REVISTA
SAN GREGORIO
ISSN: 1390 - 7247 // e-ISSN: 2528 - 7907

Revista San Gregorio. 2025; 1:Especial 2
doi: 10.36097/rsan.v1iEspecial_2.2826

ARTÍCULO ORIGINAL

Identificación de parámetros en sistemas ecuaciones diferenciales ordinarias mediante el uso de redes neuronales artificiales

Identification of parameters in ordinary differential equation systems using artificial neural networks

Francisco Javier Duque-Aldaz[1] Fernando Raúl Rodríguez-Flores[2] José Carmona Tapia[3]

[1]Universidad de Guayaquil, Ecuador.
[2]Universidad de la Habana, Cuba.
[3]Universidad de Almería, España.

Citar como: Duque-Aldaz, F.J., Rodríguez-Flores, F.R., y Carmona, J. (2024). Identificación de parámetros en sistemas ecuaciones diferenciales ordinarias mediante el uso de redes neuronales artificiales. *Revista San Gregorio*, 1(Especial_2), 15-23. http://dx.doi.org/10.36097/rsan.v1iEspecial_2.2826

Recibido: 14-02-2024 **Aceptado:** 01-08-2024 **Publicado:** 15-02-2025

RESUMEN

El objetivo de este estudio fue comparar una red neuronal artificial entrenada con Backpropagation y optimizada mediante Levenberg-Marquardt contra métodos numéricos clásicos para identificar parámetros en ecuaciones diferenciales ordinarias. Se diseñó una red neuronal multicapa con una entrada, una capa oculta de 10 neuronas y dos salidas. El modelo se entrenó con datos experimentales divididos en conjuntos de entrenamiento, validación y prueba, utilizando el algoritmo Levenberg-Marquardt para ajustar sus parámetros. La precisión se evaluó comparando con el método numérico ODE45, basado en Runge-Kutta. La red neuronal demostró un rendimiento superior, logrando una aproximación precisa y menos compleja computacionalmente. El método ODE45 presentó buenos ajustes generales, mostró limitaciones en intervalos específicos debido a picos y discontinuidades en las funciones simuladas. La red neuronal exhibió robustez para manejar dinámicas no lineales, prediciendo con alta precisión el comportamiento del sistema sin requerir un modelo matemático explícito. Su capacidad para reconocer patrones complejos con márgenes de error tolerables la consolidó como una herramienta eficaz para sistemas dinámicos. Las redes neuronales artificiales se confirmaron como una alternativa metodológica robusta, permitiendo modelar sistemas no lineales dinámicos con simplicidad, flexibilidad y potencial de escalabilidad.

Palabras clave: Métodos numéricos; ajuste de parámetro; redes neuronales artificiales; algoritmo Backpropagation; método Levenberg-Marquardt.

ABSTRACT

The objective of this study was to compare an artificial neural network trained with Backpropagation and optimized using Levenberg-Marquardt against classical numerical methods for identifying parameters in ordinary differential equations. A multilayer neural network was designed with one input, a hidden layer of 10 neurons, and two outputs. The model was trained with experimental data divided into training, validation, and test sets, using the Levenberg-Marquardt algorithm to adjust its parameters. Accuracy was evaluated by comparing it with the ODE45 numerical method, based on Runge-Kutta. The neural network demonstrated superior performance, achieving an accurate approximation with lower computational complexity. The ODE45 method provided good overall fits but showed limitations in specific intervals due to peaks and discontinuities in the simulated functions. The neural network exhibited robustness in handling nonlinear dynamics, accurately predicting the system's behavior without requiring an explicit mathematical model. Its ability to recognize complex patterns with acceptable error margins established it as an effective tool for dynamic systems. Artificial neural networks were confirmed as a robust methodological alternative, enabling the modeling of nonlinear dynamic systems with simplicity, flexibility, and scalability potential.

Keywords: Numerical methods; parameter tuning; artificial neural networks; Backpropagation algorithm; Levenberg-Marquardt method.

