STIE PASIM SUKABUMI

IORA International Virtual Conference on Operations Research 2021

Sukabumi, Indonesia, 10-11 November 2021
Email: ioraicor2021@gmail.com Web: https://iora-icor.stiepasim.ac.id/

To Mr./Mrs. Agustina Pradjaningsih

Universitas Jember

Indonesia

Paper ID: IORA21-002

Dear Mr./Mrs. Agustina Pradjaningsih

Thank you for your interest in The IORA International Virtual Conference Operation Research 2021 (IORA-ICOR), and submitting your abstract entitled

"Application of Metaheuristic Algorithm for Solving Fully Fuzzy Linear Equation System"

It is our pleasure to inform you that your abstract has been **accepted** for presentation at IORA-ICOR 2021 on November 10, 2021 in virtual zoom platform. Please submit your full paper and the payment receipt of the conference presenter fee no later than November 9, 2021.

We also attach a publication options form of the conference, please choose one and send the form along with the full paper on the conference's website at https://iora-icor.stiepasim.ac.id/ in the Submissions menu.

Please do not hesitate to contact us at ioraicor2021@gmail.com if you have any questions. Looking forward to your participation in IORA-ICOR 2021.

Sincerely,

INDONESIAN OPERATIONS RESEARCH ASSOCIATION
IORA
ICOR 2021
PASIM-UMPTB
UMI

Conference Chairman
Deden Ardiansyah, S.T. M.Kom.
Vocational School
Pakuan University
Bogor - Indonesia

Application of Metaheuristic Algorithm for Solving Fully Fuzzy Linear Equations System

Merysa Puspita Sari[1, b)], Agustina Pradjaningsih[2, a)] and Firdaus Ubaidillah [3, c)]

[1,2,3] *Mathematics Department, Faculty of Mathematics and Natural Science, Jember University,*
Kalimantan 37, Jember, Indonesia, 68121

[a)] *Corresponding author: agustina.fmipa@unej.ac.id*
[b)] *merysasari19@gmail.com*
[c)] *firdaus_u@yahoo.com*

Abstract. A linear equation is an equation in which each term contains a constant with a variable of degree one or single and can be described as a straight line in a Cartesian coordinate system. A Linear equations system is a collection of several linear equations. A system of linear equations whose coefficients and variables are fuzzy numbers is called a fully fuzzy linear equation system. This study aims to apply a metaheuristic algorithm to solve a system of fully fuzzy linear equations. The objective function used is the minimization objective function. At the same time, the metaheuristic algorithms used in this research are Particle Swarm Optimization (PSO), Firefly Algorithm (FA), and Cuckoo Search (CS). The input in this research is a fully fuzzy linear equation system matrix and parameters of the PSO, FA, and CS algorithms. The resulting output is the best objective function and the variable value of the fully fuzzy linear equations system. The work was compared for accuracy with the Gauss-Jordan elimination method from previous studies with the help of the Matlab programming language. The results obtained indicate that the Particle Swarm Optimization (PSO) algorithm is better at solving fully fuzzy linear equation systems than the Firefly Algorithm (FA) and Cuckoo Search (CS). This case can be seen from the value of the resulting objective function close to the value of the Gauss-Jordan elimination method.

INTRODUCTION

A linear equation is an equation in which each term contains a constant with a variable of degree one or single and can be described as a straight line in a Cartesian coordinate system. Linear equations system is one of the systems of equations in mathematics and is a collection of several linear equations. Two methods can solve a linear equations system: the analytical method and the numerical method. The solution of the linear equations system generally uses the analytical method. Still, if the case is a case that is difficult to solve using analytical methods, numerical methods are used to solve it [13].

Several numerical methods are often used to solve problems with systems of linear equations, namely the Half Interval Method, Linear Interpolation, Secant, Newton-Raphson, and others. However, there are weaknesses in these methods. This weakness has prompted new studies to solve linear equations systems, one of which is by using a metaheuristic algorithm. Metaheuristic algorithms can be defined as advanced heuristic-based algorithms to solve optimization problems efficiently. Some examples of metaheuristic algorithms include the Particle Swarm Optimization (PSO) algorithm, Firefly Algorithm (FA), Cuckoo Search (CS), etc.

Kusumadewi et al. [8] state that fuzzy logic is appropriate to map an input space into an output space. Lotfi A. Zadeh first introduced this concept in 1965. According to Zadeh [17], in fuzzy logic, the degree of membership is known, with a value range of 0 to 1. Based on this explanation, it can say that fuzzy membership lies in the interval [0,1]. There is a scientific development that combines equations systems with fuzzy numbers. A Linear equations system with constants in the form of fuzzy numbers is called a system of fuzzy linear equations. In addition to the fuzzy linear equation system, there is also a fully fuzzy linear equation system. The fully fuzzy linear equations system is a linear equations system in which the coefficients, variables, and constants are fuzzy numbers.

Based on the explanation, the authors are interested in solving the fully fuzzy linear equations system using a metaheuristic algorithm which will then test its accuracy by comparing it with the Gauss-Jordan elimination method. The metaheuristic algorithms in question are Particle Swarm Optimization (PSO), Firefly Algorithm (FA), and Cuckoo Search Algorithm (CS).

FUZZY LOGIC

Lotfi A. Zadeh first introduced fuzzy logic in 1965. According to Kusumadewi et al. [8], fuzzy logic is appropriate to map the input space into an output space. The workings of fuzzy logic consist of input, process, and output. Fuzzy logic has a concept that resembles the human way of thinking. Fuzzy logic can present human knowledge in a mathematical form that is more like human thinking. If, in strict logic, an element has two choices: a value of 1 or true and a value of 0 or false. Some basic definitions are reviewed [6, 7, 9, 10]

Definition 1
Set A is a fuzzy set *that* has a membership degree μ_A such that

$$\mu_A(x) = \begin{cases} 1, & if\ and\ only\ if\ x \in A \\ 0, & if\ and\ only\ if\ x \notin A \end{cases}$$

Definition 2
A fuzzy set A called convex if and only if for every $x_1, x_2 \in X$ and $\lambda \in [0, 1]$ holds

$$\mu_A(\lambda x_1 + (1 - \lambda)x_2) \geq min[\mu_A(x_1), \mu_A(x_2)]$$

Definition 3
A fuzzy set A called normal if and only if the degree of membership is not empty. In other words, we can find the point $x \in X$ with $\mu_A(x) = 1$.

