ISSN: 1392-6292/ eISSN: 1648-3510

VILNIUS TECH Vilnius Gediminas Technical University

MATHEMATICAL MODELLING and ANALYSIS

2025
Volume 30
Issue 2
Pages 322–341
https://doi.org/10.3846/mma.2025.22089

Optimizing chaotic systems by orbit counting and Fourier spectrum: FPGA implementation and image encryption application

Ricardo Rojas-Galván[a], Emanuel Mercado-Gutiérrez[b] and Esteban Tlelo-Cuautle[c] ✉

[a] *Faculty of Electronics and Communications Engineering, Universidad Veracruzana, 93390 Poza Rica, Mexico*

[b] *Department of Electro-Photonic Engineering, Centro Universitario de Ciencias Exactas e Ingenierias - UdeG, 44430 Jalisco, Mexico*

[c] *Electronics Department, Instituto Nacional de Astrofísica, Optica y Electrónica (INAOE), Luis Enrique Erro No. 1, Tonantzintla, 72840 Puebla, Mexico*

Article History:

- received August 13, 2024
- revised November 15, 2024
- accepted January 7, 2025

Abstract. The optimization of chaotic systems has been performed by considering dynamical characteristics of the mathematical models. The proposed work shows the application of genetic algorithms (GAs) to optimize the chaotic behavior of three well-known systems, namely: Lorenz, Chen and Lü. The parameters of the chaotic systems are varied in a specific range of values considered as the search space, and the evaluation of the mathematical model is performed by applying the Forward Euler method. The contribution presented herein is that the chaotic behavior is evaluated by counting the orbits in an attractor and the sparsity of them. In addition, the chaotic behavior is guaranteed by evaluating the Fourier spectrum of the time series. The solutions provided by the GA, are then implemented on a field-programmable gate array (FPGA) to verify the experimental generation of chaotic attractors. Finally, two optimized chaotic systems are synchronized and used to encrypt an image, thus confirming the appropriateness of optimizing the chaotic behavior by orbit counting and Fourier spectrum analysis.

Keywords: chaotic system; genetic algorithm; orbit; Fourier spectrum; FPGA; image encryption.

AMS Subject Classification: 34-04; 34Hxx; 49-XX.

✉ Corresponding author. E-mail: etlelo@inaoep.mx

1 Introduction

Chaotic systems have attracted the attention of researchers in different fields. For example, the authors in [16] show a comparative analysis of models of gene and neuronal networks, paying emphasis on the chaotic behavior generated by the ordinary differential equations modeling the systems. Another comparative study that applies differential evolution algorithm is given in [17], to enhance the performance in the offline controller tuning of robotic manipulators with chaos. In [19], a method for the global asymptotic stabilization of an affine control chaotic Lorenz system, via admissible (bounded and regular) feedback controls, is introduced. In [10], the authors introduce a generalization of Bateman equations model mass balance in a linear radioactive decay chain of isotopes, by applying a fractional derivative to include memory effects, and by incorporating randomness in the input parameters (decay rate and initial concentrations), since it is not possible to predict when a particular nuclide will decay from a quantum-mechanical point of view. The authors in [15], discuss the different chaotic phenomena, sensitivity analysis, and bifurcation analysis of the planer dynamical system by considering the Galilean transformation to the Lonngren wave equation (LWE) and the $(2+1)$-dimensional stochastic Nizhnik-Novikov-Veselov system. As one can infer, these topics make use of chaotic models that are also suitable to generate randomness through the design of pseudo-random number generators (PRNGs), as shown in [5,8]. However, to enhance randomness, chaotic systems must be optimized. On this regard, up to now the optimization process has been performed through considering the maximization of dynamical characteristics such as the Lyapunov exponents or Kaplan-Yorke dimension of a chaotic model, as shown in [24].

Chaotic systems have shown advantages when they are used within an optimization algorithm. For example, the authors in [4] propose a hybrid chaos-cloud salp swarm algorithm, where a chaotic map is used to enhance the diversity and to avoid it from falling into local optimum. The authors in [21], incorporate the chaotic Chimp sine cosine optimization algorithm, employing a random update strategy, to optimize hyperparameters on network intrusion detection, and segmented image-based network intrusion detection datasets. An improved chaotic Bat algorithm for optimal coordinated tuning of power system stabilizers for multimachine power system, is introduced in [23]. Similarly, the authors in [11], formulated an energy-efficient clustering mechanism using a chaotic genetic algorithm (GA), and subsequently developed an energy-saving routing system using a bio-inspired grey wolf optimizer algorithm. Chaotic systems have also shown their usefulness in applications including machine learning and deep learning, as shown in [20], where the authors introduced a based ensemble model that combines multi-verse through chaotic atom search optimization for preprocessing, which eliminates unsolicited/recurrent information in a dataset. The algorithms mentioned above, can also be used to optimize chaotic systems, as they can replace the optimization algorithms applied in [24]. In fact, recently one can find a good number of algorithms suitable for the optimization of chaotic systems. See for example the fractional order whale optimization algorithm given in [26], which incorporates the idea of fractional calculus into the mathematical structure of the conventional whale optimization algorithm. Other recently published works on optimizing chaotic systems are given in [2, 22, 25, 30].

The proposed work applies the well-known GA [9], to evaluate the chaotic behavior of three systems, namely: Lorenz [12], Chen [3], and Lü [13] systems. In this manner, the contribution of this article is oriented to evaluate the number of orbits in the attractor, and the incorporation of the Fourier spectrum analysis to rank the chaotic behavior. The solutions provided by the GA are used to implement the three oscillators into a field-programmable gate array (FPGA), as one can find a huge number of applications using FPGA based chaotic systems, see for instance the works in [1, 6, 7, 28, 29, 31]. The rest of the manuscript is organized as follows: Section 2 shows the mathematical models and simulated attractors of Lorenz [12], Chen [3], and Lü [13] systems. Section 3 shows the introduction of two procedures to evaluate the chaotic behavior, namely: Fourier spectrum and counting of orbits. The proposed optimization algorithm based on GA, is detailed in Section 4. A couple of optimal solutions for each chaotic system are implemented on FPGA, as shown in Section 5. Section 6 shows the experimental chaotic attractors observed on an oscilloscope and the use of an optimized Lorenz system in a secure communication system performing the encryption/decryption of a grey-scale image. The conclusions are summarized in Section 7.

2 Lorenz, Chen and Lü oscillators

The chaotic oscillators that are case study in the proposed work are the well-known Lorenz, Chen and Lü systems. These chaotic systems are modeled by three ordinary differential equations (ODEs), thus they are 3D systems. The ODEs modeling Lorenz system are given in Table 1 [12], where the parameters σ, ρ and β, are varied in the proposed work within the GA to find optimal solutions. The original parameter values to generate chaotic behavior are also listed in Table 1, and the attractor is given in Figure 1.

