Scientific Publications

Universal Journal of Computer Sciences and Communications, 2025, 4, 6144
www.scipublications.com/journal/index.php/ujcsc
DOI: 10.31586/ujcsc.2025.6144

Article

A Convergence of the Muller's Sequence

Rostyslav V. Bilous [1], Ivan H. Krykun [2,3,*]

[1] Company Onseo, Vinnytsia, Ukraine

[2] Department of Theory of Control Systems, Institute of Applied Mathematics and Mechanics, NAS of Ukraine, Sloviansk, Ukraine

[3] Department of Information Management, Mathematics and Statistics, Educational and Scientific Institute for Information and Communication Technologies, «KROK» University, Kyiv, Ukraine

*Correspondence: Ivan H. Krykun (iwanko@i.ua)

Abstract: In this paper, we will examine a rather complex case of the paradoxical nature of certain conclusions that may arise when studying the numerical convergence of a specific nonlinear recursive sequence, known in the literature as Muller's sequence. To analyze the peculiar computational behavior of this sequence, it is necessary to employ a powerful mathematical framework in order to understand the nontrivial issues that can arise when the software implementation of this seemingly simple mathematical problem. These challenges often stem from the limitations of numerical methods and the inherent errors in computer arithmetic, which can affect the accuracy and stability of the results, particularly when dealing with iterative methods like Muller's sequence.

Keywords: Muller's Sequence; Computer Calculations; Programming

1. Introduction

The reliability and accuracy of computer calculations represent a linchpin in computational mathematics and computer science, guiding the design, stability, and trustworthiness of algorithms used to solve scientific, engineering, and practical problems. This issue becomes acutely pronounced in the realm of numerical methods, particularly those aimed at solving nonlinear equations—such as Muller's method, an iterative approach valued for its rapid convergence characteristics.

At the heart of computational accuracy lies the architecture of floating-point arithmetic. Computer systems represent real numbers in finite precision, which inevitably introduces rounding errors, local truncation errors, and in larger systems, error propagation across iterations or computations. The recursive summation of n floating-point numbers, for instance, is highly sensitive to the ordering and method of summation, with some approaches minimizing the accumulation of rounding errors more effectively than others. Higham [1] systematically compared multiple summation strategies, emphasizing that while no single technique is universally dominant, judicious selection of the summation algorithm, tailored to the problem context, can mitigate error accumulation and thus improve reliability of calculations.

The characteristics of floating-point arithmetic, including base choice (binary, hexadecimal, etc.), define both static and dynamic error properties. As Cody's systematic assessment suggested, design choices at the level of machine representation ripple outward to affect the accuracy of every numerical process thereafter [2].

Muller's method is a prototypical iterative root-finding algorithm that utilizes quadratic interpolation. By fitting a parabola to three points and extrapolating its root, Muller's method often surpasses classical techniques like the secant or Newton-Raphson methods in convergence rate, specifically near complex roots. The broader landscape of

How to cite this paper:
Bilous, R. V., & Krykun, I. H. (2025). A Convergence of the Muller's Sequence. Universal Journal of Computer Sciences and Communications, 4(1), 20–28.
DOI: 10.31586/ujcsc.2025.6144

Received: July 11, 2025
Revised: August 28, 2025
Accepted: September 21, 2025
Published: September 24, 2025

Copyright: © 2025 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).

root-finding is rich with memory-based, hybrid, and derivative-free schemes, many of which leverage previous iterates or finite-difference approximations to improve both efficiency and robustness [3-7]. For example, Qureshi et al. introduced memory-based algorithms that thoughtfully incorporate historical information to expedite local and semilocal convergence while controlling computational expense [3].

Iterative root-finding algorithms such as Muller's share a common vulnerability: error amplification through repeated application in floating-point environments. Rounding errors, truncation errors, and inaccuracies in evaluating derivatives or function values can accumulate or even amplify from step to step, especially in ill-conditioned or stiff systems [6,8].

Numerical methods for root-finding are assessed not only by their order of convergence but also by their stability and sensitivity to input errors. Recent literature employs theoretical convergence analyses, symbolic computation, and computational experiments to map the stability landscapes of these algorithms, identifying regions in the complex plane where convergence is assured or erratic [6]. Basin of attraction visualizations—wherein each point in the plane represents an initial guess and is colored by the corresponding convergence outcome—offer graphical and intuitive metrics for the reliability and predictability of iterative schemes [6,7].

Accuracy can be further scrutinized by means of verified computations, where rigorous error bounds are established for the computed solutions, using interval arithmetic or other guarantees [9]. Techniques such as iterative refinement serve as another line of defense against inaccuracy; for example, one may apply Newton's method repeatedly to refine an approximate eigenpair until the forward and backward errors fall within desired tolerances, even in the presence of floating-point inaccuracies [8].

