Karadeniz Fen Bilimleri Dergisi, 13(4), 1361-1376, 2023. DOI: 10.31466/kfbd.1275692

KFBD

Karadeniz Fen Bilimleri Dergisi
The Black Sea Journal of Sciences
ISSN (Online): 2564-7377 https://dergipark.org.tr/tr/pub/kfbd

Araştırma Makalesi / Research Article

A Fuzzy Numerical Simulation-based Heuristic Method for Fully Fuzzy Systems of Linear Equations

Hande GÜNAY AKDEMIR[1*]

Abstract

In this paper, a new method is proposed to find the approximate solutions to fully fuzzy systems of linear equations (FFSLEs). The technique integrates a bisection method with Fuzzy Numerical Simulation (FNS). The procedure starts with generating single values of fuzzy parameters and solving the resulting crisp problems repeatedly to determine the lower and upper bounds of the solutions. After computing the mean lower and upper bound values, the obtained supremum and infimum values are considered to be the lower and upper bounds of the solutions, respectively. It is attempted to improve solutions by considering an error function related to the sum of the absolute differences between the corresponding lower and upper bounds of the left and right sides of the equalities. When very large intervals are obtained for the solutions, the bisection algorithm is applied to reduce the error value. The method intends to solve square systems of large dimensions for arbitrary fuzzy numbers (FNs) by removing non-negativity confinements of the variables and/or coefficients to be more realistic. After the computational method is presented thoroughly, some benchmark examples are finally provided.

Keywords: Approximate solution, Bisection method, Fully fuzzy systems of linear equations, Fuzzy numerical simulation.

Tam Bulanık Lineer Denklem Sistemleri için Bulanık Sayısal Simülasyon Tabanlı Sezgisel Bir Yöntem

Öz

Bu çalışmada, tam bulanık lineer denklem sistemlerine yaklaşık çözümler bulmak için yeni bir yöntem önerilmiştir. Teknik, bir ikiye bölme yöntemini bulanık sayısal simülasyon ile bütünleştirir. Prosedür, bulanık parametrelerin tek değerlerinin üretilmesi ve çözümlerin alt ve üst sınırlarının belirlenmesi için ortaya çıkan net problemlerin tekrar tekrar çözülmesiyle başlar. Ortalama alt ve üst sınır değerleri hesaplandıktan sonra, elde edilen supremum ve infimum değerler sırasıyla çözümlerin alt ve üst sınırları olarak kabul edilir. Eşitliklerin sağ ve sol taraflarının karşılık gelen alt ve üst sınırları arasındaki mutlak farkların toplamına ilişkin bir hata fonksiyonu ele alınarak çözümler geliştirilmeye çalışılmıştır. Çözümler için çok büyük aralıklar elde edildiğinde, hata değerini azaltmak için ikiye bölme algoritması uygulanır. Yöntem, daha gerçekçi olmak için değişkenlerin ve/veya katsayıların negatif olmayan sınırlamalarını kaldırarak keyfi bulanık sayılar için büyük boyutlu kare sistemleri çözmeyi amaçlar. Hesaplama yöntemi kapsamlı bir şekilde sunulduktan sonra, son olarak bazı kıyaslama örnekleri verilmektedir.

Anahtar Kelimeler: Yaklaşık çözüm, İkiye bölme yöntemi, Tam bulanık lineer denklem sistemleri, Bulanık sayısal simülasyon.

[1]Giresun University, Faculty of Arts and Science, Department of Mathematics, Giresun, Turkey, hande.akdemir@giresun.edu.tr

[1]https://orcid.org/0000-0003-3241-1560

[*]**Sorumlu Yazar/Corresponding Author**

Geliş/Received: 02.04.2023 **Kabul/Accepted:** 11.11.2023

1. Introduction

Real-world systems are too complex to be defined by exact parameters; hence, imprecision or approximation is often involved. When the coefficients of the system are imprecise and only some imperfect subjective knowledge about the actual values of the parameters is available, systems of simultaneous fuzzy linear equations appear in various areas of science, including analysis, operations research, and engineering. The solutions to such systems cannot be found precisely since we face fuzziness inherited from the ambiguity of the parameters. Therefore, we consider linear systems having a fuzzy vector as the right-hand side, a fuzzy coefficient matrix, and a fuzzy vector of unknowns.

For instance, while designing an engineering system optimally, we must solve systems of equations with imprecise variables and parameters resulting from experimental data, expert knowledge, and structural uncertainties. In the economic order of quantity models of supply chain management, we encounter supply disruption and demand fluctuation due to delays and failures in supply as a result of quality issues, machine breakdowns, power cuts, labor strikes, accidents, and natural disasters. This makes it necessary to address uncertainty in supply-demand equilibrium equations. Other areas of application involve manufacturing model representation, chemical reaction balance, electrical network flow problems, income-expenditure equilibrium in economics, and curve fitting.