Table 1. Mathematical models, parameter values and equilibrium points for Lorenz, Chen and Lü oscillators.

System	Equations	Parameter	Equilibrium points $EP_{1,2,3}$
Lorenz [12]	$\dot{x} = \sigma(y-x)$ $\dot{y} = x(\rho - z) - y$ $\dot{z} = xy - \beta z$	$\sigma = 10$ $\rho = 28$ $\beta = \frac{8}{3}$	$[0,0,0]$, $\left[\sqrt{b(r-1)}, \sqrt{b(r-1)}, r-1\right]$, $\left[-\sqrt{b(r-1)}, -\sqrt{b(r-1)}, r-1\right]$
Chen [3]	$\dot{x} = a(y-x)$ $\dot{y} = x(c-a) - xz + cy$ $\dot{z} = xy - bz$	$a = 35$ $b = 3$ $c = 28$	$[0,0,0]$, $\left[b(2c-a), b(2c-a), 2c-a\right]$, $\left[-b(2c-a), -b(2c-a), 2c-a\right]$
Lü [13]	$\dot{x} = a(y-x)$ $\dot{y} = -xz + cy$ $\dot{z} = xy - bz$	$a = 36$ $b = 3$ $c = 20$	$EP_1 = [0,0,0]$, $EP_2 = [\sqrt{bc}, \sqrt{bc}, c]$, $EP_3 = [-\sqrt{bc}, -\sqrt{bc}, c]$

The Chen chaotic system is described by the ODEs given in the second row in Table 1, where now the parameters a, b and c, are varied to find optimal solutions. The original parameter values to generate chaotic behavior are also listed in that Table, and the attractor is given in Figure 2.

The Lü chaotic system is described by the ODEs given in the third row in Table 1, where similar to the Chen system, the parameters a, b and c, are varied to find optimal solutions. The parameter values to generate chaotic behavior are listed in Table 1, and the attractor is given in Figure 3.

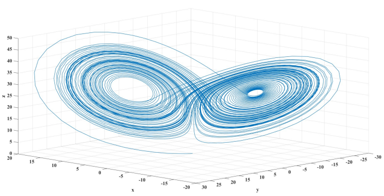


Figure 1. Lorenz attractor.

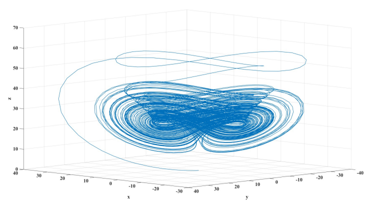


Figure 2. Chen attractor.

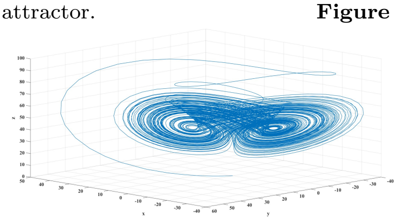


Figure 3. Lü attractor.

3 Chaotic behavior evaluation

As mentioned above, to enhance randomness, chaotic systems can be optimized. However, the optimization process has been performed through considering the maximization of dynamical characteristics such as the Lyapunov exponents or Kaplan-Yorke dimension [24]. These processes can be performed statistically or by manipulating the ODEs. However, as the main goal is to generate an attractor, then this section shows how to evaluate chaotic behavior by computing the Fourier spectrum of a time series and by counting the orbits between two state variables forming a 2D attractor. Both procedures are used as fitness functions within the genetic algorithm (GA) that is detailed in Section 4.

3.1 Frequency spectrum of chaotic time series

The frequency spectrum of a time series is the distribution of amplitudes for each frequency that is found across an undulatory phenomenon, which is a superposition of numerous frequencies. This definition can be used to evaluate the characteristic of the 3D oscillators, because their chaotic time series are a composition of arrays of multiple frequencies on each state variable (x, y, z). To demonstrate the undulatory phenomenon of a chaotic system, Figure 4 shows the three time series of Lorenz system by using the default parameters given in Table 1. The Fourier spectrum of these time series is computed by applying the Fast Fourier Transform (FFT) to produce the plots given in Figure 5.

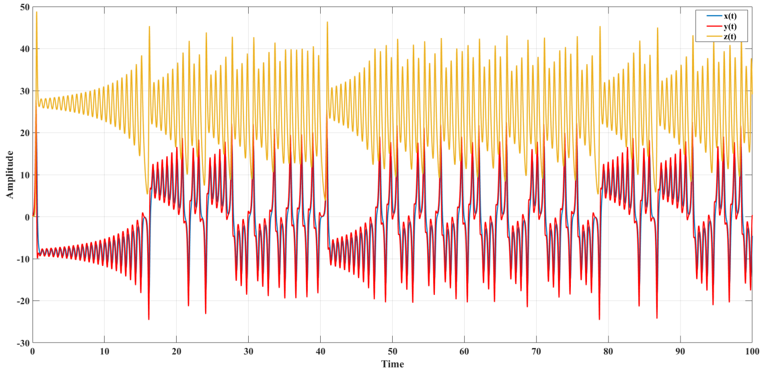


Figure 4. Time series of Lorenz system.

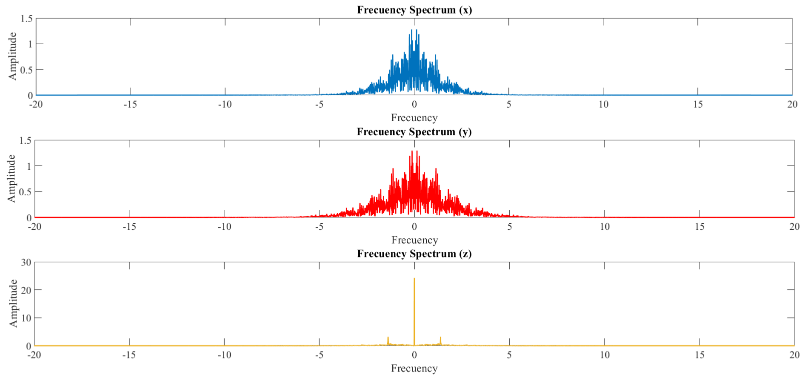


Figure 5. Frequency spectrum of the state variables of Lorenz system.