INTRODUCCIÓN

Numerical methods allow solving mathematical problems using numerical operations with the aid of the computer (Diamessis, 1965). Mathematical models can generally be expressed by equations or inequalities in partial or algebraic derivatives. The calculated numbers displayed in computer graphics reflect the view of reality provided by the chosen computational process (Hari Rao & Yadaiah, 2005).

The problem of parameter fitting or parameter identification in the various branches of engineering has been gaining importance until it has become of great interest to many (Sage & Melsa, 1971). The purpose of parameter identification is to develop or improve mathematical representations of physical systems using experimental data (Dennis & Schnabel, 1983). In this way, an attempt is made to bridge the gap between the real world and that of mathematical models that purport to represent the real world, in a way that helps to better understand the former and improve the latter.

Numerical methods are characterized by transforming the systems of differential equations of the control problem into a system of algebraic equations, in such a way that they depend on a limited number of unknowns (Chintha & Chatterjee, 2022).

In our daily life we are surrounded by phenomena that can be simulated, modeled, or represented by dynamical systems. For example, mathematical biology, ecology (Jovanović et al., 2022) or epidemiology (Guo et al., 2022), in chemical reactions (de Souza et al., 2013), when trying to understand and predict interactions between different species, substances, processes; multidimensional dynamical systems with different parameters can be found (Zhang et al., 2021). These systems may depend on certain parameters, such as reaction rate, time, position, density, or equilibrium constants (van Ophem & Desmet, 2022).

Therefore, by knowing the dynamics of these systems, we will be able to predict values of a certain magnitude to respond to real-world problems or needs, so that we can anticipate and reduce the impact of possible negative effects (Yan et al., 2021).

The probabilistic characterization of the latter is obtained from input-output databases collected over a long period of time. The latter is not difficult to substantiate in industrial chemical processes, where ergodic processes generally occur. On the other hand, in such industrial processes deterministic disturbances occur naturally, say for example in heat exchangers that ensure fluid temperature values which feed the consumers of the fluid itself and the number of consumers cannot be predicted in advance, but the delivered flow value can (Meidani & Barati Farimani, 2023).

Another example of perturbations acting on such processes can be the supply of reactants to continuous chemical reactors, where the reactants are not controlled (regulated) and therefore can vary, remaining constant at given time intervals (Lee et al., 2023). Similarly, the supply of raw materials in general can vary in quality as they come from different sources. This problem, together with the fact that in the presence of high temperature values there are decantation of minerals contained in the fluids and incrustations on the inner surface of the fluid conduits (pipelines) make the parameters of the process models do not coincide with the theoretical parameters of the mathematical models (generally differential equations or systems of differential equations) obtained from the physical-chemical laws that govern the behavior of the processes (De-kui & Xing-min, 2023).

One way to approach these problems is through experimentation with the industrial processes of interest. This requires performing experiments where it is possible to measure the stimuli applied to the process and the reactions to such stimuli. These measurements provide a data set or numerical database by measuring the different process variables. The database obtained and stored in the computational media is used to estimate the parameters of the proposed models.

The purpose of this research is to compare the performance of an artificial neural network trained with Backpropagation and optimized by Levenberg-Marquardt against classical numerical methods, evaluating its ability to identify parameters in systems of ordinary differential equations.

Artificial Neural Networks ANNs

Some of the research works in the field of artificial neural networks are event prediction and simulations (Lastre Aleaga et al., 2015), recognition and classification (González-Medina & Vázquez, 2015), data processing and modeling (Miranda et al., 2020), control engineering (Sarmiento-Ramos, 2020), artificial intelligence (Erazo Yánez & Navarrete Cedillo, 2023).

Figueredo and Ballesteros (2016) highlighted the fundamental characteristics of artificial neural networks, highlighting their structure composed of multiple processing elements that simulate the functioning of biological neurons. The authors pointed out that these computational elements are distinguished by establishing weighted connections among themselves, enabling a distributed representation of information. The distinctive feature of these systems lies in their ability to implement a learning process that allows them to acquire knowledge progressively, adapting dynamically to the input data.

Artificial Neural Network Model

Artificial neural networks are forecasting methods based on non-complex or simple mathematical models. These models allow complex nonlinear relationships to be realized between the output or response variable(s) and their predictors (Sancho Caparrini, 2017).