Definition 4
If a fuzzy set is convex and normalized, and its membership function is defined in $\Re$ and is continuous, it is called a fuzzy number. So the fuzzy number in the fuzzy set is represented as an interval of real numbers with fuzzy boundaries.

Definition 5
A fuzzy number $\tilde{A} = (p, q, r)$ is called a triangular fuzzy number if it has a membership function structure such that $\tilde{A} \geq 0$ if and only if $p - q \geq 0$.

Definition 6
The triangular fuzzy number $\tilde{A} = (p, q, r)$ is said to be non-negative if $p \geq 0$.

Definition 7
Two fuzzy numbers $\tilde{A} = (p, q, r)$ and $\tilde{B} = (l, m, n)$ can be called equivalent if and only if $p = l, q = m$, and $r = n$.

Definition 8
A fuzzy number $\tilde{A} = (p, q, r)$ is called triangular fuzzy numbers if its membership function is of the following form:

$$\mu(x) = \begin{cases} 1 - \dfrac{a - x}{b}, & a - b \leq x \leq a, b > 0 \\ 1 - \dfrac{x - a}{c}, & a \leq x \leq a + c, c > 0 \\ 0, & others \end{cases}$$

Suppose $\tilde{A} = (p, q, r)$ and $\tilde{B} = (l, m, n)$ are two triangular fuzzy numbers; according to Dubois et al. [2], arithmetic operations on fuzzy numbers representing triangular curves are defined as follows:

- Addition: $\tilde{A} \oplus \tilde{B} = (p + l, q + m, r + n)$

- Subtraction: $\tilde{A} \ominus \tilde{B} = (p - l, p - m, r - n)$
- Multiplication: if $\tilde{A} \geq 0$ and $\tilde{B} \geq 0$ then $\tilde{A} \otimes \tilde{B} = (pl, pm + mq, pn + lr)$
- Scalar multiplication. Let λ be scalar then

$$\lambda \otimes \tilde{A} = \lambda \otimes (p, q, r) = \begin{cases} (\lambda p, \lambda q, \lambda r), & \lambda \geq 0 \\ (-\lambda p, -\lambda q, -\lambda r), & \lambda < 0 \end{cases}$$

FULLY FUZZY LINEAR EQUATIONS SYSTEM

The fully fuzzy linear equation system is a system of linear equations in which the coefficients, variables, and right-hand side constants are fuzzy numbers. According to Muruganandam et al. [10], the general form of a fully fuzzy linear equation system is as follows:

$$\begin{gathered} (\tilde{a}_{11} \otimes \tilde{x}_1) \oplus (\tilde{a}_{12} \otimes \tilde{x}_2) \oplus \ \dots \ \oplus (\tilde{a}_{1n} \otimes \tilde{x}_n) = \tilde{b}_1 \\ (\tilde{a}_{21} \otimes \tilde{x}_1) \oplus (\tilde{a}_{22} \otimes \tilde{x}_2) \oplus \ \dots \ \oplus (\tilde{a}_{2n} \otimes \tilde{x}_n) = \tilde{b}_2 \\ \vdots \\ (\tilde{a}_{n1} \otimes \tilde{x}_1) \oplus (\tilde{a}_{n2} \otimes \tilde{x}_2) \oplus \ \dots \ \oplus (\tilde{a}_{nn} \otimes \tilde{x}_n) = \tilde{b}_n \end{gathered} \tag{1}$$

where
$\tilde{x}_i$ = Variable i-th
$\tilde{a}_{ij}$ = Coefficient of $\tilde{x}_i$
$\tilde{b}_i$ = right-hand side constants i-th
for $1 \leq i \leq n$ and $1 \leq j \leq n$

Based on Equation 1, the fully fuzzy linear equation system $\tilde{A} \otimes \tilde{x} = \tilde{b}$ can be written as follows:

$$(A, M, N) \otimes (x, y, z) = (b, h, g),$$

whereas by using arithmetic operations we get, $(Ax, Ay + Mx, Az + Nx) = (b, h, g)$,
so based on Definition 7, we get the form:

$$Ax = b;\ Ay + Mx = h;\ Az + Nx = g \tag{2}$$

PARTICLE SWARM OPTIMIZATION (PSO) ALGORITHM

Kennedy and Eberhart developed Particle Swarm Optimization (PSO) in 1995. Particle Swarm Optimization (PSO) is a swarm intelligence algorithm that simulates the social behavior of a group of birds. The PSO algorithm uses a population of a set of particles, where each particle represents a possible solution to an optimization problem. Each particle has a function value that is evaluated by the function to be optimized. The velocity (velocity) adapted from the search area to move and stored as the best position ever achieved. So that from the behavior of the bird collection, which is then used to solve optimization problems.

Each particle revolves around the multidimensional search space and adjusts its position based on the particle's experience and the neighboring particle's experience. Each particle has a position $z_i = [z_{i1}, z_{i2}, \dots, z_{iN}]$ and a velocity $v_i = [v_{i1}, v_{i2}, \dots, v_{iN}]$, where i represents the i-th particle and N represents the dimensions of the search space or the number of unknown variables. Initialization of the PSO algorithm begins by randomly assigning the initial position of the particle (solution) and then finding the optimal value. In each iteration, each particle updates its position following the two best values, namely the best solution that has been obtained by each particle (pbest) and the best solution in the population (gbest). After getting the best two values, the position and velocity of the particles are updated using the following equation [12]:

$$v_i^{t+1} = \theta v_i^t + c_1 r_1 (pbest_i^t - z_i^t) + c_2 r_2 (gbest_i^t - z_i^t) \tag{3}$$

$$z_i^{t+1} = z_i^t + v_i^{t+1} \tag{4}$$

where
θ = Weight coefficient of inertia
v_i^t = The velocity of the i-th particle in the t-th iteration
z_i^t = Solution (position) of the i-th particle in the t-th iteration
c_1, c_2 = Acceleration coefficient
r_1, r_2 = Random variable