There exist numerous approaches that efficiently address several important classes of such problems. A typical approach to solving problems involving uncertain parameters is to adapt exact methods with some modifications. Early studies included the integration of fuzzy arithmetic with crisp techniques, for example, decomposition. Some relevant literature to date is given as follows. Dehghan and colleagues (Dehghan and Hashemi, 2006; Dehghan et al., 2006; Dehghan et al., 2007) investigated approximate non-negative solutions of square systems using extended fuzzy operations and several well-known crisp iterative techniques. See also (Ezzati et al., 2012; Kumar et al., 2012; Ezzati et al., 2014). Allahviranloo et al. (2011) proposed a method to obtain maximal and minimal symmetric solutions, which are not necessarily non-negative, of fully fuzzy linear systems based on 1-cut expansion. Square systems with arbitrary solutions considering interval arithmetic, 0, and 1-cuts were examined in (Moloudzadeh et al., 2013; Allahviranloo et al., 2014; Behera and Chakraverty, 2017). Otadi et al. (2011), Mosleh (2013), and Jeswal and Chakraverty (2019) computed arbitrary approximate solutions of square systems with triangular coefficients with fuzzy neural networks. By using linear programming with equality constraints, Otadi and Mosleh (2012) and Babbar et al. (2013a) studied the nonnegative fuzzy solution to fully fuzzy matrix equations with triangular and trapezoidal coefficients, respectively. For the arbitrary triangular solutions of FFSLEs, Babbar et al.

(2013b) suggested simple numerical methods based on decomposition. Mosleh and Otadi (2015) and Guo et al. (2018) discussed the concept of the fuzzy inverse matrix for LR-type arbitrary fuzzy solutions. Strong arbitrary solutions were covered in (Ahlatcioglu et al., 2016; Kocken et al., 2016; Albayrak, 2017) for square and non-square systems with triangular and trapezoidal coefficients, using mixed integer programming. Both trapezoidal and hexagonal coefficients were taken into account in (Ziqan et al., 2022).

Chanas and Nowakowski (1988) proposed an approach that enables assigning a precise numerical value to a fuzzy variable by simulating values of uniform random variables related to the fuzzy variable. Rao and Chen (1998) presented a computational methodology based on a search algorithm and bisection method for the solution of simultaneous linear equations involving fuzzy input parameters. Akdemir and Kocken (2022) constructed an FNS-based bisection procedure for a fuzzy linear regression model. The defuzzification-based technique was described in (Allahviranloo et al., 2008) for resolving FFSLEs, and its basics were on the ideas of value, ambiguity, and fuzziness. In order to determine the multiplicative inverse of a square fuzzy matrix with triangular or trapezoidal components, Akdemir (2023) provided a computational approach based on FNS and bisection.

Considering the given literature, only small-scale problems were mostly included in all studies. Calculating the exact solutions may be difficult and impractical, especially for larger dimensions. When the problem size and complexity grow, the application of the proposed methods becomes more challenging. In such large-scale problems, adopting a computational or heuristic method considering approximate solutions can complement and enhance the literature.

In this study, we propose a two-stage heuristic that solves a large number of systems in the first stage using the generated parameters to identify the initial intervals containing the set of possible values of the variables required for the execution of the second stage's bisection algorithm. We aim to approximate the right-hand side target values by minimizing the dissimilarity between the two sides. Once we identify the bounds for the solutions, the bisection procedure allows us to search for the appropriate intervals to deal with overestimating and underestimating by narrowing the wide initial intervals. Depending on whether the overall error decreases or not, the estimated intervals are narrowed with a predetermined number of bisection iterations. There could be asymmetry in the left-side and right-side narrowing. A significant feature of our algorithm is that it enables us to make use of crisp equation-solving software. The approach can be used even in cases when the coefficient matrix is crisp.

The proposed method is a straightforward, two-stage approach that takes advantage of computer technology and intertwines predefined, effective, crisp techniques. Scenarios, which are used to represent particular realizations, are generated and then replaced in the equations, and the resulting crisp systems are solved; thereby, we acquire projections that indicate which intervals contain the solutions. These initial intervals, which are very wide by nature, are narrowed down by the bisection method to reduce the error. The method is a procedure that improves the error since it does not ignore any term in the multiplication operation, compared to other methods in the literature that provide approximate solutions. Even though the multiplication uses the max and min functions, since crisp systems are solved, this operation gets simpler.

The paper is structured as follows: In the following section, we give a brief introduction to fuzzy arithmetic and then define FFSLEs. In Section 3, we develop a numerical method for calculating the approximate solution to FFSLEs. In Section 4, we provide numerical examples to illustrate our methodology. We conclude in Section 5.

2. Materials and Methods

Before explaining the method, we provide some fundamental concepts about FNs.

Definition 2.1. (Fuzzy Set) A set

$$\tilde{a} = \left\{ \langle x, \mu_{\tilde{a}}(x) \rangle \middle| x \in X \right\}$$

is said to be a fuzzy set on a universal set X, where the function $\mu_a : X \to [0,1]$ denotes the membership function of $\tilde{a}$.

A normal fuzzy subset of the real line, i.e., there exists $\exists x \in \mathbb{R}$ such that $\mu(x) = 1,$ with a convex membership function is called a FN.

Definition 2.2. ($LR-$type FN) Let $L, R : \mathbb{R}^+ \to [0,1]$ be decreasing functions with $L(0)=1,\ L(1)=0,\ R(0)=1,\ R(1)=0,$ and $L(x), R(x) < 1$ for $x > 0.$ The FN $\tilde{a}$ is called $LR-$type if there exist $l, r\ (l < r), \alpha > 0$ and $\beta > 0,$ with the membership function:

$$\mu_{\tilde{a}}(x) = \begin{cases} L\left(\dfrac{l-x}{\alpha}\right), & \text{if } x < l \\ 1, & \text{if } l \le x < r \\ R\left(\dfrac{x-r}{\beta}\right), & \text{if } r \le x \end{cases}$$

where α, β are called left and right spreads (or fuzziness), respectively. Then, we denote $\tilde{a} = (l, r, \alpha, \beta)_{LR}$. Also, $\tilde{a}_L = l - \alpha$ and $\tilde{a}_U = r + \beta$ are called lower and upper bounds, respectively.