Looking at the frequency spectrum of the state variables of Lorenz system shown in Figure 5, it is possible to quantify the number of dominant frequencies that each state variable has. This article counts the most dominant pikes in the spectrum and depreciates the ones that are on minor amplitude around a given threshold. Therefore, it is important to mention that the proposed method maintains a certain relation among the amplitudes of the three dominant frequencies or otherwise the form of the attractor could be lost. In addition, it was found out that the number of dominant frequencies must guard a certain proximity amongst them to generate the attractor. For example, taking the frequency spectrum of Lorenz system, it was counted on each state variable x, y, and z, over 979, 1159 and 1107 dominant frequencies, respectively. The amplitudes of the dominant frequencies, are close enough to each other in order to generate the attractor in the butterfly form. In fact, after performing several tests on different chaotic systems, it was found that Lorenz, Chen and Lü attractors have the same relation rule, which can be established as when the distance among the dominant frequencies in a state variable (axis), must be around ±18% the highest axis amplitude.

In summary, in order to adopt the Fourier spectrum as a fitness to evaluate chaotic behavior of time series, it is necessary to count the number of pikes over the frequency spectrum but taking into account the relation among the dominant frequencies. The frequencies that do not have a given percentage relation, can be penalized on their fitness, meaning that the solutions that do not have this desired characteristic will not be taken as an appropriate solution.

3.2 Counting of orbits in 2D portraits

It is well-known that a periodic signal has one orbit in a portrait, so that a chaotic signal must have as many orbits as possible. This idea leads us to introduce a procedure to count orbits and their sparsity in a 2D portrait. In this manner, the way in which the orbits are evaluated, is by considering or drawing a straight line over an attraction region, as sketched in Figure 6, where the line intercepts the attractor. By counting the intersections that the straight line has with the attractor, it provides the number of orbits and the sparsity or difference of length among the orbits.

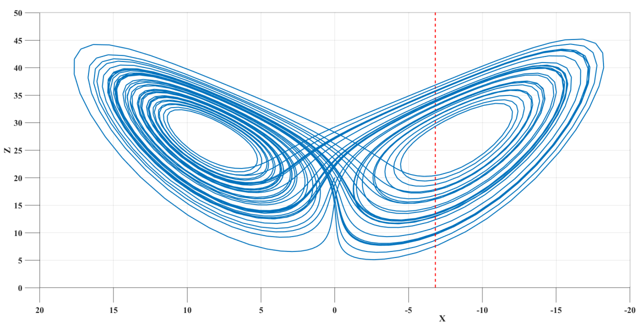


Figure 6. Procedure to evaluate the orbits in a 2D portrait and their sparsity.

For the three case study, as the attractors are similar, the straight line can be placed over one stability point. The points of stability can be calculated by solving the ODEs given in Table 1, when the derivatives are equal to zero. In this manner, this process leads to find the equilibrium points (EP), which are also given in the right side in Table 1.

In the proposed work, the intersection lines are placed over the equilibrium points on the state variable x. For Lorenz system, the line is placed over one of the wings because the number of orbits in their wings also helps to evaluate the sparsity among the orbits. For the Chen and Lü systems, the line is placed in the center of the attractor, because it is important to procure the crown form that makes it distinct from the Lorenz attractor form.

Adding this procedure and the Fourier spectrum as fitness functions to perform the optimization by applying a GA, not only makes the optimization process more robust when finding new solutions, but also it guarantees that the form of the attractor is more defined, as shown in the next sections.

4 Optimization of chaotic systems by Genetic Algorithm

The GA is one of the most used in optimization problems, and it is adapted herein to evaluate chaotic behavior of a mathematical model by performing fitness evaluations of the Fourier spectrum and orbits counting. A GA is based on the reproduction and the evolution of the living beings in which only the more profitable will be surpassing upon the next generation.

As described in [9, 14, 27]. The GA adapted herein performs seven steps for the optimization of 3D chaotic systems from their mathematical models. The first step consists on the creation of an initial population, which number of individuals is varied until the algorithm shows better convergence. The second step evaluates the individuals according to a defined fitness function, which in this case it is associated to the Fourier spectrum and orbits counting, as described in Section 3. Once the individuals are evaluated, they can be classified to be selected according to some criteria. Afterwards, the individuals are evolved by performing genetic operations, such as crossover between pairs of individuals. In the fifth step, a certain percentage is considered to mutate the individuals. The best individuals are saved to evolve in the next generation, and finally, the seventh step considers the selection, which in this case this process is performed by elitism.

As the adapted GA to optimize chaotic systems works with bits encoding the population, then the crossover and mutation operations affect the composition of the individuals by changing their bits. The method of crossing the different individuals used in this GA is called Niching using Adaptive Mating (NAM), which cross the individuals in a crossed way, meaning that the best fit solutions will cross with the worst fit solutions. The NAM method guarantees that the search can be more extensive because it will allow to try different combinations and avoid settling on just one kind of solution. The crossover operation is then performed as follows:

1. Sort of the individuals in descending order.

$$\text{Order the population } P \text{ as: } \text{fitness}(I_1) \geq \text{fitness}(I_2) \geq \cdots \geq \text{fitness}(I_n)$$

2. Matching of the individuals:

$$\begin{aligned} \text{Individual}_1 = I_1 \quad &\text{with} \quad \text{Individual}_n = I_n, \\ \text{Individual}_2 = I_2 \quad &\text{with} \quad \text{Individual}_{n-1} = I_{n-1}, \\ &\vdots \\ \text{Individual}_{n/2} = I_{n/2} \quad &\text{with} \quad \text{Individual}_{n/2+1} = I_{n/2+1}. \end{aligned}$$

It is important to mention that in the GA, not every pair of the individuals will cross because there is a certain percentage or probability of crossing. This step provides a more extensive search of solutions, thus avoiding the algorithm to stay on a single type of solutions.

The mutation operation performed in the proposed work is called *bit flip mutation*. This method operates randomly by selecting the chromosome to mutate and then flipping their bits from 0 to 1, or 1 to 0. This method also introduces randomness in the optimization algorithm and allows the diversity in the creation of generations. This mutation operation is developed as follows,

1. From the current population of individuals from P, select randomly the individuals $I[j]$ to be mutated.

2. Flip the bits from the selected individuals:

$$\text{If } I[j] = 0, \quad I[j] \leftarrow 1,$$

$$\text{If } I[j] = 1, \quad I[j] \leftarrow 0.$$

The mutations in the evolution of the chromosomes is crucial due to the perturbations over the genetic material of the individuals, and this helps to make the exploration in a more efficient way. The mutation process guarantees more probabilities on providing optimal or near-optimal solutions that with the current generation would not be possible.

For the selection of the best solutions, the GA creates an initial population P_n and then by crossing and mutating the individuals, a new and evolved population P_{n+1} will emerge. Subsequently, these two generations are compared among each other according to their fitness and at the end, there will be a population combining the best of the two previous populations. Table 2 shows the process on how the solutions are compared.

Table 2. Comparison of the solutions between generation P_n and P_{n+1}.