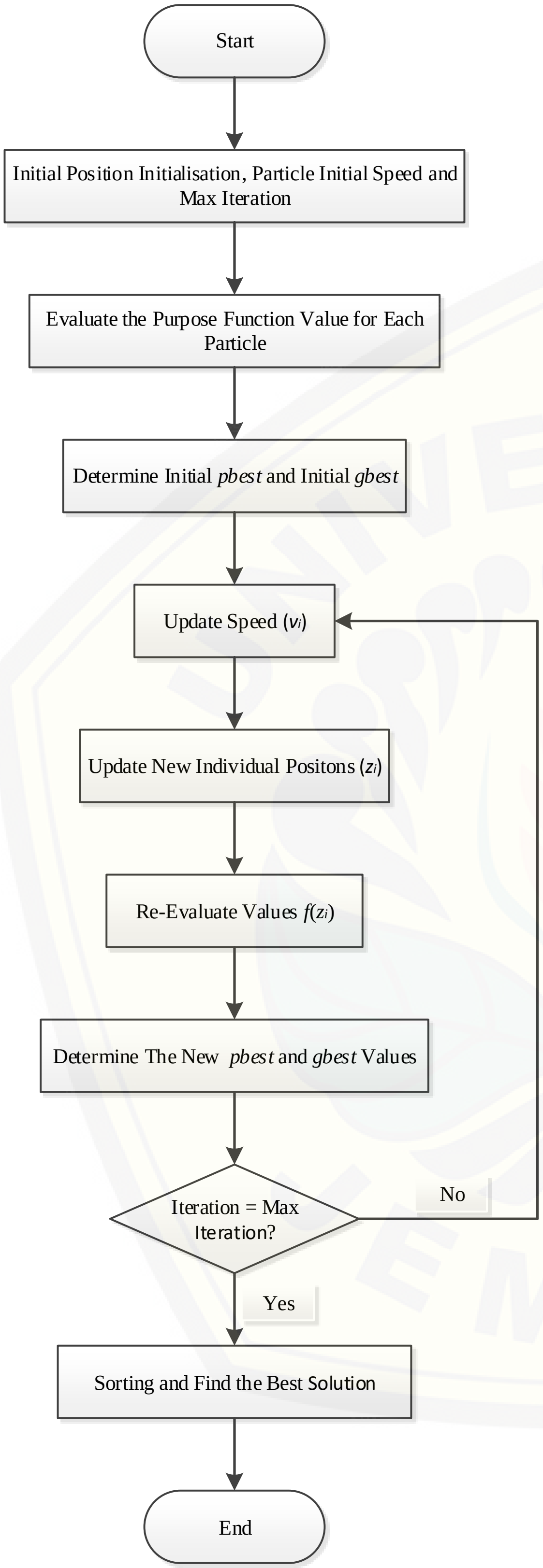

Start
Initial Position Initialisation, Particle Initial Speed and Max Iteration
Evaluate the Purpose Function Value for Each Particle
Determine Initial *pbest* and Initial *gbest*
Update Speed (v_i)
Update New Individual Positons (z_i)
Re-Evaluate Values $f(z_i)$
Determine The New *pbest* and *gbest* Values
Iteration = Max Iteration?
No
Yes
Sorting and Find the Best Solution
End

FIGURE 1. Particle Swarm Optimization algorithm

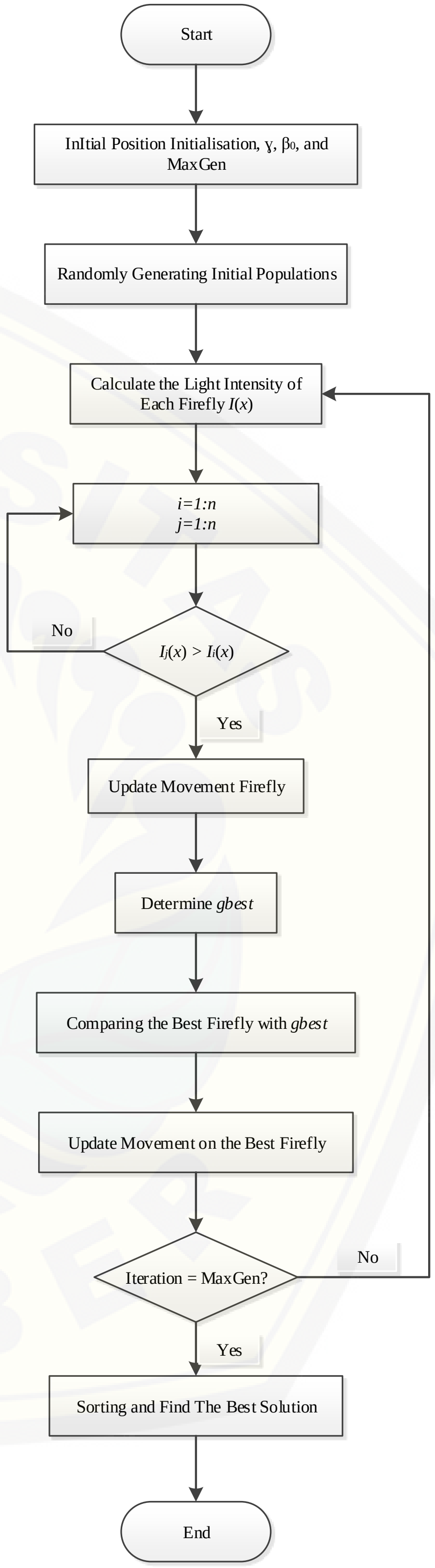

Start
InItial Position Initialisation, ɣ, β_0, and MaxGen
Randomly Generating Initial Populations
Calculate the Light Intensity of Each Firefly $I(x)$
$i=1:n$
$j=1:n$
$I_j(x) > I_i(x)$
No
Yes
Update Movement Firefly
Determine *gbest*
Comparing the Best Firefly with *gbest*
Update Movement on the Best Firefly
Iteration = MaxGen?
No
Yes
Sorting and Find The Best Solution
End

FIGURE 2. Firefly algorithm

The acceleration parameters c_1,c_2, r_1, and r_2, are random numbers with uniform distribution in the interval [0,1]. The parameters c_1 and c_2 are positive constants to control how far a particle will move in a single iteration. Low values allow the particle to travel far from the target area before pulling back, while high values result in a sudden movement towards the target area. Based on previous research, it has been shown that the acceleration coefficient must meet $c_1 + c_2 < 4(1+\theta)$ and $c_1 = c_2$; assigning different values to c_1 and c_2 usually leads to an increase in work [3].