If $L(x)=R(x)=\max\{0,1-x\}$, then $\tilde{a}$ is called a Trapezoidal FN. If $L(x)=R(x)=\max\{0,1-x\}$, and $l=r$, then we get a Triangular FN. We denote a triangular FN as $\tilde{a}=(c,\alpha,\beta)$, where the center $\tilde{a}_C=c$ is referred to as the most likely value or mode.

2.1. Arithmetic Operations on Fuzzy Numbers

In this section, we review some fuzzy arithmetic operations on triangular FNs. Similar definitions can be given for trapezoidal FNs. Since the sets of triangular or trapezoidal FNs are closed under these operations, we solely concentrate on these FNs in this study.

Let $\tilde{a}=(c_1,\alpha_1,\beta_1)$ and $\tilde{b}=(c_2,\alpha_2,\beta_2)$ be two triangular FNs, and $\lambda\in\mathbb{R}$, then we have:

- Addition

$$\tilde{a}+\tilde{b}=(c_1+c_2,\alpha_1+\alpha_2,\beta_1+\beta_2),$$

- Multiplication

$$\tilde{a}\times\tilde{b}=(c,\alpha,\beta), \tag{1}$$

where the center, lower and upper bounds are:

$$\left(\tilde{a}\times\tilde{b}\right)_C=c=c_1c_2,$$

$$\left(\tilde{a}\times\tilde{b}\right)_L=c-\alpha=\min\{(c_1-\alpha_1)(c_2-\alpha_2),(c_1-\alpha_1)(c_2+\beta_2),(c_1+\beta_1)(c_2-\alpha_2),(c_1+\beta_1)(c_2+\beta_2)\},$$

$$\left(\tilde{a}\times\tilde{b}\right)_U=c+\beta=\max\{(c_1-\alpha_1)(c_2-\alpha_2),(c_1-\alpha_1)(c_2+\beta_2),(c_1+\beta_1)(c_2-\alpha_2),(c_1+\beta_1)(c_2+\beta_2)\},$$

respectively.

- Scalar Multiplication

$$\lambda\tilde{a}=\begin{cases}(\lambda c_1,\lambda\alpha_1,\lambda\beta_1), & \text{if } \lambda\geq 0\\ (\lambda c_1,-\lambda\beta_1,-\lambda\alpha_1), & \text{if } \lambda<0.\end{cases}$$

2.2. Fully Fuzzy Linear Systems

We are interested in finding the approximate solutions of a square linear system whose coefficient matrix, unknowns, and the right-hand sides are all arbitrary triangular fuzzy numbers. To avoid confusion, it is worth mentioning that the fuzzy parameters involved in FFSLEs are assumed to be mutually independent.

Consider a square fuzzy coefficient matrix $\tilde{A} = (\tilde{a}_{ij})_{n\times n}$, such that $\tilde{A} = (A, M, N)$ consisting of a center $A = (a_{ij}) \in \mathbb{R}^{n\times n}$, left spread $M = (\alpha_{ij}) \geq 0$, and right spread $N = (\beta_{ij}) \geq 0$, which are crisp matrices with appropriate dimensions; and a fuzzy right-hand side vector $\tilde{b} = (\tilde{b}_i)_{n\times 1}$, represented by $\tilde{b} = (b, h, g)$ with the crisp vectors of centers $b \in \mathbb{R}^n$, and non-negative spreads $h, g \in \mathbb{R}^n$; and a fuzzy unknown vector $\tilde{x} = (\tilde{x}_j)_{n\times 1}$, such that $\tilde{x} = (x, y, z)$ with the crisp vectors of centers $x \in \mathbb{R}^n$, and non-negative spreads $y, z \in \mathbb{R}^n$. Then, $n\times n$ FFSLE is of the form:

$$\begin{aligned}
&\tilde{a}_{11}\times\tilde{x}_1 + \tilde{a}_{12}\times\tilde{x}_2 + \ldots + \tilde{a}_{1n}\times\tilde{x}_n = \tilde{b}_1 \\
&\tilde{a}_{21}\times\tilde{x}_1 + \tilde{a}_{22}\times\tilde{x}_2 + \ldots + \tilde{a}_{2n}\times\tilde{x}_n = \tilde{b}_2 \\
&\ddots \\
&\tilde{a}_{n1}\times\tilde{x}_1 + \tilde{a}_{n2}\times\tilde{x}_2 + \ldots + \tilde{a}_{nn}\times\tilde{x}_n = \tilde{b}_n.
\end{aligned} \tag{2}$$

The matrix form of System (2) is as follows:

$\tilde{A}\times\tilde{x} = \tilde{b}.$

With the assumption that each element of fuzzy matrices is non-negative and A is invertible, Dehghan et al. (2006) characterized the approximate solution as the solution of the following system:

$$\begin{cases} Ax = b, \\ Ay + Mx = g, \\ Az + Nx = h \end{cases}$$

with the help of approximate formulas for the extended multiplication operation, since the multiplication operator in Equation (1) cannot be expressed analytically.