An Artificial Neural Network generally consists of three major parts: an input layer, with units representing the input fields ; one or more hidden layers; and an output layer, with a unit or units representing the target fields . The units are connected with different connection strengths (or weights) . Input data are presented to the first layer and values are propagated from each neuron to each neuron in the next layer. Eventually, a result is delivered from the output layer.

Figure 1 shows an example of the comparison of the biological neural model with the artificial neural network model, comparing each of their parts.

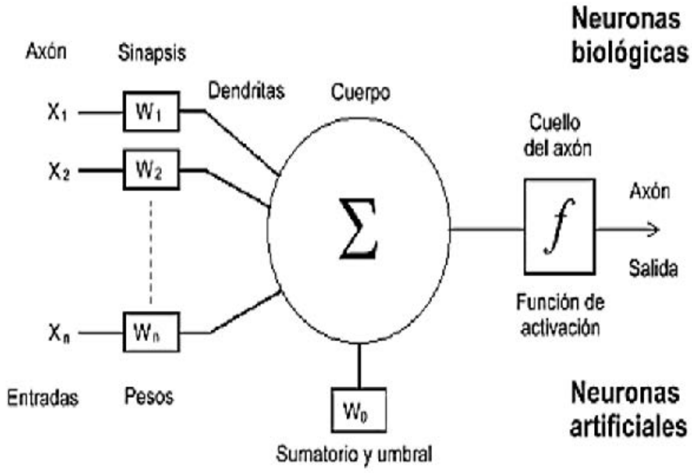

Figure 1. *Artificial neural network model.*

Table 1 provides a systematic deconstruction of neural elements, mapping the intrinsic components of biological neural systems against their algorithmic representations in artificial neural networks, revealing the sophisticated biomimetic strategies employed in contemporary computational modeling.

Table 1. *Relationship between biological neural network and artificial neural network.*

Biological neural network	Artificial neural network
Dendrites	Inputs
Cell nucleus	Nodes
Synapses	Weights
Axon	Output

Levenberg-Marquardt Optimization Method

The Levenberg-Marquardt LM method, initially devised for nonlinear parameter estimation problems, has additionally proven to be useful for the solution of linear ill-conditioned problems (Binu & Rajakumar, 2021). The Levenberg-Marquardt algorithm is a hybrid technique that uses Gauss-Newton and steepest descent approaches to converge to an optimal solution. The hybrid approach is often used to trade off the best features of different algorithms to solve a wider range of problems (Modest, 2003).

When the performance function is in the form of a sum of squares (as is typical in feedforward network training), then the Hessian matrix can be approximated as:

$$H = J(x_c)^T J(x_c)$$

And the gradient can be calculated as:

$$g = J(x_c)^T R(x_c)$$

The Levenberg-Marquardt algorithm uses this approximation to the Hessian matrix in the after Newton-like update:

$$x_{c+1} = x_c - (J(x_c)^T J(x_c) + \mu_c I)^{-1} J(x_c)^T R(x_c)$$

The Levenberg-Marquardt algorithm may be the fastest method for training feedforward neural networks of moderate size (up to several hundred weights) (Raju P & Terzija, 2023). This algorithm has an efficient implementation in MATLAB software, because the solution of the matrix equation is a built-in function in the program, so its attributes become more robust in the MATLAB environment.

Backpropagation Algorithm for Neural Networks

The Backpropagation or backpropagation or backward error propagation algorithm is one of the most fundamental building blocks in a neural network (Sadeq et al., 2023). Rumelhart et al. (1986) popularized it in their paper "Learning representations by back propagation errors". The Backpropagation algorithm is a method used to train or find the weights in a multilayer feed-forward artificial neural network. Therefore, a network structure of one or more layers must be defined, where the previous layer is fully connected to the next layer. The Backpropagation algorithm can be used in both classification and regression problems (Wang et al. 2020).

The Backpropagation algorithm repeatedly adjusts the connection weights in the ANN to minimize a measure of the difference between the actual output vector of the network and the desired output vector. Backpropagation aims to minimize the cost function by adjusting the weights and biases of the network (Tarigan et al., 2017).