Generation P_n	Comparison	Generation P_{n+1}	New generation
$I(1)$	$<$	$J(4)$	$J(4)$
$I(4)$	$>$	$J(2)$	$I(4)$
$I(5)$	$>$	$J(3)$	$I(5)$
$I(2)$	$>$	$J(1)$	$I(2)$
$I(3)$	$<$	$J(5)$	$J(5)$

For the selection process, the proposed work uses the elitism method, which is also called *replacement of the worst*. The main objective of the elitism is to preserve the best solutions upon the next generation, so that this method consists in replacing the worst solution of a population with the best one. This method is quite simple but offers a great advantage when creating the solutions. The selection process is given in Table 3, where a current generation is updated by replacement of the worst individual.

Table 3. Selection process by executing the method called replacement of the worst

Current Generation:

Fitness	Individual
4	$J(4)$
4	$I_(4)$
5	$I(5)$
2	$I(2)$
5	$J(5)$

Updated Generation:

Fitness	Individual
4	$J(4)$
4	$I_(4)$
5	$I(5)$
5	$I(5)$
5	$J(5)$

The optimization of chaotic systems by applying the GA is then given in the pseudocode detailed in Algorithm 1. It includes the parameters for the simulation of the chaotic systems by applying a numerical method, which includes a step size h that is required to perform the implementation of the best solutions into a field-programmable gate array (FPGA), as shown in Section 5.

The GA for the optimization of Lorenz, Chen and Lü systems, was executed during 50 generations, with 30 individuals in the population, a crossover probability of 60%, and a mutation probability of 30%. Each individual was encoded by using 12 bits and the search spaces of the design parameters were set to: $0 < \sigma \leq 70, 0 < \rho \leq 65$, and $0 < \beta \leq 25$ for Lorenz system; $0 < a \leq 60, 0 < b \leq 12$, and $0 < c \leq 40$ for Chen system; and $0 < a \leq 70, 0 < b \leq 20$, and $0 < c \leq 40$ for Lü system. The initial conditions were established as: $(x_0, y_0, z_0) = (0.1, 0.1, 0.1)$, $h = 0.005$, and time simulation = 100,000 iterations for Lorenz system; $(x_0, y_0, z_0) = (0.1, 0.1, 0.1)$, $h = 0.004$, and 100,000 iterations for Chen system; and $(x_0, y_0, z_0) = (0.1, 0.1, 0.1)$, $h = 0.005$, and 100,000 iterations for Lü system. The parameters of the best ten solutions for each chaotic system are given in Table 4. A couple of these solutions are used to synthesize the chaotic oscillators into an FPGA, as shown in Section 5.

5 FPGA implementation

The previous section detailed the adaptation of the GA [9], to optimize three well-known chaotic systems, namely: Lorenz [12], Chen [3], and Lü [13]. Table 4 provides optimized parameters to proceed to the FPGA implementation of those solutions. Recall that one can find a huge number of works showing FPGA implementations of different chaotic systems, as shown in [1, 6, 7, 28, 29, 31]. The first step is the discretization of the ODEs by applying a numerical method, as shown in [24]. The second step consists on the block description of the equations and the definition of the computer arithmetic by using 32 bits, which in the proposed work is established in the fixed-point format 1.11.20. The third step is the Verilog description to perform the synthesis into an FPGA.

5.1 Discretization of the ODEs

Given the ODEs modeling the chaotic systems, as the ones given in Table 1 for Lorenz, Chen, and Lü, their approximation can be performed by applying

Algorithm 1 Genetic algorithm to optimize chaotic systems by Fourier transform and orbits counting.

```
Define: Number of generations, Population size, Size of bits for the individuals, Search spaces of the parameters
Define the parameters of the chaotic system: Initial conditions, Interval of time simulation, Step size h of the numerical method
Define the probabilities of the crossover and mutation operations
Generate the initial population of chromosomes in binary representation
for Each generation do
    Keep the current generation of chromosomes as population P_n
    Choose the crossover points of the chromosomes
    Evaluate the fitness of the individuals in the current generation using the Fourier spectrum and orbits counting methods given in Section 3
    Order the chromosomes according to the fitness from highest to lowest
    Use the crossover method of Niching using Adaptive Mating (NAM)
    for each set of individuals in P_n called fathers do
        if a random number is higher than the crossing probability then
            Cross the fathers in order to create new sons
        else
            Maintain the fathers
        end if
    end for
    Apply the mutations to the chromosomes selected according to the given mutation probability
    Evaluate the fitness of the new individuals in population P_{n+1} using the Fourier spectrum and orbits counting methods given in Section 3
    Compare the passed and the new individuals and keep the best solutions for the next generation
    Select individuals by replacement of the worst, as shown in Table 3
end for
Print the best parameters values for the system according to the fitness
```

a numerical method to allow FPGA implementation. In the proposed work, all these equations are discretized by applying Forward Euler method, which iterative equation is given in (5.1),

$$w_{n+1} = w_n + hf(w_n), \tag{5.1}$$

where h is the step size that must be established to guarantee convergence and to increase the operation frequency for the FPGA synthesis, as shown in [24].

Applying (5.1) to the ODEs given in Table 1, one gets the discretized equations given in (5.2) for Lorenz, (5.3) for Chen, and (5.4) for Lü systems.

$$\begin{aligned} x_{n+1} &= x_n + h(\sigma(y_n - x_n)) \\ y_{n+1} &= y_n + h(x_n(\rho - z_n) - y_n) \\ z_{n+1} &= z_n + h(x_n y_n - \beta z_n) \end{aligned} \tag{5.2}$$

Table 4. Best ten solutions provided by the GA for Lorenz, Chen and Lü systems.