We consider the approximate solution that is intended to minimize the following total error function:

$$error = \sum_{i=1}^{n} \sum_{r=L,U} e_{ir} \tag{3}$$

where $e_{ir} = \left| \sum_{j=1}^{n} (\tilde{a}_{ij}\times\tilde{x}_j)_r - (\tilde{b}_i)_r \right|,\ i = 1,\ldots,n,\ r = L,U.$

We must emphasize that we use the multiplication operator given in Equation (1), and aim to obtain two close (or similar) FNs on both sides of the equalities as much as possible.

3. Proposed Heuristic

Our algorithm is a two-stage approach, given in Subsections 3.1 and 3.2. In the first stage, the lower and upper bounds of the solutions are estimated, and the corresponding spread-based errors are computed. For each equation, the errors that occurred in right and left spreads are taken into consideration independently. Then, starting with the spread having the highest inaccuracy, the new value of this width is assumed to be equal to half of the previous value. Depending on whether the total error value improves or not, the original spread is narrowed to a certain extent with a prescribed number of bisection iterations. The optimal narrowing is one that minimizes the total error value. After that, moving on to the spread with the second largest error, the bisection procedure is repeated. The process is completed when the errors in all spreads are reduced. Therefore, the bisection algorithm is performed $2n$ times per system. The errors are updated in the case of calibration of intervals belonging to the solutions in any iteration.

3.1. Fuzzy Numerical Simulation

The foundation of our algorithm is based on the generation of single values of FNs. When we solve the crisp problems, obtained by substituting the generated values of the fuzzy parameters in the system of equations, we can estimate the intervals in which the solutions are located. It is obvious that this process needs to be repeated too many times in order to provide meaningful results. The literature contains a wide variety of efficient techniques for solving crisp equation systems. Therefore, we attempt to make use of these non-fuzzy techniques to provide better approximations. Any desired equation-solving method can be used in the intermediate steps of the algorithm.

Now let us review the FNS technique we will use to generate single values of parameters. To simulate a single value of a fuzzy coefficient using FNS (Chanas and Nowakowski, 1998), we first generate two independent random numbers, u and v from the uniform distribution. Considering the values of v and $(1-v)$ as weights, the convex combination or weighted arithmetic average of the endpoints of the $u-$cut interval can then be used to replicate samples of the parameters. For a triangular coefficient $\tilde{a}_{ij}=\left(a_{ij},\alpha_{ij},\beta_{ij}\right)$ in System (2), the following equation is derived to generate samples:

$$\bar{a}_{ij}=v_{ij}\left[a_{ij}-\left(1-u_{ij}\right)\alpha_{ij}\right]+\left(1-v_{ij}\right)\left[a_{ij}+\left(1-u_{ij}\right)\beta_{ij}\right],$$

where $i,j=1,\ldots,n,\forall u_{ij},v_{ij}\sim U(0,1)$ are independent.

Similarly, for a triangular right-hand side (or target) value $\tilde{b}_i=(b_i,h_i,g_i)$ in System (2),

$$\bar{b}_i = V_i\left[b_i - (1-U_i)h_i\right] + (1-V_i)\left[b_i + (1-U_i)g_i\right],$$

such that $i = 1,\dots,n,\ \forall U_i, V_i \sim U(0,1)$ and independent. A similar procedure can be applied to trapezoidal coefficients and targets.

The center values of the solutions $\tilde{x}_C$ are considered to be solutions of the 1-cut system: $Ax = b.$

When the simulated $\bar{a}_{ij}$ and $\bar{b}_i$ coefficients are plugged into a large number of systems and these systems are solved, a wide range of potential values is obtained for each unknown. For any unknown, it is regarded as a potential value for the lower (upper) bound if the potential value from the solution of the system of crisp equations is less (greater) than the center value. These corresponding potential values are first averaged, and then the supremum and infimum of these large numbers of averages are calculated for the lower and upper bounds, respectively. This part of the process is summarized in the following pseudo-code:

```
Function FNSbasedProcedureForBounds
Input: $A, M, N, b, h, g$ and large numbers $T, K$
Output: $\tilde{x}_C, \tilde{x}_L, \tilde{x}_U, e_{ir},\ i = 1,\dots,n,\ r = L,U$
(1) Solve the 1-cut system $Ax = b,$ and set $\tilde{x}_C := x$
(2) for $t = 1$ to $T$ do
(3) for $k = 1$ to $K$ do
(4) Generate $\bar{A}, \bar{b}$
(5) Solve the crisp system $\bar{A}x = \bar{b},$ and store $x_j^k$ for all $j = 1,\dots,n$
(6) end for
(7) Store the mean value of the $x_j^k$'s less (or similarly greater) than $(\tilde{x}_j)_C$ in $(\tilde{x}_j)_L^t$ (or $(\tilde{x}_j)_U^t$),
    for all $j = 1,\dots,n$
(8) end for
(9) Set $(\tilde{x}_j)_L = \max_t (\tilde{x}_j)_L^t$ and $(\tilde{x}_j)_U = \min_t (\tilde{x}_j)_U^t$ for all $j = 1,\dots,n$
(10)    For the attained solution, set $e_{ir},\ i = 1,\dots,n,\ r = L,U$ as in Equation (3)
end Function
```

This stage of our algorithm is similar to the search algorithm in (Rao and Chen, 1998). In Step 9, in the computation of the lower (similarly upper) bound of the solutions, the union of the nested $(-\infty, \tilde{x}_L)$ ($(\tilde{x}_U, \infty)$, respectively) intervals is considered.