The training of the ANN with the Backpropagation algorithm follows a gradient descent approach that exploits the chain rule. The level of fit is determined by the gradients of the cost function with respect to these parameters (Vishwakarma et al., 2020).

The main features of Backpropagation are the iterative, recursive, and efficient method through which it calculates the updated weight to improve the network until it is unable to perform the task for which it is being trained.

METODOLOGÍA

Comparison of numerical methods to identify dynamic systems

A quantitative, applied, and experimental study was conducted. The research compared the behavior of neural networks with classical parameter fitting methods in ordinary differential equations (ODEs). The study utilized three data vectors: t, y1, y2, where y1 and y2 represented two functions of time.

The data for t vs y1 are presented in Figure 2 in black, while t vs y2 are illustrated in Figure 3 in black. The solution of the Cauchy problem in the integration interval [0,10], with specified initial values or Initial Value Problem, will be analyzed to evaluate the performance of different methodological approaches.

$$y_1(0) = 10\ ;\ y_2(0) = 10$$

The model was defined by a system of two nonlinear ordinary differential equations with unknown coefficients α and β:

$$\frac{dy_1}{dt} = y_2 - \alpha * (y_1 * y_2)/(1 + y_2 * y_2) \qquad (1)$$

$$\frac{dy_2}{dt} = -y_1 - \beta * y_2/(1 + y_1 * y_1) \qquad (2)$$

$$Y = \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} \in \mathbb{R}^2 \qquad (3)$$

$$\forall t \in [0,10] \qquad (4)$$

Behavior of the output variable y_1 of the process with respect to time

Figure 2 showed the fit between the experimental data represented by circles and the model obtained with ODE 45; a very good fit was observed at almost all points, except for the interval [7,9], where a jump peak was present, and the function lost its smoothness.

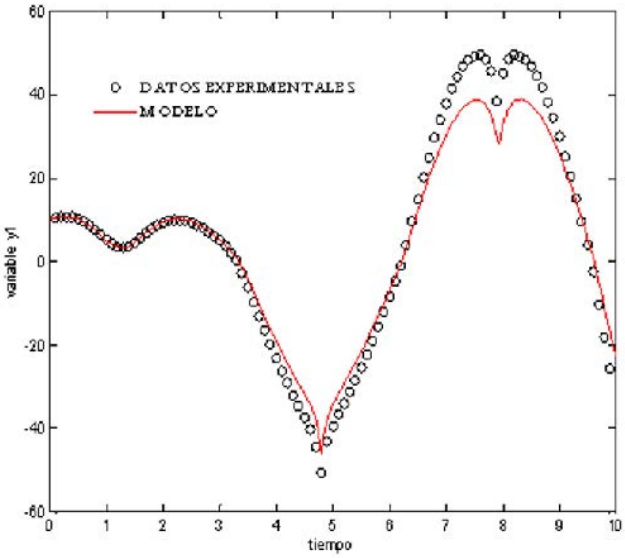

Figure 2. *Behavior of the output variable y_1 of the process with respect to time.*

Behavior of the output variable y_2 of the process with respect to time

Figure 3 revealed the fit between the experimental data represented by circles and the model obtained with ODE 45, evidencing a good fit with the exception of intervals [3,4], [6,7] and [9,10], where jumps occurred and the function did not maintain its continuity.

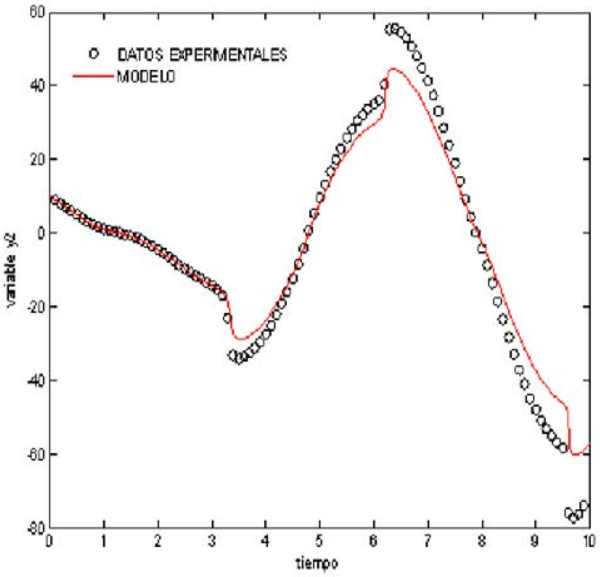

Figure 3. *Behavior of the output variable y_2 of the process with respect to time.*