The weight coefficient of inertia (θ) serves to reduce the velocity of the particle. A high value of the weight coefficient of inertia causes an increase in the share of global exploration (global exploration), while a low value emphasizes local search. To not focus too much on one search and keep looking for a new search area in a specific dimension space, it is necessary to find a balanced value of the inertia weight coefficient (θ) to maintain global and local searches. The following formula can find the coefficient of inertia weight:

$$\theta = \theta_{max} - i\frac{\theta_{max} - \theta_{min}}{i_{max}} \tag{5}$$

where θ_{max} and θ_{min}n are the initial and final values of the coefficient of inertia weight, respectively, i_{max} is the maximum number of iterations used, and i is the iteration. Previous research shows that the weight coefficient of inertia must be in the range of 0 and 1 [4]. Particle Swarm Algorithm can be seen in the Figure 1.

FIREFLY ALGORITHM (FA)

The Firefly Algorithm (FA) was developed in late 2007 by Xin-She Yang at Cambridge University. This algorithm is based on the blinking pattern and behavior of fireflies. According to Yang [15], the FA uses three rules that are considered ideal, i.e.:

- Fireflies are *unisex*, so one firefly can be attracted to another firefly regardless of gender.
- The level of attraction of the firefly will be proportional to the level of brightness of the firefly. The farther the distance between the fireflies, the brightness level of the fireflies will decrease or disappear. So for every firefly that blinks, a less bright firefly (dimmer) will approach a brighter firefly. If neither of the two fireflies is brighter, then the fireflies will move randomly.
- The brightness level of fireflies will be determined by the objective function of the given problem.

There are two important problems in the FA algorithm: the variation of light intensity and the formulation of attractiveness. The objective function and attractiveness will determine the brightness of the firefly is proportional to the brightness. So for every two fireflies that blink, the firefly with the less bright light will move towards the firefly with the brighter light. The objective function influences the light intensity in fireflies. The level of light intensity for fireflies x is formulated as follows:

$$I(x) = \frac{1}{1 + f(x)} \tag{6}$$

where the value of $I(x)$ is the light intensity level on the fireflies x which is inversely proportional to the solution $f(x)$ or the objective function of the problem to be searched.

Attractiveness (β) is relative because the intensity of light depends on other fireflies. Therefore, the assessment results will differ depending on the distance between one firefly and another (r_{ij}). There are several cases where the light intensity will decrease from the source due to being absorbed by media such as air and others. So that attractiveness (β) can be determined with a distance (r) as follows:

$$\beta = \beta_0 e^{-\gamma r^2} \tag{7}$$

where β_0 is the attraction when there is no distance between the fireflies ($r = 0$) and $\gamma \in [0, \infty]$ is the light absorption coefficient [1].

The distance or the distance between the fireflies i and j at positions z_i and z_j is each a Cartesian distance which is formulated as follows:

$$r_{ij} = \| z_i - z_j \| = \sqrt{\sum_{t=1}^{n} \left(z_i^t - z_j^t\right)^2} \tag{8}$$

wherez_i^t is the t-th component of z_i on firefly i and $z_j^t$is the t-th component of z_j on firefly j.
The movement of firefly i due to attraction to other fireflies j, whose light intensity is brighter, is called movement. The movement causes the change in the position of the fireflies according to the following formula:

$$z_i^{t+1} = z_i^t + \beta\left(z_j^t - z_i^t\right) + \alpha\left(rand - \frac{1}{2}\right) \tag{9}$$

where the first term (z_i^t) is the old position of the firefly, the second term occurs because of interest; the third term is the random movement of the fireflies with being the coefficient of the random parameter, and rand is a random number in the interval $[0,1]$. The implementation of the FA algorithm usually uses $\beta_0 = 1, \alpha \in [0,1]$, and $\gamma \in [0, \infty]$ [1]. Firefly Algorithm can be seen in the Figure 2.

CUCKOO SEARCH ALGORITHM (CS)

Cuckoo Search (CS) is a metaheuristic algorithm developed by Xin-she Yang and Deb in 2009. This algorithm is inspired by the parasitic nature of several cuckoo species that lay eggs in the nests of other host birds (other species). Some host birds can come into direct conflict with annoying cuckoos. For example, if the host bird discovers that the egg in the nest is not its own, it will either discard the foreign egg or leave the nest and build the nest elsewhere. Some cuckoo species, such as the New World brood-parasite Tapera, have evolved to the point where the color and pattern of their eggs mimic those of the chosen host species [11]. Cuckoo Search Algorithm can be seen in the Figure 3.

Yang and Deb [14] used Lévy Flights, an extension of the random walk, to develop the Cuckoo Search (CS) algorithm. Lévy Flights is a random walk distributed by Lévy. Lévy Flights can map fruit flies in search of food. Fruit flies will focus on one point when looking for food; if fruit flies feel the food is running out, they will look elsewhere. This study also explained that Lévy Flights helped the CS algorithm in the search because the search steps were getting wider and wider. The CS algorithm uses Lévy Flights to get better accuracy than other optimization algorithms in determining the optimal point. Yang and Deb, in their research in 2009, stated that specific rules must be met in the use of this algorithm, i.e.:

- Each bird lays eggs at the same time and then throws the eggs in a randomly selected nest.
- Bird nests that are considered the best will be continued for the next generation.
- The number of bird nests in a colony is fixed.
- The probability of recognizing a cuckoo egg (P_a) placed in the host bird's nest is 0 and 1.

The last rule can be approximated with the parameter P_a to determine the worst solution of nests to be replaced with new nests at random. In the maximization problem, to simplify the application, a simple representation can be used: each egg in the nest represents a solution, and the eggs represent a new solution. The goal is to use a new solution that can replace the solution in the hive. Then the eggs in the nest will be selected and evolved by removing the eggs considered less good. In some cases, the original parent nest may have two eggs; in other words, a nest may contain more than one solution. However, to simplify the problem, a nest can only store one solution [15].
The step randomization process with Lévy Flights uses the following formula:

$$z_i^{(t+1)} = z_i^t + s \oplus Lévy\ (\alpha, \beta, \gamma, \delta) \tag{10}$$

where $z^{(t+1)}$ is the new solution, z_i^t is the old solution, , s is the step size associated with the level of the problem being worked on, and the sign $\oplus$ which means times or multiplies. Lévy Flights or what is called Lévy Stable Distribution in MATLAB, has several parameters, including [5]:

- Alpha (α) is the first form parameter with an interval of $0 < \alpha \leq 2$.
- Beta (β) is the second form parameter with an interval of $-1 \leq \beta \leq 1$.
- Gamma (γ) is a scale parameter that has an interval of $0 < \gamma < \infty$.
- Delta (δ) is a location parameter that has an interval of $-\infty < \delta < \infty$.