To summarize, at this stage, the generated *K* crisp systems are solved, and the values less and greater than the center value are stored separately. The average of the obtained values is taken independently, and all these processes are repeated *T* times. The greatest of the *T* average values is the output value for the left endpoint, i.e., the lower bound of the solution interval. Similarly, the output value for the right endpoint, i.e., the upper bound, is the lowest of the averages of values larger than the center value. The errors for the approximate solution, which measure how similar the left and right extremes for the left and right sides of each row of equations are, are also noted. In the next stage, bisection operations will be performed starting from the row and the bound with the largest error.

3.2. Bisection Method

The bisection algorithm is applied to the spreads in accordance with the order in which $e_{ir},\ i=1,\dots,n,\ r=L,U$ values are listed in descending order. This ranking will be determined once at the beginning of the process and will remain the same until the end of the algorithm. A one-to-one mapping between rows and columns is considered. For instance, an update at the upper range of $\tilde{x}_j$ results from an error related to the upper range of the ith equation such that $i=j$. The following procedure is applied $(2n)-$times, since $p=1,\dots,n,\ s=L,U.$ Here, *M* is the parameter that indicates how many bisections will be done.

```
**Function** $Bisection(p,s)$
**Input:** $\tilde{x}_C,\tilde{x}_L,\tilde{x}_U,\ e_{ir},\ i=1,\dots,n,\ r=L,U$ and bisection repetitions $M$
**Output:** $error$ and $(\tilde{x}_p)_s$
(1) Initiate $h_L=0,\ h_U=1$ (If $e_{ps}$ is relatively much larger than the other error values, then
    initiate $h_L=0.5,\ h_U=1$ to accelerate the procedure)
(2)   Set $m=1,\ (\tilde{x}_p)_{s,m}^{new} := (\tilde{x}_p)_s$ and $error_m := error$ computed using Equation (3)
(3)   for $m=2$ to $M$ do
(4)     $h := (h_L+h_U)/2$
(5)     $(\tilde{x}_p)_{s,m}^{new} := h(\tilde{x}_p)_C + (1-h)(\tilde{x}_p)_s$
```

```
(6)  Update $error$ using the *new* value $\left(\tilde{x}_p\right)_{s,m}^{new}$ for $\left(\tilde{x}_p\right)_s$ and store it in $error_m$
(7)  If $error_m < error_{(m-1)}$ then
(8)      $h_L := h$
(9)      else
(10)     $h_U := h$
(11) end if
(12) end for
(13)     Return $\left(\tilde{x}_p\right)_s := \left(\tilde{x}_p\right)_{s,\omega}^{new}$ such that $error_\omega,\ \omega \in \{1,\dots,M\}$ is the minimum error i.e.,
       $error_\omega = \min_{m\in\{1,\dots,M\}} error_m$
(14)     Update $error$
  end Function
```

At this stage, bisections will be done for all rows and spreads with positive error values. Even if the bisection operations are performed a given number of times, let us say *M*, the excess bisections are undone if the error gets worse.

4. Findings and Discussion

All computational experiments were performed using MATLAB R2018a on a computer with an Intel Core i5-7400 CPU (3.00 GHz) and 4 GB of RAM running MS Windows 10 Pro. In our calculations, we assume that $T = K = 1000,\ M = 20.$ As a pseudo-random generator, we use the MATLAB function “rand”. To demonstrate the methodology proposed in this paper, we consider the following examples:

Example 4.1 Our first example is Test 3.2 from (Dehghan et al., 2006). Consider the following FFSLE of size 2×2:

$$\begin{aligned}&\left(5,1,1\right)\times(x_1,y_1,z_1)+\left(6,1,2\right)\times(x_2,y_2,z_2)=\left(50,10,17\right)\\&\left(7,1,0\right)\times(x_1,y_1,z_1)+\left(4,0,1\right)\times(x_2,y_2,z_2)=\left(48,5,7\right)\end{aligned}$$

After implementing the first stage of our algorithm, the relevant error values measuring the initial lower and upper bound dissimilarities of the left and right sides of the equations are computed as follows: $e_{1L} = 9.0889$, $e_{1U} = 14.2831$, $e_{2L} = 10.0483$, $e_{2U} = 11.5687$. Thus, in the second stage, the bisection algorithm is applied sequentially to the intervals:

$\left[(\tilde{x}_1)_C, (\tilde{x}_1)_U\right]$, $\left[(\tilde{x}_2)_C, (\tilde{x}_2)_U\right]$, $\left[(\tilde{x}_2)_L, (\tilde{x}_2)_C\right]$, and $\left[(\tilde{x}_1)_L, (\tilde{x}_1)_C\right]$,

respectively.

To give more details, before bisection operations, the first stage of the algorithm gives the initial intervals as $\left[(\tilde{x}_1)_L, (\tilde{x}_1)_U\right] = [2.9367, 4.8513]$, $\left[(\tilde{x}_2)_L, (\tilde{x}_2)_U\right] = [3.8329, 6.5219]$, and the initial error as 44.9889. The solution of the 1-cut system is $(\tilde{x}_1)_C = 4, (\tilde{x}_2)_C = 5.$ At this stage, $T \times K = 1000000$ crisp systems are solved in two nested loops, and thus several realizations are covered. Since this error value is quite large, the second stage has started in order to narrow those too-wide spreads. Refer to Table 1 for the solutions and corresponding errors after bisection operations. In this example, a system with an exact solution is considered. The method produces a solution with a lower error value compared to the error of the previously proposed approximate solution; thus, there is an improvement in the error.

