valmin_row);  muta_col=round(((valmax_col-
valmin_col)*(rand)) + valmin_col);
         new_valor=((VmaxE1-VminE1)*(rand)) + VminE1;  Pop_chrom(muta_row,muta_col)=new_valor;
      end
end
```

General structure of the algorithm in C++ to implement ABC in MATLAB™

```
FS = 50; D=5;   X_i=zeros(SP,D);  Z1=zeros(FS,1);  Z2=zeros(FS,1); TRIALS=zeros(FS,1);  Z3=zeros(FS,1);
V_i=zeros(SP,D);
for k1=1:1:(FS*D)
%Randomly generate X_i and V_i. Use ANN to evaluate X_i; its response is denoted as YANN.
   fila1 = fila1 + 1;  Xij=Xminj+(rand)*(Xmaxj-Xminj);  X_i(fila1,col_val1) = Xij;
```

```
end
for k10=1:1:ciclos  % Produce a new food source Vij from Xij
   fila1 = fila1 + 1;
   valor1=(NumVariables - 1)*(rand) + 1;   valor_J=round(valor1);
   valor2=(Num_abejas-1)*(rand)+1;  valor_K=round(valor2);
   Oij = ((1-(-1))*(rand)) + (-1); % Random number between -1 and 1
   Xij=X_i(fila1,valor_J);  % valor_J is the jth dimension of X_i

   Xkj=X_i(valor_K,valor_J);  % valor_K is the k-th food source
   Vij = Xij + Oij*(Xij-Xkj);  V_i(fila_val1,valor_J)=Vij;
   for k2=1:1:Num_abejas
      % Determine the response of the ANN based on the variables of X_i
      fit_i = 1/(1+z1);  %z1 is the response of the ANN
      Z1(fila1,columna1)=z1;   Z1(fila1,columna2)=fit_i;
      % Determine the response of the ANN based on the variables of V_i
      fit_i = 1/(1+z1);  %z1 es la respuesta de la ANN
      Z2(fila1,columna1)=z1;  Z2(fila1,columna2)=fit_i;
   end
   for k5=1:1:Num_abejas  % Greedy selection process
      fila1 = fila1 + 1;
      R1=Z1(fila_val1,columna2);  R2=Z2(fila_val1,columna2);
      if R2 >= R1
         valXX3(fila1,:)=V_i(fila1,:);  Z3(fila1,:)=Z2(fila1,:);  TRIALS(fila1,columna1)=0;
      else
         valXX3(fila1,:)=X_i(fila1,:);  Z3(fila1,:)=Z1(fila1,:);  TRIALS(fila1,columna1)= 1 + TRIALS(fila1,columna1);
      end
   end
   Fit_Fun=Z3(:,2);   costORDE = sort(Fit_Fun,'descend');   Sel_Prob=zeros(FS,1);
   t=t+1; n=Num_abejas; k=0;  fila1 = 0;
   for k7=1:1:Num_abejas  % Ranking Selection
      k = k + 1;    fila1 = fila1 + 1;
      at=0.2+((3*t)/(4*N));  Pk=(1/n)+((at)*((n+1-(2*k))/(n*(n+1))));  Sel_Prob(fila1,1)=Pk;
   end
   for k6=1:1:FS
      r1=rand;
      for k7=1:1:FS
         fila2 = fila2 + 1;  Pi=Sel_Prob(fila2,columna1);
         if r1 > Pi
            % A new food source is generated and compared to the current food source. If it is better, the process described
above is performed.
         end
      end
   end
   fila1 = 0;  abejaExplo = 0;
   for k9=1:1:Num_abejas  % Scout Bee Phase
      fila1 = fila1 + 1;   prue_abej=TRIALS(fila1,columna1);
      if prue_abej >= trialslimit
         abejaExplo = abejaExplo + 1;
         if abejaExplo <= 1
            Xij=VminE1+(rand)*(VmaxE1-VminE1);   TRIALS(fila_val1,columna1)=0;
         end
      end
   end
end
```