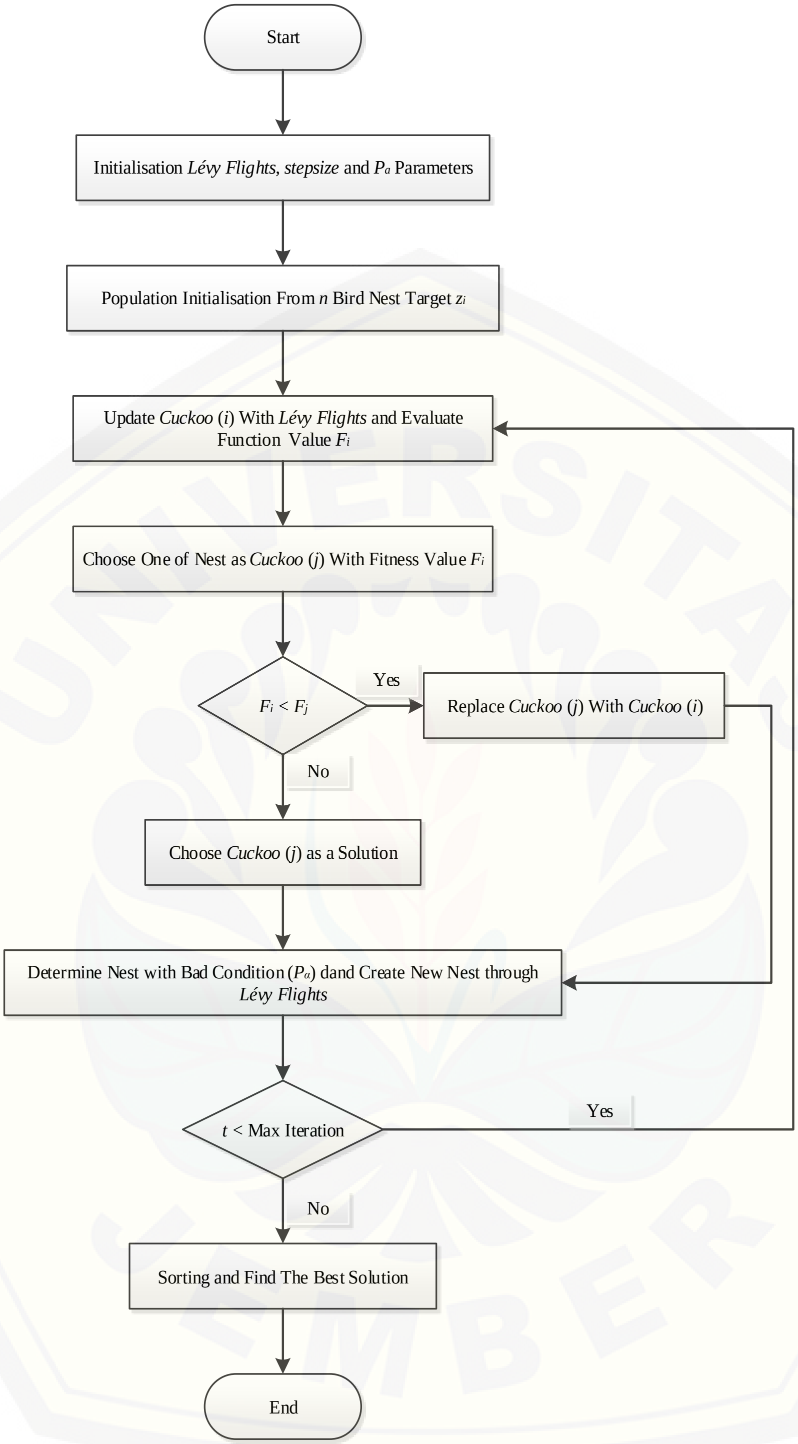
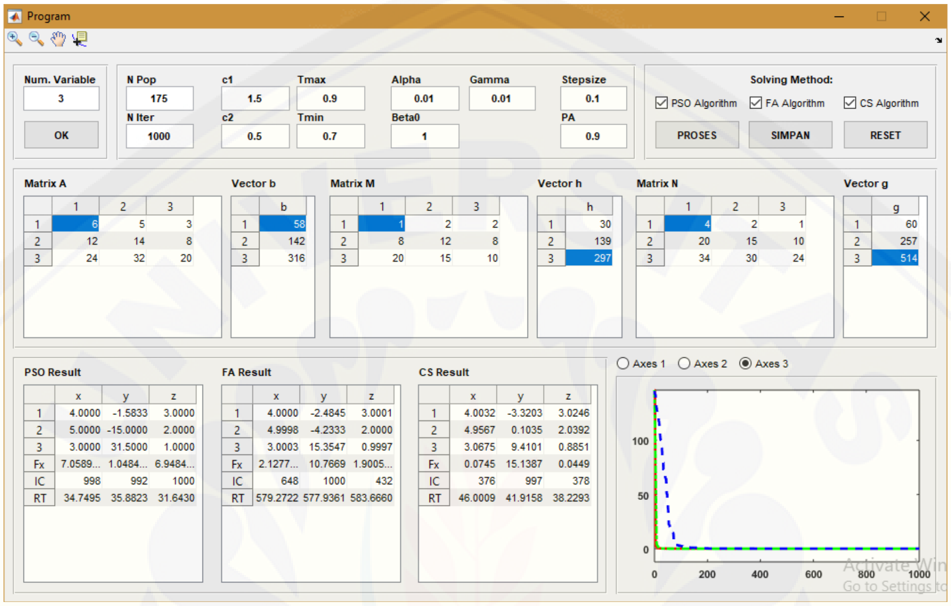