Lorenz	Chen
$\sigma = 12.04$, $\rho = 29.44$, $\beta = 4.49$	$a = 36.8$, $b = 5.08$, $c = 27.15$
$\sigma = 12.32$, $\rho = 29.09$, $\beta = 3.18$	$a = 40.81$, $b = 6.48$, $c = 29.25$
$\sigma = 34.57$, $\rho = 54.89$, $\beta = 12.42$	$a = 56.6$, $b = 7.79$, $c = 37.44$
$\sigma = 31.09$, $\rho = 52.85$, $\beta = 11.32$	$a = 60.0$, $b = 10.32$, $c = 41.0$
$\sigma = 23.85$, $\rho = 51.94$, $\beta = 7.77$	$a = 55.43$, $b = 7.82$, $c = 39.26$
$\sigma = 39.57$, $\rho = 54.03$, $\beta = 12.56$	$a = 54.84$, $b = 6.07$, $c = 38.67$
$\sigma = 17.52$, $\rho = 43.17$, $\beta = 4.96$	$a = 44.34$, $b = 4.2$, $c = 32.3$
$\sigma = 38.94$, $\rho = 54.95$, $\beta = 13.10$	$a = 54.6$, $b = 7.07$, $c = 38.08$
$\sigma = 35.00$, $\rho = 41.99$, $\beta = 6.30$	$a = 55.48$, $b = 5.1$, $c = 37.58$
$\sigma = 67.85$, $\rho = 64.38$, $\beta = 20.03$	$a = 29.97$, $b = 2.96$, $c = 21.77$
Lü	
$a = 37.46$, $b = 6.52$, $c = 23.17$	
$a = 39.96$, $b = 8.7$, $c = 24.4$	
$a = 58.89$, $b = 9.18$, $c = 33.59$	
$a = 55.6$, $b = 9.47$, $c = 33.74$	
$a = 51.20$, $b = 9.27$, $c = 29.68$	
$a = 59.35$, $b = 15.31$, $c = 34.19$	
$a = 50.67$, $b = 12.08$, $c = 29.79$	
$a = 61.95$, $b = 14.10$, $c = 36.74$	
$a = 66.85$, $b = 8.28$, $c = 38.01$	
$a = 69.04$, $b = 8.24$, $c = 38.06$	

$$\begin{aligned} x_{n+1} &= x_n + h(a(y_n - x_n)) \\ y_{n+1} &= y_n + h((c-a)x_n - x_n z_n + c y_n) \\ z_{n+1} &= z_n + h(x_n y_n - b z_n) \end{aligned} \tag{5.3}$$

$$\begin{aligned} x_{n+1} &= x_n + h(a(y_n - x_n)) \\ y_{n+1} &= y_n + h(-x_n z_n + c y_n) \\ z_{n+1} &= z_n + h(x_n y_n - b z_n) \end{aligned} \tag{5.4}$$

5.2 Block description of the discretized equations

The FPGA synthesis of discretized equations requires their block description to identify arithmetic and logic operations. For example, considering Lorenz equations given in (5.2), one can identify arithmetic operations like addition, subtraction and multiplication. These operations can be performed by designing digital blocks that can include two inputs whose bits length depend on the computer arithmetic. In addition, these blocks can include pines for the connection of the clock (CLK) pulse and reset (RST). In this manner, Figure 7 shows the block description of (5.2), where the output of each block indicates the operation that is performed. The blocks labeled Integrator FE block processes the step size h and adds the state variable in the current iteration n. The outputs are then the state variables at iteration $n + 1$, whose values are stored in registers that include an enable (ENB) pin.

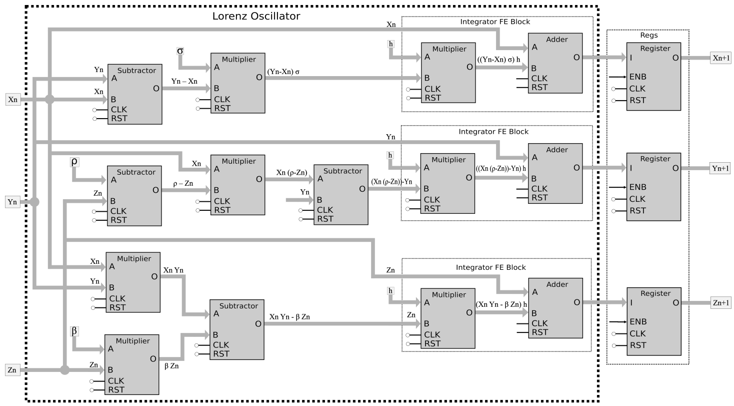


Figure 7. Block description of Lorenz system from (5.2).

The block description shown in Figure 7, requires the design of a finite state machine (FSM) to control the iterations between n and $n + 1$, because each block has a delay and therefore, each design needs a determined number of clock cycles (CCs) to process the data from iteration n to $n+1$. Basically, the FSM is designed to control the iterative process that is described in a high-level, as shown in Figure 8. In this block description one can see that the CLK is connected to both blocks, namely: Lorenz oscillator and Regs. However, as the block Lorenz oscillator requires a determined number of CCs, the data stored in Regs is only available when pin enable is activated. These registers are also used to store the initial conditions.

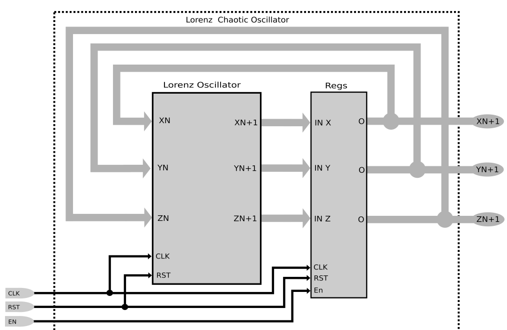


Figure 8. Macro-block description of Lorenz system.

The block descriptions of Chen and Lü systems, are performed in a similar manner, beginning from their discretized equations given in (5.3) and (5.4), respectively. All these blocks are also designed considering a computer arithmetic with 32 bits that are distributed in fixed-point format, thus using 1.11.20 notation for Lorenz, Chen, and Lü systems. It is important to mention that the number of bits that represent the integer part in each case, was established according to the maximum number that arises among all operations in the discretized equations, where the maximum values may arise between the multiplication of two state variables. One can find more details in [24].

5.3 Verilog descriptions of the 3D chaotic systems

The block description that was developed from the discretized equations of the ODEs modeling a chaotic system, is now described under Verilog to proceed to the FPGA synthesis. This section shows two descriptions: Top module for the macro-block description shown in Figure 8, and another one for the detailed block description, as shown in Figure 7, for Lorenz system.

The Top module description of Lorenz system is given in Listing 1, which defines the computer arithmetic with 32 bit, but it can be changed as this is taken as parameter value n. One can see the definition of the clock, reset and enable pines, and the definition of six buses of 32 bits described as wire signed.

Listing 1: Verilog description as Top module for Lorenz system

```
module Top_Module_Lorenz (clk,rst,X,Y,Z,en);
  parameter n = 32;
  input clk,rst, en;
  output [n-1:0] X,Y,Z;
  wire signed [n-1:0] o1,o2,o3,o4,o5,o6;
  processlo U1 (clk, rst, X, Y, Z, o1, o2, o3);
  integration U2 (clk,rst,o1,o2,o3, X,Y,Z, o4,o5,o6);
  REGISTERS U3 (clk, rst, en, o4, o5, o6, X, Y, Z);
endmodule
```

The Top module of Chen system is given in Listing 2, which includes the initial conditions $(x_0, y_0, z_0) = (4.246, 4.728, 13.470)$ as parameter values. However, in the Verilog code one can see that these values are given in fixed-point format using 1 bit to represent the sign, 11 bits to represent the integer part, and 20 bits to represent the decimal part of the given float number.