The task consisted of finding the values of α and β that would define the mathematical model in the state space y_1 and y_2, considering the experimental database. This problem became known as the Parameter Configuration Problem in Differential Equations.

The problem was solved in MATLAB® using ODE45 and the optimization function fminsearch.

The ODE45 function was based on a Runge-Kutta type algorithm (4,5) developed from the Euler method improved by Dormand-Prince. It was a one-step method, i.e., to determine $x(t_{i+1})$, it was necessary to know only the solution at the immediately preceding time $x(t_i)$.

The fminsearch function, an optimizer of linear functions with constraints, found the value of the x variables that minimized the function described in fun, starting from an initial value at x0. The algorithm implemented was the Simplex method (Nelder & Mead, 1965). This method represented a direct search based on evaluations of the objective function, without using numerical or analytical gradients; it was suitable for functions that could present discontinuities.

The results of the behavior of the system defined by equations (1), (2), (3) and (4) after the estimation of the parameters α and β were presented in Figures 2 and 3 by continuous curves in red.

At the conclusion of the optimization processing, the values α = 5.0501 and β = -5.0998 were obtained with a convergence criterion of 1.000000 e-04.

Modeling with artificial neural networks

To obtain modeling using Artificial Neural Networks, the obtained database was used, which was divided into three data sets to train the neural network, validate the resulting model, and estimate the future behavior with the obtained network (Margaglio & Uria, 1994). The MATLAB environment was employed as a software platform.

The architecture of the Perceptron neural network was designed based on the premises outlined by Saldarriaga (2022):

- The activation function was chosen primarily according to the preferred path, and selecting one of them did not affect the network's ability to solve the problem.
- The number of neurons in the input layer, as well as in the output layer, was determined by the variable defining the problem. According to the experimental data, only one variable was present in the input of the network, while two variables were required at the output.
- The designer determined the number of hidden layers and the neurons in each layer. There was no specific method or rule to determine the optimal number of neurons in a hidden layer to solve such a problem. In most practical applications, these values were determined through trial and error.
- For the hidden layers, the least number was initially considered, and if the required settings could not be achieved, the number of hidden layers gradually increased.
- A single hidden layer was chosen to initiate the Perceptron network training. Several tuning tests were conducted during the experimentation to find a suitable network size. These tests included altering the number of neurons in the hidden layer using various configurations provided by the MATLAB Neural Network Toolbox.

Representation of an artificial neural network in terms of the ANN Toolbox

Figure 4 presented a representation of an artificial neural network using the MATLAB® Neural Network Toolbox nomenclature. The network was configured with a hidden layer employing a hyperbolic tangent activation function and an output layer with a linear activation function.

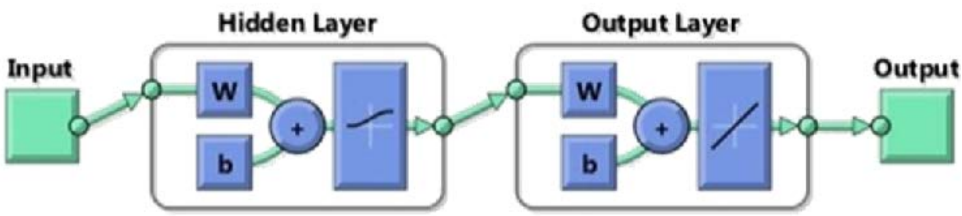

Figure 4. *ANN representation used with MatLab.*

For training the neural network, the Trainlm function (Levenberg-Marquardt backpropagation) was applied. This training function updated the weights and biases based on the Levenberg-Marquardt Optimization Method. It was identified as the fastest backpropagation algorithm available in the MATLAB® Toolbox in most cases and was recommended as the primary supervised learning algorithm, despite its higher memory requirements compared to other algorithms.