Start
Initialisation *Lévy Flights*, *stepsize* and P_a Parameters
Population Initialisation From *n* Bird Nest Target z_i
Update *Cuckoo* (*i*) With *Lévy Flights* and Evaluate Function Value F_i
Choose One of Nest as *Cuckoo* (*j*) With Fitness Value F_i
$F_i < F_j$
Yes
Replace *Cuckoo* (*j*) With *Cuckoo* (*i*)
No
Choose *Cuckoo* (*j*) as a Solution
Determine Nest with Bad Condition (P_a) dand Create New Nest through *Lévy Flights*
t < Max Iteration
Yes
No
Sorting and Find The Best Solution
End

FIGURE 3. Cuckoo Search algorithm

RESULTS

In this study, the authors create a simulation program for the PSO, FA, and CS algorithms using MATLAB software in the form of a Graphical User Interface (GUI). The program is used to test the effect of the parameters of each algorithm. The results of comparing the PSO, FA and CS algorithms on the solution of a fully fuzzy linear equation system and the influence of these parameters can be seen in Figure 4.



Num. Variable: 3
N Pop: 175
N Iter: 1000
c1: 1.5
c2: 0.5
Tmax: 0.9
Tmin: 0.7
Alpha: 0.01
Gamma: 0.01
Beta0: 1
Stepsize: 0.1
PA: 0.9

Solving Method: ☑ PSO Algorithm ☑ FA Algorithm ☑ CS Algorithm

Matrix A

	1	2	3
1	6	5	3
2	12	14	8
3	24	32	20

Vector b

	b
1	58
2	142
3	316

Matrix M

	1	2	3
1	1	2	2
2	8	12	8
3	20	15	10

Vector h

	h
1	30
2	139
3	297

Matrix N

	1	2	3
1	4	2	1
2	20	15	10
3	34	30	24

Vector g

	g
1	60
2	257
3	514

PSO Result

	x	y	z
1	4.0000	-1.5833	3.0000
2	5.0000	-15.0000	2.0000
3	3.0000	31.5000	1.0000
Fx	7.0589...	1.0484...	6.9484...
IC	998	992	1000
RT	34.7495	35.8823	31.6430

FA Result

	x	y	z
1	4.0000	-2.4845	3.0001
2	4.9998	-4.2333	2.0000
3	3.0003	15.3547	0.9997
Fx	2.1277...	10.7669	1.9005...
IC	648	1000	432
RT	579.2722	577.9361	583.6660

CS Result

	x	y	z
1	4.0032	-3.3203	3.0246
2	4.9567	0.1035	2.0392
3	3.0675	9.4101	0.8851
Fx	0.0745	15.1387	0.0449
IC	376	997	378
RT	46.0009	41.9158	38.2293

○ Axes 1 ○ Axes 2 ● Axes 3

FIGURE 4. MATLAB program display

Numerical Results

The parameters tested: for PSO are $\theta_{max}, \theta_{min}, c_1,$ and c_2, for FA are α and γ, for CS are s and P_a. The program parameter test was carried out ten times. The best parameter values obtained are used for the final simulation of the PSO, FA, and CS algorithms. The reader can obtain the complete result of the parameter test through agustina.fmipa@unej.ac.id.

a. PSO Parameters

1. θ_{min} and θ_{max} Parameters Test
   The parameter values of tested are $0.1, 0.3, 0.5, 0.7$ and 0.9, with the supporting parameter values, namely, $N\ pop = 100, iteration = 1000, c_1 = 1$ and $c_2 = 1$. The best combination obtained is $\theta_{max} = 0.9$ and $\theta_{min} = 0.7$.
2. c_1 and c_2 Parameters Test
   The parameter values of tested are $0.5, 1, 1.5$ and $2,$ with the supporting parameter values, namely, $N\ pop = 100, iteration = 1000, \theta_{min} = 0.7$ and $\theta_{max} = 0.9$. The best combination obtained is $c_1 = 1.$ and $c_2 = 0.5$.

b. FA Parameters

1. α Parameter Test
   The parameter values of tested are $0.01, 0.05, 0.1, 0.3, 0.5, 0.7$ and 0.99, with the supporting parameter values, namely, $N\ pop = 100, iteration = 1000, \beta_0 = 1$ and $\gamma = 1$. The best value obtained is $\alpha = 0.01$. The results obtained from the test are shown in Figure 5, where the x axis represents the α and the y axis represents the total value of $f(x)$.

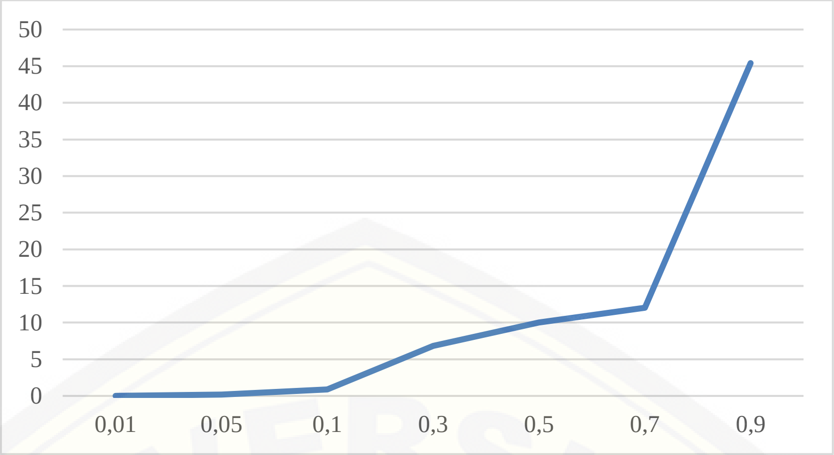


FIGURE 5. α parameter test results

2. γ Parameter Test

The parameter values of tested are $0.01, 0.05, 0.1, 0.2, 0.5, 1, 1.5, 2, 5$ and 10, with the supporting parameter values, namely, $N\ pop = 100, iteration = 1000, \alpha = 0.01$ and $\beta_0 = 1$. The best value obtained is $\gamma = 0.1$. The results obtained from the test are shown in Figure 6, where the x axis represents the γ and the y axis represents the total value of $f(x)$.