Listing 2: Verilog description as Top module for Chen system

```
module Top_Module_Chen (RST,CLK,ENB,XN1,YN1,ZN1);
  parameter n=32;
  parameter XN0 = 32'h43EF9E;
  parameter YN0 = 32'h4BA5E3;
  parameter ZN0 = 32'hD7851F;
   input RST,CLK,ENB;
   output [n-1:0] XN1,YN1,ZN1;
   wire [n-1:0] RX,RY,RZ;
   wire OPRF,OPR;
   assign OPRF = OPR & ENB;
processlo V1 (RST,CLK,XN1,YN1,ZN1,RX,RY,RZ);
```

```
REGS_OUT_3 #(XN0,YN0,ZN0)V2(RST,CLK,OPRF,RX,RY,RZ,
        XN1,YN1,ZN1);
COUNTER V3 (RST,CLK,OPR);
endmodule
```

The Top module description of Lü system is similar to the one for Lorenz system. As one can see, these Top modules instantiate a module called processlo. This module is refined in a low level of abstraction, as shown in Listing 3 for Lorenz system, which defines the coefficients of the mathematical model given in Table 1. In this manner, one can change the values of the coefficients by any of the optimized ones given in Table 4, but recall that those values must be represented in fixed-point notation.

Listing 3: Module processlo for Lorenz system

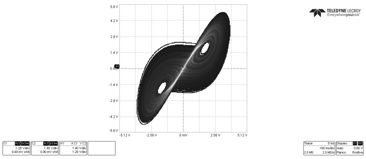
```
module processlo (CLK,RST,Xi,Yi,Zi,xo,yo,zo);
parameter n = 32;
input CLK,RST;
input signed [n-1:0] Xi,Yi,Zi;
output [n-1:0] xo,yo,zo;
wire signed [n-1:0] o1,o2,o3,o4,o5;
//s=sigma, r=rho, b=beta
wire [n-1:0] s;
wire [n-1:0] r;
wire [n-1:0] b;
assign s = 32'h00C51EB8;
assign r = 32'h01D170A4;
assign b = 32'h0032E148;
//...
endmodule
```

6 Experimental results and image encryption

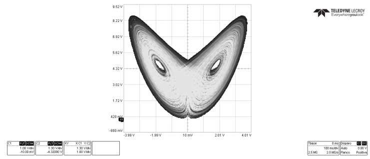
This section shows the FPGA synthesis of the best solutions given in Table 4, for each chaotic system to observe experimental attractors. Two optimized systems are synchronized to develop a secure image transmission system.

6.1 Observation of experimental attractors

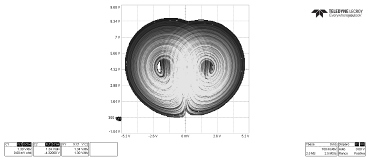
The observation of experimental attractors from the FPGA synthesis, is performed by reprogramming the coefficient values of the chaotic systems by using the optimal solutions given in Table 4. Therefore, Figure 9 shows the FPGA experimental results using the best optimal values in the first row in Table 4 for Lorenz system. The synthesis was performed by using the FPGA Cyclone II EP4CGX150DF31C7. A Digital-to-Analog (DAC) converter of 16 bits is connected to the FPGA to observe the attractors on a Teledyne Lecroy oscilloscope.



(a) Attractor in the plane X-Y.



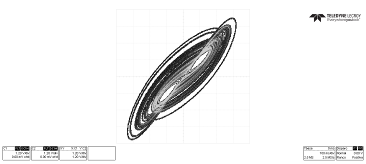
(b) Attractor in the plane X-Z.



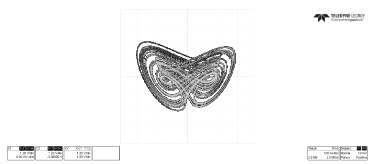
(c) Attractor in the plane Y-Z.

Figure 9. FPGA results for Lorenz system using: σ=12.04, ρ=29.44, β=4.49.

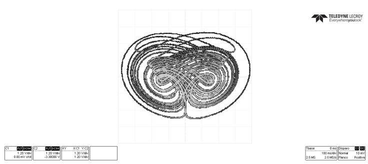
Figure 10, shows the FPGA experimental results using the best optimal values in the first row in Table 4 for Chen system.



(a) Attractor in the plane X-Y.



(b) Attractor in the plane X-Z.



(c) Attractor in the plane Y-Z.

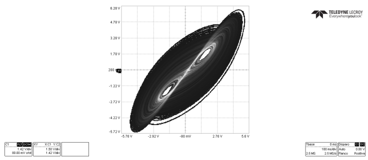
Figure 10. FPGA results for Chen system using: a=36.8, b=5.08, c=27.15.

Figure 11, shows the FPGA experimental results using the best optimal values in the first row in Table 4 for Lü system.

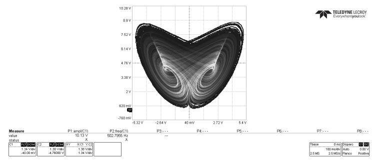
The hardware resources that were required for the FPGA implementation of two optimal cases for each chaotic system, are given in Table 5. It can be appreciated that Lorenz and Chen systems require similar hardware resources and work with similar frequencies, around 100 MHz and 76 MHz, respectively. The highest operating frequency is reached by Lü system with 321.54 MHz. this chaotic system also requires lower number of logic elements and registers.

6.2 Image encryption by synchronizing two chaotic systems

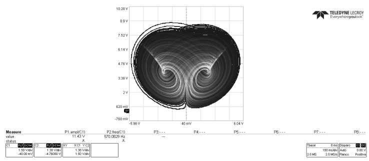
This section shows the development of a secure communication system. On this regard, the seminal work introduced by Pecora-Carroll [18], showed the



(a) Attractor in the plane X-Y.



(b) Attractor in the plane X-Z.



(c) Attractor in the plane Y-Z.

Figure 11. FPGA results for Lü system using: a=37.46, b=6.52, c=23.17.

Table 5. Hardware resources consumption for the experimental attractors using FPGA Cyclone II EP4CGX150DF31C7.