Table 1. Solutions of Example 4.1

	Approximate solution reported in (Dehghan et al., 2006)	Approximate solution using proposed method	Exact solution reported in (Allahviranloo and Mikaeilvand, 2011)
(x_1, y_1, z_1)	$(4, 1/11, 0)$	$(4, 0.1661, 1.6238\text{e}-06)$	$(4, 1/14, 1/26)$
(x_2, y_2, z_2)	$(5, 1/11, 1/2)$	$(5, 2.2261\text{e}-06, 0.3805)$	$(5, 1/7, 9/26)$
error	1.7727	0.4800	0.0000

Example 4.2 Our second example is Test 4.1 from (Dehghan et al., 2006). Consider the following FFSLE of size 3×3:

$$(4,3,2)\times(x_1,y_1,z_1)+(5,2,1)\times(x_2,y_2,z_2)+(3,0,3)\times(x_3,y_3,z_3)=(71,54,76)$$
$$(7,4,3)\times(x_1,y_1,z_1)+(10,6,5)\times(x_2,y_2,z_2)+(2,1,1)\times(x_3,y_3,z_3)=(118,115,129)$$
$$(6,2,2)\times(x_1,y_1,z_1)+(7,1,2)\times(x_2,y_2,z_2)+(15,5,4)\times(x_3,y_3,z_3)=(155,89,151)$$

Refer to Table 2 for the solutions and corresponding errors. This example takes into account a different-sized system with no exact solution, and it is observed that the error decreased again.

Table 2. Solutions of Example 4.2

	Approximate solution reported in (Dehghan et al., 2006)	Approximate solution using proposed method	Exact solution
(x_1, y_1, z_1)	$(4, 2, 2)$	$(4, 7.3355, 7.4337)$	N/A
(x_2, y_2, z_2)	$(8, 3, 5)$	$(8, 4.1054\text{e}-05, 3.9242\text{e}-05)$	
(x_3, y_3, z_3)	$(5, 1, 4)$	$(5, 6.4508\text{e}-06, 6.7129\text{e}-06)$	
error	137	58.2116	

Example 4.3 Our next example is Example 4.9 from (Kumar et al., 2013). Consider the following FFSLE of size 2×2:

$$\left(3,2,3\right)\times(x_1,y_1,z_1)+\left(-2,1,1\right)\times(x_2,y_2,z_2)=\left(5,16,17\right)$$
$$\left(-4,1,2\right)\times(x_1,y_1,z_1)+\left(4,2,1\right)\times(x_2,y_2,z_2)=\left(-4,12,22\right)$$

Refer to Table 3 for the solutions and corresponding errors. A small error value is obtained for this case with negative coefficients.

Table 3. Solutions of Example 4.3

	Approximate solution using proposed method	Exact solution reported in (Kumar et al., 2013)
(x_1,y_1,z_1)	$(3,3.9123,0.9185)$	$(3,2,1)$
(x_2,y_2,z_2)	$(2,1.7025e-05,2.1115e-05)$	$(2,0,2)$
error	4.8084	0.0000

Example 4.4 Our next example is Example 4.17 from (Kumar et al., 2013). Consider the following FFSLE of size 2×2:

$$\left(2,0,1\right)\times(x_1,y_1,z_1)+\left(2,0,2\right)\times(x_2,y_2,z_2)=\left(6,5,9\right)$$
$$\left(1,0,1\right)\times(x_1,y_1,z_1)+\left(2,1,1\right)\times(x_2,y_2,z_2)=\left(5,5,6\right)$$

Refer to Table 4 for the solutions and corresponding errors. Although the method does not produce exact solutions, it is effective in generating approximate solutions with a relatively small error.

Table 4. Solutions of Example 4.4

	Approximate solution using proposed method	Exact solution reported in (Kumar et al., 2013)
(x_1,y_1,z_1)	$(1,1.2669e-05,1.2493e-05)$	$(1,2,0)$
(x_2,y_2,z_2)	$(2,2.4874,0.9742)$	$(2,0,1)$
error	1.5928	0.0000

Example 4.5 Our next example is Example 18 from (Allahviranloo et al., 2013). Consider the following FFSLE of size 2×2:

$$\left(3,2,2\right)\times(x_1,y_1,z_1)+\left(2,1,0\right)\times(x_2,y_2,z_2)=\left(8,3,2\right)$$
$$\left(1,0,2\right)\times(x_1,y_1,z_1)+\left(4,1,1\right)\times(x_2,y_2,z_2)=\left(6,4,6\right)$$

Refer to Table 5 for the solutions and corresponding errors. It has also been observed that the error is improved compared to existing methods.

Table 5. Solutions of Example 4.5

	Approximate solution using proposed method	Approximate solution reported in (Allahviranloo et al., 2013)	Alternative approximate solution reported in (Allahviranloo et al., 2013)
(x_1, y_1, z_1)	$(2, 0.4346, 1.5731e-06)$	$(2, 2/5, 1/8)$	$(2, 1, 2/5)$
(x_2, y_2, z_2)	$(1, 0.6399, 0.2102)$	$(1, 2/5, 1/8)$	$(1, 1, 3/11)$
error	6.1915	7.0750	11.1091