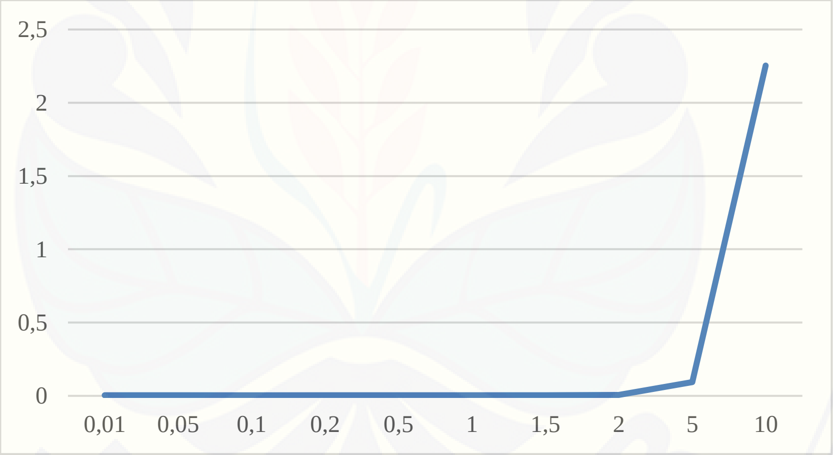


FIGURE 6. γ parameter test results

c. CS Parameters

1. s Parameter Test

The parameter values of tested are $0.01, 0.05, 0.1, 0.2, 0.5, 1, 1.5, 2, 5$ and 10, with the supporting parameter values, namely, $N\ pop = 100, iteration = 1000, P_a = 0.5$ and Lévy Flights parameter generated automatically in the MATLAB program. The best value obtained is s=0.1. The results obtained from the test are shown in Figure 7, where the x axis represents the s and the y axis represents the total value of $f(x)$.

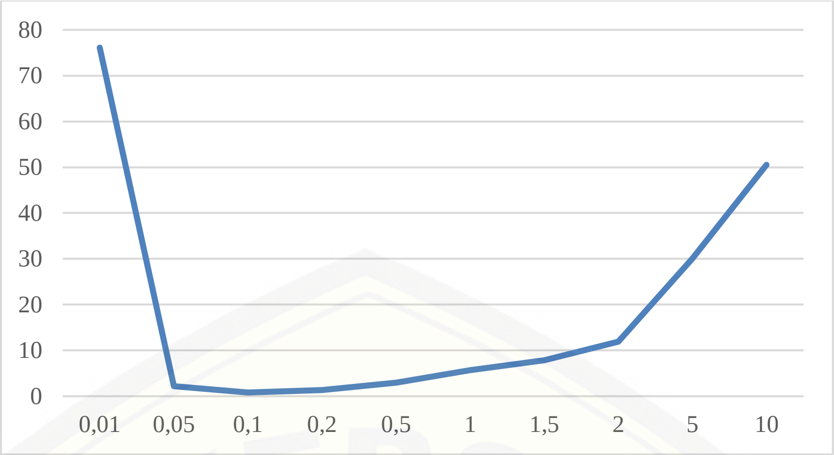


FIGURE 7. s parameter test results

2. P_a Parameter Test
   The parameter values of tested are $0.1, 0.3, 0.5, 0.7$ and 0.9, , with the supporting parameter values, namely, $N\ pop = 100, iteration = 1000, s = 0.1$ and Lévy Flights parameters which are generated automatically in the MATLAB program. The best value obtained is $P_a = 0.9$. The results obtained from the test are shown in Figure 8, where the x axis represents the P_a and the y axis represents the total value of $f(x)$.

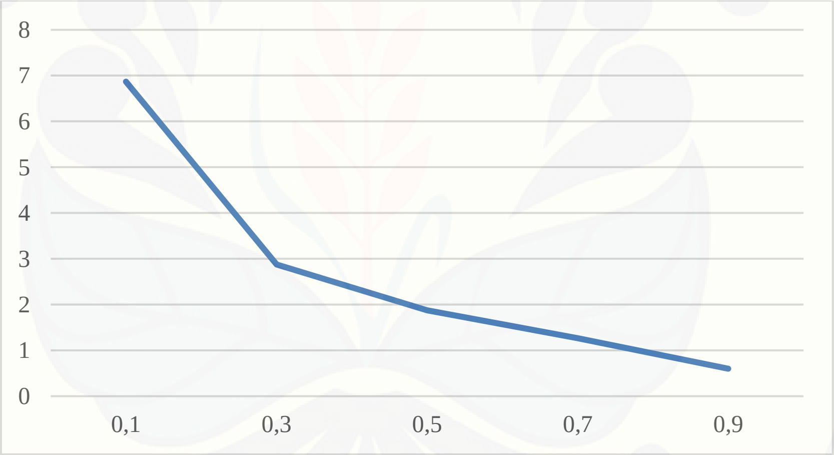


FIGURE 8. P_a parameter test results

d. $N\ pop$ Parameter Test
   The parameter values of tested are $25, 50, 75, 100, 125, 150$ and 175, with iteration=1000. The parameter values of the PSO algorithm used are, $\theta_{max} = 0.9, \theta_{min} = 0.7, c_1 = 1.5$ and $c_2 = 0.5$. The parameter values of the FA algorithm used are $\alpha = 0.01, \beta_0 = 11$ and $\gamma = 0.1$. Meanwhile, the CS algorithm parameter values used are $s = 0.1, P_a = 0.9$ and the Lévy Flights parameter. The best value obtained is $N\ pop = 175$. The results obtained from the test are shown in Figure 9, where the x axis represents the $N\ pop$ and the y axis represents the total value of $f(x)$.