System	σ—a value	ρ—b value	β—c value	Logic elements	Registers	Embedded Multiplier 9 bit	Frequency (MHz)
Lorenz	12.04	29.44	4.49	805	415	44	101.84
Lorenz	12.32	29.09	3.18	810	415	44	100.54
Lü	37.46	6.52	23.17	49	22	0	321.54
Lü	39.96	8.7	24.4	49	22	0	321.54
Chen	36.8	5.08	27.15	819	344	52	76.99
Chen	40.81	6.48	29.25	819	344	52	73.6

synchronization of two identical dynamical systems. In this manner, the proposed work uses two optimized Lorenz to be synchronized from the equations given in Table 1. The application of the Pecora-Carroll method leads to define one system to be the master or transmission system and a second system is adapted to become the slave or receiver system, as described in (6.1).

$$\begin{aligned}
Xm_{n+1} &= Xm_n + h(\sigma(Ym_n - Xm_n)),\\
Ym_{n+1} &= Ym_n + h(Xm_n(\rho - Zm_n) - Ym_n),\\
Zm_{n+1} &= Zm_n + h(Xm_nYm_n - \beta Zm_n),\\
Ys_{n+1} &= Ys_n + h(Xm_n(\rho - Zs_n) - Ys_n),\\
Zs_{n+1} &= Zs_n + h(Xm_nYs_n - \beta Zs_n).
\end{aligned} \tag{6.1}$$

The slave system is then modeled by only two equations of the chaotic system, while the third equation is taken as it is from the master system. In this manner, the state variables of the slave system Ys and Zs, synchronize with the state variables of the master system Xm, Ym and Zm. The synchronized signals can be used to transmit an image that can be encrypted by performing XOR operations as shown in Figure 12. One can see that an original image of n bits represented as A[n], is processed through an XOR operation with the n bits generated by the state variable Ym from the master chaotic system, which data is stored in the string B[n]. Thus the channel transmits encrypted data, which is recovered in the slave or receiver system by performing the XOR operation now between B[n] and the n bits generated by the state variable Ys from the slave chaotic system, which data is stored in C[n].

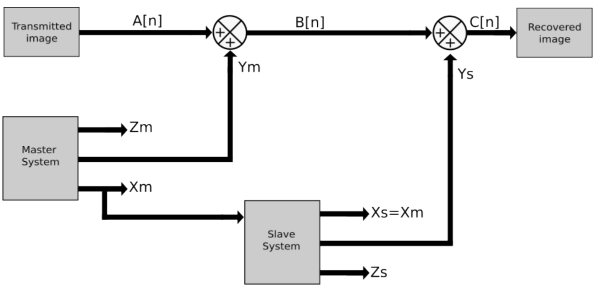
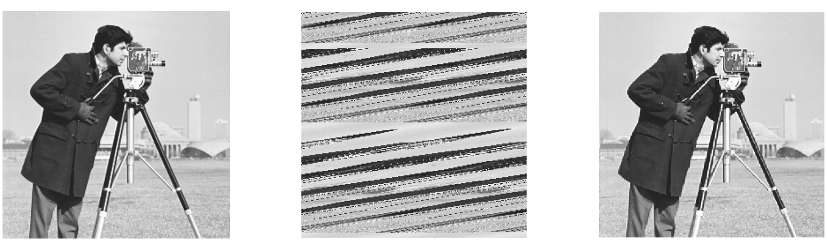


Figure 12. Block description of an image encryption/decryption by synchronizing two chaotic systems applying the Pecora-Carrol method.

It is important to mention that the synchronization process requires a transition time to accomplish zero synchronization error, so that the bits generated in the same state variables in the master and slave systems, have the same logic values. In the proposed work, the first 500 iterations of the chaotic systems were discarded, and afterwards, each iteration generates 8 bits in the state variable. These 8 bits are then processed on the XOR operation with each pixel of the image that in this case it is a grey scale image having 256×256 pixels. The experimental encryption/decryption results are shown in Figure 13.



(a) Original image. (b) Encrypted data. (c) Recovered image.

Figure 13. Experimental result for the encryption and decryption of a grey scale image, using the chaos-based secure communication system shown in Figure 12.

7 Conclusions

A genetic algorithm (GA) has been adapted to optimize three chaotic systems, namely: Lorenz, Chen and Lü systems. The fitness function was devoted to analyze the Fourier spectrum of the time series, and to count the orbits in a portrait between two state variables. Both procedures were detailed to guarantee the successfulness of finding chaotic attractors having sparse orbits through plotting a line around the equilibrium points of the attraction region. The best optimal coefficient values provided by the GA, were taken for each chaotic system to develop their FPGA implementation. The computer arithmetic was established to use fixed-point notation with 1.11.20 format. The experimental attractors were obtained by using the FPGA Cyclone II EP4CGX150DF31C7, and the hardware resources consumptions were given in Table 5. Finally, the optimized Lorenz system was synchronized in a master-slave topology to develop a secure image transmission system, for which a grey scale image was processed. The image encryption/decryption application confirms the suitability on performing the optimization of chaotic systems by evaluating the Fourier spectrum and orbits counting of the attractors.

Acknowledgements

Ricardo Rojas-Galván and Emanuel Mercado-Gutiérrez want to thank to the scientific Delfin's summer program developed during June-July of 2024. Esteban Tlelo-Cuautle is on Sabbatical Leave at CINVESTAV supported by CONAHCyT-Mexico: Apoyos Complementarios para Estancias Sabaticas Vinculadas a la Consolidacion de Grupos de Investigacion, in 2023.

References

[1] M.S. Azzaz, R. Kaibou and B. Madani. Co-design based FPGA implementation of an efficient new speech hyperchaotic cryptosystem in the transform domain. *Integration, the VLSI Journal*, **97**, Jul. 2024. https://doi.org/10.1016/j.vlsi.2024.102197.

[2] H. Bao, P. Ren, K. Xu, L. Yang, H. Zhou, J. Li, Y. Li and X. Miao. Energy efficient memristive transiently chaotic neural network for combinatorial optimization. *IEEE TCAS-I*, **71**(8):3708–3716, Aug. 2024. https://doi.org/10.1109/TCSI.2024.3406167.

[3] G. Chen and T. Ueta. Yet another chaotic attractor. *IJBC*, **9**(7):1465–1466, 1999. https://doi.org/10.1142/S0218127499001024.

[4] J. Dai and L.-H. Fu. Research on the hybrid chaos-coud salp swarm algorithm. *Commun. Nonlinear Sci. Numer. Simul.*, **138**, Nov. 2024.

[5] L.G. de la Fraga and B. Ovilla-Martinez. Generating pseudo-random numbers with a brownian system. *Integration, the VLSI Journal*, **96**, May. 2024. https://doi.org/10.1016/j.vlsi.2023.102135.

[6] M. Devipriya and M. Brindha. Reconfigurable architecture for DNA diffusion technique-based medical image encryption. *JCSC*, **32**(4), Mar. 15 2023. https://doi.org/10.1142/S0218126623500652.