Example 4.6 Our last example is Test 4.1 from (Dehghan and Hashemi, 2006). Consider the FFSLE of size 3×3 with the following matrices:

$$A=\begin{bmatrix}19 & 12 & 6\\ 2 & 4 & 1.5\\ 2 & 2 & 4.5\end{bmatrix},\ b=\begin{bmatrix}1897 & 434.5 & 535.5\end{bmatrix}^T,$$

$$M=\begin{bmatrix}1 & 1.5 & 0.5\\ 0.1 & 0.1 & 0.2\\ 0.1 & 0.1 & 0.1\end{bmatrix},\ h=\begin{bmatrix}427.7 & 76.2 & 88.3\end{bmatrix}^T,$$

$$N=\begin{bmatrix}1 & 1.5 & 0.2\\ 0.1 & 0.4 & 0.2\\ 0.2 & 0.3 & 0.1\end{bmatrix},\ g=\begin{bmatrix}526.2 & 109.3 & 131.9\end{bmatrix}^T.$$

Refer to Table 6 for the solutions and corresponding errors. For this problem, we take the first h_L as 0.5 to speed up the process. In this case, there is an improvement in the error.

Table 6. Solutions of Example 4.6

	Approximate solution reported in (Dehghan and Hashemi, 2006)	Approximate solution reported in (Ezzati et al., 2014)	Approximate solution using proposed method
(x_1, y_1, z_1)	$(37.0000, 7.0000, 13.3016)$	$(37.9237, 10.1207, 10.1207)$	$(37.0000, 9.6825, 10.8616)$
(x_2, y_2, z_2)	$(62.0000, 5.5000, 4.5794)$	$(61.8675, 5.0181, 5.0181)$	$(62.0000, 4.4591, 7.5356)$
(x_3, y_3, z_3)	$(75.0000, 10.2000, 13.9196)$	$(75.5411, 12.0458, 12.0458)$	$(75, 7.0935, 8.4877)$
error	70.2364	52.4729	51.5282

5. Conclusions and Recommendations

We present an approach to compute approximate solutions of FFSLEs, where components are triangular FNs. The proposed numerical method appears as a strong alternative when exact solutions of large dimension systems of equations whose parameters cannot be given as precise numbers cannot be obtained. This method first solves a large number of possible equations and then obtains very large ranges of solutions. After that, it tries to reduce the error by narrowing these ranges. The algorithm is implemented and successfully tested by solving numerous benchmark problems. At first glance, it appears that our algorithm reduces the total error of the approximate solutions obtained with the existing techniques. Also, we can safely assume that a large total error value will occur with our algorithm when FFSLEs have no solution.

Under certain conditions, the proposed methodology can be extended to intuitionistic systems by applying it to two separate systems. Additional application areas include dual, non-linear, and matrix equation systems. Extensions to the other cases are reserved for future research.

Acknowledgements

The authors would like to express their sincere gratitude to the editor and the anonymous reviewers for their insightful comments and recommendations.

Authors’ Contributions

The author confirms sole responsibility for the following: study conception and design, data collection, analysis and interpretation of results, and manuscript preparation.

Statement of Conflicts of Interest

There is no conflict of interest to disclose.

Statement of Research and Publication Ethics

The author declares that this study complies with Research and Publication Ethics.

References

Ahlatcioglu, M., Albayrak, I., Kocken, H. G., and Ozkok, B. A. (2016). A mixed integer programming approach to a square fully fuzzy linear equation. *Journal of Intelligent and Fuzzy Systems*, 31(3), 2009-2015. https://doi.org/10.3233/JIFS-16227

Akdemir, H. G. (2023). Approximate Fuzzy Inverse Matrix Calculation Method using Scenario-based Inverses and Bisection. *Fundamental Journal of Mathematics and Applications*, 6(1), 42-50. https://doi.org/10.33401/fujma.1195121

Akdemir, H. G., and Kocken, H. G. (2022). A new fuzzy linear regression algorithm based on the simulation of fuzzy samples and an application on popularity prediction of Covid-19 related videos. *Journal of Statistics and Management Systems*, 1-17. https://doi.org/10.1080/09720510.2021.2016988

Albayrak, I. (2017). On fuzzy solutions of the nonsquare fully fuzzy linear equation system with arbitrary triangular fuzzy numbers. *Journal of Intelligent and Fuzzy Systems*, 33(6), 3929-3938. https://doi.org/10.3233/JIFS-17774

Allahviranloo, T., Hosseinzadeh, A. A., Ghanbari, M., Haghi, E., and Nuraei, R. (2014). On the new solutions for a fully fuzzy linear system. *Soft Computing*, 18, 95-107. https://doi.org/10.1007/s00500-013-1037-3

Allahviranloo, T., Kiani, N. A., Barkhordary, M., and Mosleh, M. (2008). Homomorphic solution of fully fuzzy linear systems. *Computational Mathematics and Modeling*, 19, 282-291. https://doi.org/10.1007/s10598-008-9004-z

Allahviranloo, T., and Mikaeilvand N. (2011). Non zero solutions of the fully fuzzy linear systems. *Applied and Computational Mathematics*, 10(2), 271-282.