RESULTADOS Y DISCUSIÓN

Training of the designed neural network was performed, varying the number of neurons in the hidden layer RedNeru05=5, RedNeru10, RedNeru20. The performance results are shown in Table 2.

Table 2. *Performance results*

Neural Network	Performance
RedNeru05	performance = 52.0823
RedNeru10	performance = 4.0007
RedNeru20	performance = 2.6268

Neural network tuning with 10 neurons in the hidden layer

Figure 5 shows the neural network fitting with 10 neurons in the hidden layer. The performance of the network with different numbers of neurons in the hidden layer indicates that the best performance is provided by the network with the highest number of neurons in the hidden layer, however, if it is observed there is not much difference between the performance of RedNeru10 and RedNeru20, although the processing time of RedNeru20 is much higher than that of RedNeru10; so depending on the degree of accuracy or adjustment that is desired, a sacrifice must be made between accuracy and operation time. There are many practical cases in engineering where RedNeru10 performance is acceptable.

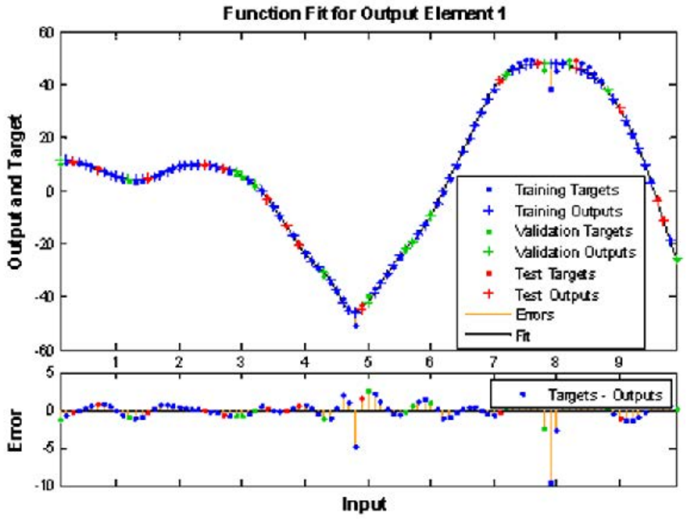

Figure 5. *ANN model adjustment.*

The results obtained in this research highlight the effectiveness of artificial neural networks (ANNs) for parameter identification in systems of ordinary differential equations (ODEs). The multilayer neural network, trained with the backpropagation algorithm and optimized using the Levenberg-Marquardt method, showed superior performance compared to classical numerical methods, such as the Runge-Kutta-based ODE45.

The neural network demonstrated a remarkable ability to approximate the system behavior with high accuracy, even in the presence of nonlinear dynamics and inconsistent experimental data. This aligns with previous studies that have highlighted the ability of ANNs to handle complex and nonlinear problems (Cybenko, 1989; Wang et al., 2020; Zhang et al., 2021).

In comparison, the ODE45 method, although accurate in general, showed limitations at specific intervals due to peaks and discontinuities in the simulated functions. This finding highlights the advantage of ANNs in terms of robustness and adaptability.

Traditional numerical methods, such as ODE45, rely heavily on the accuracy and completeness of the mathematical model. In contrast, ANNs do not require an explicit mathematical model, which allows them to better adapt to variations and perturbations in the data (Dennis & Schnabel, 1983; Yan et al., 2021).

The ability of ANNs to recognize complex patterns with tolerable error margins consolidates them as an effective tool for dynamic systems, overcoming the limitations of numerical methods in situations of high nonlinearity and discontinuities.

The implementation of ANN for parameter identification in dynamical systems has significant implications in various areas of engineering and applied sciences. For example, in industrial processes where conditions may vary and traditional mathematical models may not be sufficient, ANNs offer a flexible and scalable solution (Gonzáles-Medina & Vázquez, 2015; Lee et al., 2023).