[7] M. Devipriya, M. Sreenivasan and M. Brindha. Reconfigurable architecture for image encryption using a three-layer artificial neural network. *IETE Journal of Research*, **70**(1):473–486, Jan. 2 2024. https://doi.org/10.1080/03772063.2022.2127940.

[8] M. Gafsi, R. Amdouni, M.A. Hajjaji, A. Mtibaa and E.-B. Bourennane. Hardware implementation of a strong pseudorandom number generator based block-cipher system for color image encryption and decryption. *Int. J. Circuit Theory Appl.*, **51**(1):410–436, Jan. 2023. https://doi.org/10.1002/cta.3415.

[9] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[10] M. Jornet. Generalized polynomial chaos expansions for the random fractional bateman equations. *Appl. Math. Comput.*, **479**, Oct. 15 2024. https://doi.org/10.1016/j.amc.2024.128873.

[11] H. Khujamatov, M. Pitchai, A. Shamsiev, A. Mukhamadiyev and J. Cho. Clustered routing using chaotic genetic algorithm with grey wolf optimization to enhance energy efficiency in sensor networks. *Sensors*, **24**(13), Jul. 2024. https://doi.org/10.3390/s24134406.

[12] E.N. Lorenz. Deterministic nonperiodic flow. *J. Atmos. Sci.*, **20**:130–141, 1963. https://doi.org/10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2.

[13] J. Lü and G. Chen. A new chaotic attractor coined. *IJBC*, **12**(03):659–661, 2002. https://doi.org/10.1142/S0218127402004620.

[14] R. Malhotra, N. Singh and Y. Singh. Genetic algorithms: Concepts, design for optimization of process controllers. *Computer and Information Science*, **4**(2):39–54, 2011. https://doi.org/10.5539/cis.v4n2p39.

[15] M.M. Miah, F. Alsharif, Md.A. Iqbal, J.R.M. Borhan and M. Kanan. Chaotic phenomena, sensitivity analysis, bifurcation analysis, and new abundant solitary wave structures of the two nonlinear dynamical models in industrial optimization. *Math.*, **12**(13), Jul. 2024. https://doi.org/10.3390/math12131959.

[16] D. Ogorelova and F. Sadyrbaev. Comparative analysis of models of genetic and neuronal networks. *Math. Model. Anal.*, **29**(2):277–287, 2024. https://doi.org/10.3846/mma.2024.19714.

[17] M. F. Parra-Ocampo, O. Serrano-Perez, A. Rodriguez-Molina, M.G. Villarreal-Cervantes, G. Hernandez, M.E. Sanchez-Gutierrez and V.M. Silva-Garcia. Enhancing the performance in the offline controller tuning of robotic manipulators with chaos: a comparative study with differential evolution. *Int. J. Dynam. Control*, 2024 Apr. 11 2024. https://doi.org/10.1007/s40435-024-01423-6.

[18] L.M. Pecora and T.L. Carroll. Synchronization in chaotic systems. *Phys. Rev. Lett.*, **64**(8):821–824, Feb. 19 1990. https://doi.org/10.1103/PhysRevLett.64.821.

[19] H.M. Perez and J. Solis-Daun. Global stabilization of a bounded controlled lorenz system. *IJBC*, **34**(07), Jun. 15 2024. https://doi.org/10.1142/S0218127424500895.

[20] P. Pitchandi, M. Nivaashini and R.K. Grace. Optimized ensemble learning models based on clustering and hybrid deep learning for wireless intrusion detection. *IETE Journal of Research*, 2024 Jun. 19 2024. https://doi.org/10.1080/03772063.2024.2367042.

[21] N.P. Ponnuviji, E. Nirmala, F.M. H. Fernandez and K. Anitha. ALRN-RCS: Advanced approach to network intrusion detection using attention long-term recurrent networks and chaotic optimization. *IETE Journal of Research*, 2024 Jul. 24 2024. https://doi.org/10.1080/03772063.2024.2370949.

[22] X. Qin and Y. Zhang. Image encryption algorithm based on coa and hyperchaotic lorenz system. *Nonlinear Dyn.*, **112**(12):10611–10632, Jun. 2024. https://doi.org/10.1007/s11071-024-09632-6.

[23] M. Tadj, L. Chaib, A. Choucha, M. Alhazmi, A. Alwabli, M. Bajaj and S.A. Dost Mohammadi. Improved chaotic bat algorithm for optimal coordinated tuning of power system stabilizers for multimachine power system. *Sci. Rep.*, **14**(1), Jul. 2 2024. https://doi.org/10.1038/s41598-024-65101-5.

[24] E. Tlelo-Cuautle, L.G. De La Fraga, O. Guillén-Fernández and A. Silva-Juárez. *Optimization of Integer/Fractional Order Chaotic Systems by Metaheuristics and Their Electronic Realization.* CRC press, 2021. https://doi.org/10.1201/9781003042181.

[25] A. Toktas, U. Erkan, D. Ustun and Q. Lai. Multiobjective design of 2d hyperchaotic system using leader pareto grey wolf optimizer. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2024 Jun. 6 2024. https://doi.org/10.1109/TSMC.2024.3401412.

[26] A. Wadood, B.S. Khan, H. Albalawi and A.M. Alatwi. Design of the novel fractional order hybrid whale optimizer for thermal wind power generation systems with integration of chaos infused wind power. *Fractal. Fract.*, **8**(7), Jul. 2024. https://doi.org/10.3390/fractalfract8070379.

[27] D.S. Weile and E. Michielssen. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety*, **91**(9):992–1007, 2006. https://doi.org/10.1016/j.ress.2005.11.018.

[28] F. Yu, S. Xu, Y. Lin, Y.M. Gracia, W. Yao and Sh. Cai. Dynamic analysis, image encryption application and fpga implementation of a discrete memristor-coupled neural network. *IJBC*, **34**(06), May. 2024. https://doi.org/10.1142/S0218127424500688.

[29] F. Yu, Sh. Xu, X. Xiao, W. Yao, Y. Huang, Sh. Cai and Y. Li. Dynamic analysis and fpga implementation of a 5d multi-wing fractional-order memristive chaotic system with hidden attractors. *Integration, the VLSI Journal*, **96**, May. 2024. https://doi.org/10.1016/j.vlsi.2023.102129.

[30] J. Yu, W. Xie and L. Zhang. Multiobjective optimization of chaotic image encryption based on ABC algorithm and DNA coding. *IJBC*, **34**(05), Apr. 2024. https://doi.org/10.1142/S0218127424500573.

[31] J. Zhang, J. Yang, L. Xu and X. Zhu. The circuit realization of a fifth-order multi-wing chaotic system and its application in image encryption. *Int. J. Circuit Theory Appl.*, **51**(3):1168–1186, Mar. 2023. https://doi.org/10.1002/cta.3490.