Allahviranloo, T., Salahshour, S., Homayoun-Nejad, M., and Baleanu, D. (2013, January). General solutions of fully fuzzy linear systems. In *Abstract and Applied Analysis* (Vol. 2013). Hindawi. https://doi.org/10.1155/2013/593274

Allahviranloo, T., Salahshour, S., and Khezerloo, M. (2011). Maximal-and minimal symmetric solutions of fully fuzzy linear systems. *Journal of Computational and Applied Mathematics*, 235(16), 4652-4662. https://doi.org/10.1016/j.cam.2010.05.009

Babbar, N., Kumar, A., and Bansal, A. (2013a). Linear programming approach to find the solution of fully fuzzy linear systems with arbitrary fuzzy coefficients. *Journal of Intelligent and Fuzzy Systems*, 25(3), 747-753. https://doi.org/10.3233/IFS-120681

Babbar, N., Kumar, A., and Bansal, A. (2013b). Solving fully fuzzy linear system with arbitrary triangular fuzzy numbers. *Soft Computing*, 17(4), 691-702. https://doi.org/10.1007/s00500-012-0941-2

Behera, D., and Chakraverty, S. (2017). A note on "A new method for solving an arbitrary fully fuzzy linear system". *Soft Computing*, 21, 7117-7118. https://doi.org/10.1007/s00500-016-2254-3

Chanas, S., and Nowakowski, M. (1988). Single value simulation of fuzzy variable. *Fuzzy Sets and Systems*, 25(1), 43-57. https://doi.org/10.1016/0165-0114(88)90098-X

Dehghan, M., and Hashemi, B. (2006). Solution of the fully fuzzy linear systems using the decomposition procedure. *Applied Mathematics and Computation*, 182(2), 1568-1580. https://doi.org/10.1016/j.amc.2006.05.043

Dehghan, M., Hashemi, B., and Ghatee, M. (2006). Computational methods for solving fully fuzzy linear systems. *Applied Mathematics and Computation*, 179(1), 328-343. https://doi.org/10.1016/j.amc.2005.11.124

Dehghan, M., Hashemi, B., and Ghatee, M. (2007). Solution of the fully fuzzy linear systems using iterative techniques. *Chaos, Solitons and Fractals*, 34(2), 316-336. https://doi.org/10.1016/j.chaos.2006.03.085

Ezzati, R., Khezerloo, S., Mahdavi-Amiri, N., and Valizadeh, Z. (2014). Approximate Nonnegative Symmetric Solution of Fully Fuzzy Systems Using Median Interval Defuzzification. *Fuzzy Information and Engineering*, 6(3), 331-358. https://doi.org/10.1016/j.fiae.2014.12.005

Ezzati, R., Khezerloo, S., Valizadeh, Z., and Mahdavi-Amiri, N. (2012). New models and algorithms for solutions of single-signed fully fuzzy LR linear systems. *Iranian Journal of Fuzzy Systems*, 9(3), 1-26.

Guo, X., Wei, Y., and Li, Z. (2018). Further investigation to approximate fuzzy inverse. *Journal of Intelligent and Fuzzy Systems*, 35(1), 1161-1168. https://doi.org/10.3233/JIFS-18027

Jeswal, S. K., and Chakraverty, S. (2019). Connectionist model for solving static structural problems with fuzzy parameters. *Applied Soft Computing*, 78, 221-229. https://doi.org/10.1016/j.asoc.2019.02.025

Kocken, H. G., Ahlatcioglu, M., and Albayrak, I. (2016). Finding the fuzzy solutions of a general fully fuzzy linear equation system. *Journal of Intelligent and Fuzzy Systems*, 30(2), 921-933. https://doi.org/10.3233/IFS-151813

Kumar, A., Bansal, A., and Babbar, N. (2013). Fully fuzzy linear systems of triangular fuzzy numbers (a, b, c). *International Journal of Intelligent Computing and Cybernetics*, 6(1), 21-44. https://doi.org/10.1108/17563781311301508

Kumar, A., Neetu, and Bansal, A. (2012). A new computational method for solving fully fuzzy linear systems of triangular fuzzy numbers. *Fuzzy Information and Engineering*, 4(1), 63-73. https://doi.org/10.1007/s12543-012-0101-5

Moloudzadeh, S., Allahviranloo, T., and Darabi, P. (2013). A new method for solving an arbitrary fully fuzzy linear system. *Soft Computing*, 17(9), 1725-1731. https://doi.org/10.1007/s00500-013-0986-x

Mosleh, M. (2013). Evaluation of fully fuzzy matrix equations by fuzzy neural network. *Applied Mathematical Modelling*, 37(9), 6364-6376. https://doi.org/10.1016/j.apm.2013.01.011

Mosleh, M., and Otadi, M. (2015). A discussion on "Calculating fuzzy inverse matrix using fuzzy linear equation system". *Applied Soft Computing*, 28, 511-513. https://doi.org/10.1016/j.asoc.2014.11.035

Otadi, M., and Mosleh, M. (2012). Solving fully fuzzy matrix equations. *Applied Mathematical Modelling*, 36(12), 6114-6121. https://doi.org/10.1016/j.apm.2012.02.005

Otadi, M., Mosleh, M., and Abbasbandy, S. (2011). Numerical solution of fully fuzzy linear systems by fuzzy neural network. *Soft Computing*, 15(8), 1513-1522. https://doi.org/10.1007/s00500-010-0685-9

Rao, S. S., and Chen, L. (1998). Numerical solution of fuzzy linear equations in engineering analysis. *International Journal for Numerical Methods in Engineering*, 42(5), 829-846. https://doi.org/10.1002/(SICI)1097-0207(19980715)42:5%3C829::AID-NME386%3E3.0.CO;2-G

Ziqan, A., Ibrahim, S., Marabeh, M., and Qarariyah, A. (2022). Fully fuzzy linear systems with trapezoidal and hexagonal fuzzy numbers. *Granular Computing*, 7(2), 229-238. https://doi.org/10.1007/s41066-021-00262-6