In the context of mathematical biology and epidemiology, where systems are inherently nonlinear and complex, ANNs can provide more accurate and adaptive models for prediction and control of biological and epidemiological phenomena (Guo et al., 2022; Rumelhart et al., 1986)

This study contributes to the existing body of knowledge by empirically demonstrating the superiority of ANNs over classical numerical methods in the identification of EDO parameters. This reinforces the position of ANNs as a robust and efficient methodology for modeling nonlinear dynamical systems.

Furthermore, the results obtained support the hypothesis that ANNs, when properly trained and optimized, can overcome the limitations of traditional numerical methods, providing more accurate and less computationally complex solutions.

CONCLUSIONES

The study concluded that artificial neural networks trained using supervised learning techniques, particularly the Backpropagation algorithm, provide a robust and efficient approach for simulating and predicting the behavior of dynamic systems. The Levenberg-Marquardt optimization method was effective in refining the network parameters.

The network configuration with 10 neurons in the hidden layer emerged as the most suitable for achieving a high degree of accuracy with minimal complexity. While numerical methods depend on precise and complete model formulations, neural networks demonstrated superior adaptability and robustness, making them a reliable tool for addressing real-world dynamic modeling challenges. Their scalability and ability to manage errors reinforce their potential for broader application.

REFERENCIAS

Binu, D., & Rajakumar, B. R. (2021). *Artificial Intelligence in Data Mining*. Elsevier Inc. https://doi.org/10.1016/C2019-0-01255-1

Chintha, H. P., & Chatterjee, A. (2022). Identification and parameter estimation of non-polynomial forms of damping nonlinearity in dynamic systems. *International Journal of Non-Linear Mechanics*, *143*. https://doi.org/10.1016/j.ijnonlinmec.2022.104017

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Math. Control Signal Systems*, *2*, 303-314. https://doi.org/10.1007/BF02551274

De-kui, L., & Xing-min, W. (2023). Dislocated function projective partial synchronization between dynamical systems. *Alexandria Engineering Journal*, *66*, 919-926. https://doi.org/10.1016/j.aej.2022.10.065

Dennis, J. E., & Schnabel, R. B. (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall International.

Diamessis, J. E. (1965). A new method for determining the parameters of physical systems. *Proceedings of the IEEE*, *53*(2), 205-206. https://doi.org/10.1109/PROC.1965.3642

Erazo Yánez, C. P., & Navarrete Cedillo, G. A. (2023). Detección de pistas aéreas ilegales en imágenes digitales empleando técnicas de inteligencia artificial. *Ciencia y Poder Aéreo*, *18*(1), 47-56. https://doi.org/10.18667/cienciaypoderaereo.758

Figueredo Ávila, G., & Ballesteros Ricauter, J. (2016). Identificación del estado de madurez de las frutas con redes neuronales artificiales, una revisión. *Ciencia y Agricultura*, *13*(1), 117-132. http://dx.doi.org/10.19053/01228420.4811

Gonzáles-Medina, L. E., & Vázquez, R. A. (2015). Clasificación de Malware mediante Redes Neuronales Artificiales. *Revista del Centro de Investigación*, *11*(44), 69-102. https://www.redalyc.org/articulo.oa?id=34242142004

Guo, Y., O'Brien, J., Sivakumar, M., & Jiang, G. (2022). Back-estimation of norovirus infections through wastewater-based epidemiology: A systematic review and parameter sensitivity. *Water Research*, *219*. https://doi.org/10.1016/j.watres.2022.118610

Hari Rao, V. S., & Yadaiah, N. (2005). Parameter identification of dynamical systems. *Caos, solitones y fractales*, *23*(4), 1137-1151. https://doi.org/10.1016/j.chaos.2003.09.047

Jovanović, A., Stevanović, A., Dobrota, N., & Teodorović, D. (2022). Ecology based network traffic control: A bee colony optimization approach. *Engineering Applications of Artificial Intelligence*, *115*. https://doi.org/10.1016/j.engappai.2022.105262

Lastre Aleaga, A., Méndez Garcés, E., & Cordovés García, A. (2015). Sistema automatizado para la predicción de flujo de carga en subestaciones eléctricas mediante redes neuronales artificiales. *Enfoque UTE*, *6*(3), 20-35. https://www.redalyc.org/articulo.oa?id=572260848002

Lee, J., De Brouwer, E., Hamzi, B., & Owhadi, H. (2023). Learning dynamical systems from data: A simple cross-validation perspective, Part III: Irregularly-sampled time series. *Physica D: Nonlinear Phenomena*, *443*(133546). https://doi.org/10.1016/j.physd.2022.133546

Margaglio, E., & Uria, M. (1994). *Identification of Multivariable Industrial Processes Using Neural Networks: An Application*. IEEE.