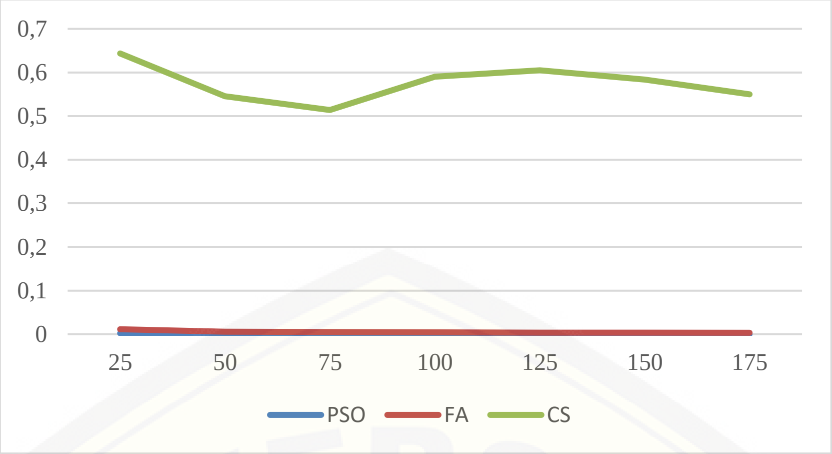


FIGURE 9. N pop parameter test results

Final Simulation

The results of the above calculations are used to solve the problem of a fully fuzzy linear equation system [10], which has been solved using Gauss Jordan elimination. The problem of the fully fuzzy linear equation system is

$$(6,1,4)\otimes\tilde{x}_1\oplus(5,2,2)\otimes\tilde{x}_2\oplus(3,2,1)\otimes\tilde{x}_3 = (58,30,60)$$

$$(12,8,20)\otimes\tilde{x}_1\oplus(14,12,15)\otimes\tilde{x}_2\oplus(8,8,10)\otimes\tilde{x}_3 = (142,139,257)$$

$$(24,10,34)\otimes\tilde{x}_1\oplus(32,30,30)\otimes\tilde{x}_2\oplus(20,19,24)\otimes\tilde{x}_3 = (216,297,514)$$

The solution of fully fuzzy linear equations using PSO, FA and CS algorithms and their comparison with Gauss-Jordan Elimination [10] can be seen in Table 1 with $\tilde{x}_1 = (x_1, y_1, z_1)$, $\tilde{x}_2 = (x_2, y_2, z_2)$, $\tilde{x}_3 = (x_3, y_3, z_3)$.

TABLE 1. Comparison results PSO, FA and CS algorithm with Gauss-Jordan elimination method

Variabel		PSO	FA	CS	Gauss-Jordan
$\tilde{\boldsymbol{x}}_1$	x_1	4	4	4.001048	4
	y_1	−1.58488	−2.48789	−2.93751	1
	z_1	3	3	2.997779	3
$\tilde{\boldsymbol{x}}_2$	x_2	5	4.999978	5.004566	5
	y_2	−14,9961	−4.1594	−0.06793	0.5
	z_2	2	2.000039	1.982704	2
$\tilde{\boldsymbol{x}}_3$	x_3	3	3.000032	2.99109	3
	y_3	31.49563	15.24044	9.231667	0.5
	z_3	1	0.999918	1.032798	1

Based on Table 1, it is known that the calculation using the PSO algorithm is the result that is closest to the Gauss-Jordan elimination method, although there are still unequal values. Different values can be caused by using a small number of populations and iterations. However, this study shows that the PSO algorithm has better accuracy in solving a fully fuzzy linear equation system than the FA and CS algorithms.

CONCLUSION

Metaheuristic algorithms, especially Particle Swarm Optimization (PSO), Firefly Algorithm (FA), and Cuckoo Search (CS), can solve the fully fuzzy linear equation system. Based on the results obtained, the Particle Swarm Optimization (PSO) algorithm has the best accuracy results. It is close to the results of the Gauss-Jordan elimination method when compared to the Firefly Algorithm (FA) and Cuckoo Search (CS) algorithms.

REFERENCES

1. M. K. A. Ariyaratne, T. G. I. Fernando and S. Weerakoon. A modified firefly algorithm to solve univariate nonlinear equations with complex roots. *International Conference on Advances in ICT for emerging Regions*: 160-167. 2015
2. D. Dubois and H. Prade. *Fuzzy Sets and Systems*: *The Applications*. New York: Academic Press.
3. M. Jiang, Y. P. Luo, and S. Y. Yang. Stochastic convergence analysis and parameter slection of the standar particle swarm optimization algorithm. *Information Processing Letters*. 102: 8-16. 2007.
4. S. Kessentini and D. Barchiesi. Particle swarm optimization with adaptive inertia weight. *International Journal of Machine Learning and Computing*. 5(5): 368-373. 2015.
5. M. Kateregga, S. Mataramvura, and D. Taylor. Parameter estimation for stable distributions with application to commodity futures log-returns. *Cogent Economics & Finance*. 5: 1318813. 2017.
6. A. Kauffman and M. M. Gupta. *Introduction Fuzzy Arithmetic*. New York: Van Nostrand Reinhold. 1985.
7. G. J. Klir, Clair U. S., and B. Yuan. Fuzzy *Set Theory Fundations and Applications*. New Jersey: Prentice Hall. 1995.
8. S. Kusumadewi and H. Purnomo. *Aplikasi Logika* Fuzzy *untuk Pendukung Keputusan*. Yogyakarta: Graha Ilmu. 2004.
9. K. H. Lee. *First Course on* Fuzzy *Theory and Applications*. New York: Springer Berlin Heidelberg. 2005.
10. S. Muruganandam, K. A. Razak and K. Rajakumar. Solving fully fuzzy linear systems by Gauss Jordan Elimination Method. *Internatiol Coference on Physics and Photonics Processes in Nano Sciences*. 2019. DOI: http://dx.doi.org/10.1088/1742-6596/1362/1/012087.
11. R. B. Payne, M. D. Sorenson, and K. Klitz. *The Cuckoos*. England: Oxford University Press. 2005.
12. A. Rosita, Y. Purwananto, and R. Soelaiman. Implementasi algoritma particle swarm untuk menyelesaikan sistem persamaan nonlinear. *Jurnal Teknik ITS*. 1: 2301-9271. 2012.
13. Sahid. *Pengantar Komputasi Numerik Dengan Matlab*. Yogyakarta: Andi. 2005.
14. X. S. Yang and S. Deb. Cuckoo search via levy flights. *Proc of World Congress on Nature and Biologically Inspired Computing*: 210-214. 2009.
15. X. S. Yang. *Nature-Inspire Metaheuristik Algorithm*. Edisi 2. United Kingdom: Luniver Press. . 2010.
16. X. S. Yang and S. Deb. Multiobjective cuckoo search for design optimization. *Computers and Operations Research*. 40: 1616-1624. 2013.
17. L. A. Zadeh. Fuzzy sets. *Information and Control*. 8(2): 338-353. 1965.