Meidani, K., & Barati Farimani, A. (2023). Identification of parametric dynamical systems using integer programming. *Expert Systems with Applications*, *219*, 119622. https://doi.org/10.1016/j.eswa.2023.119622

Modest, M. F. (2003). *Radiative Heat Transfer*. Elsevier Inc. https://doi.org/10.1016/B978-0-12-503163-9.X5000-0

Raju P, E., & Terzija, V. (2023). Power system loads modeling. *Encyclopedia of Electrical and Electronic Power Engineering*, 186-208. https://doi.org/10.1016/B978-0-12-821204-2.00140-9

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533-536. https://doi.org/10.1038/323533a0

Sage, A. P., & Melsa, J. L. (1971). System Identification (First Edition ed.). Academic Press: New York.

Saldarriaga, J. F. (2022). Application of an artificial neural networks for predicting the heat transfer in conical spouted bed using the Nusselt module. *Heliyon*, *8*(11), e11611. https://doi.org/10.1016/j.heliyon.2022.e11611

Sancho, F. (2017). *Redes Neuronales: una visión superficial*. Universidad de Sevilla. http://www.cs.us.es/~fsancho/?e=72

Sarmiento-Ramos, J. L. (2020). Aplicaciones de las redes neuronales y el deep learning a la ingeniería biomédica. *Revista UIS Ingenierías*, *19*(4), 1-18. https://doi.org/10.18273/revuin.v19n4-2020001

Tarigan, J., Nadia, Diedan, R., & Suryana, Y. (2017). Plate Recognition Using Backpropagation Neural Network and Genetic Algorithm. *Procedia Computer Science*, *116*, 365-372. https://doi.org/10.1016/j.procs.2017.10.068

van Ophem, S., & Desmet, W. (2022). Physics-based sound radiation estimation from multiple speakers by combined lumped parameter and reduced-order finite element modeling. *Mechanical Systems and Signal Processing*, *167*(Part B). https://doi.org/10.1016/j.ymssp.2021.108585

Wang, Y., Liu, J., Li, R., Suo, X., & Lu, E. (2020). Precipitation forecast of the Wujiang River Basin based on artificial bee colony algorithm and backpropagation neural network. *Alexandria Engineering Journal*, *59*(3), 1473-1483. https://doi.org/10.1016/j.aej.2020.04.035

Yan, J., Laflamme, S., Hong, J., & Dodson, J. (2021). Online parameter estimation under non-persistent excitations for high-rate dynamic systems. *Mechanical Systems and Signal Processing*, 161. https://doi.org/10.1016/j.ymssp.2021.107960

Zhang, Y., Zhu, X., & Gao, J. (2021). Hidden physics model for parameter estimation of elastic wave equations. *Computer Methods in Applied Mechanics and Engineering*, 381. https://doi.org/10.1016/j.cma.2021.113814

Conflictos de interés:

Los autores declaran no tener conflictos de interés.

Contribución de los autores:

Francisco Javier Duque-Aldaz, Fernando Raúl Rodríguez-Flores y José Carmona Tapia: Conceptualización, curación de datos, análisis formal, investigación, metodología, supervisión, validación, visualización, redacción del borrador original y redacción, revisión y edición.

Descargo de responsabilidad/Nota del editor:

Las declaraciones, opiniones y datos contenidos en todas las publicaciones son únicamente de los autores y contribuyentes individuales y no de Revista San Gregorio ni de los editores. Revista San Gregorio y/o los editores renuncian a toda responsabilidad por cualquier daño a personas o propiedades resultantes de cualquier idea, método, instrucción o producto mencionado en el contenido.