The imperative for reliable computation scales with problem size and physical consequence. Modern engineering, physics, and data-driven science demand iterative algorithms capable of not only efficiently finding roots, but doing so with documented error margins—even as problem dimensionality expands by orders of magnitude [4,7,9,10]. In nuclear physics mean-field calculations or large-scale linear systems, as in Ryssens et al. and Ozaki et al., both the method of discretization and the granularity of finite elements or mesh points must be carefully tuned for dependable results [9,10]. As such, accuracy-guaranteed approaches become indispensable, especially when algorithmic stability and model fidelity are paramount.

2. Muller's sequence.

So the problem of reliability (accuracy) of computer calculations is one of the fundamental problems of computer science [11,12]. Muller's sequence [13] and the accuracy of computer calculations are fundamental concepts in the study of numerical methods. Muller's method, an iterative approach for finding the roots of nonlinear equations, has seen wide application in computational mathematics due to its ability to converge faster than other methods, like Newton-Raphson.

However, the accuracy of these methods is heavily influenced by the precision of the computational environment and the handling of floating-point errors. As computational technology advances, understanding how errors propagate and affect calculations is crucial for maintaining reliability in scientific and engineering applications.

At the same time, the problem is not only the possibility of an incorrect result of theoretical research. As practice shows, the consequences of incorrect computer calculations can be catastrophic. There is a famous case: on February 25, 1991, during the Gulf War I (Operation Desert Storm), a "Patriot" air defense system failed to intercept an enemy missile, and the projectile hit a barracks of US soldiers, killing 28 people. An official investigation [14] showed that 24-bit interceptor processors make a time conversion error of 0.013 seconds every hour. "Patriot" was not restarted for more than 100 hours, and as

a result of the accumulation of such seemingly minor errors, during this time there was an error in calculating the position of the missile at 600 meters.

Some other examples of real losses caused by rounding errors or problems in representing numbers in computer memory are given in [15].

In paper [16], the authors considered an example of incorrect program behaviour, namely Rump's example. In addition to mathematical explanations, they focused on software-based methods for controlling the accuracy of complex computations.

This paper investigates the implications of computational accuracy, focusing on Muller's sequence and its limitations in the context of modern computer calculations.

2.1. Problem statement

We consider a rather complex case of the paradoxical nature of some conclusions that we will have when studying the numerical convergence of a certain nonlinear recurrent sequence, known in the literature as the Muller's sequence.

A rigorous mathematical apparatus is required a powerful mathematical apparatus to understand those non-trivial problems that may arise during the software implementation of a rather simple at first glance mathematical problem.

Let's consider the mathematical object that was given in the fundamental work [13] devoted to the problem of applied computer calculations. This is a second-order nonlinear inhomogeneous recurrent equation, which we will call Muller's sequence:

$$\begin{cases} u(n) = 111 - \frac{1130}{u(n-1)} + \frac{3000}{u(n-1)\cdot u(n-2)}, \\ u(0) = 2,\ u(1) = -4. \end{cases} \tag{1}$$

Let's try to perform numerical calculations to visualize the convergence, rather than merely stating that the point 100 for sequence (1) is stable, and the point 6 is unstable.

We implement a simple Python program:

```
u0 = 2
u1 = -4
for n  in range(2, 31):
    un = 111 - 1130 / u1 + 3000 / u0 / u1
    u0 = u1
    u1 = un
    print("%.10f" % un)
```

Figure 1. Python implementation of the Muller's sequence

The results of this program for the first 30 values of the Muller's sequence are displayed in Table 1 in the "Python" column.

Table 1. The mathematical and Python program's results for first 30 values of the Muller's sequence

n	Exact value	Approximate value	Python
1	2	2.0000000000	2.0000000000
2	-4	-4.0000000000	-4.0000000000
3	37/2	18.5000000000	18.5000000000
4	347 / 37	9.3783783784	9.3783783784
5	2707 / 347	7.8011527378	7.8011527378
6	19367 / 2707	7.1544144810	7.1544144810

7	131827 / 19367	6.8067847369	6.8067847369
8	869087 / 131827	6.5926327687	6.5926327687
9	5605147 / 869087	6.4494659338	6.4494659339
10	35584007 / 5605147	6.3484520567	6.3484520587
11	223269667 / 35584007	6.2744385982	6.2744386301
12	1388446127 / 223269667	6.2186957398	6.2186962479
13	8574817387 / 1388446127	6.1758373049	6.1758454797
14	52669607447 / 8574817387	6.1423590812	6.1424914958
15	322121160307 / 52669607447	6.1158830666	6.1180393880
16	1963244539967 / 322121160307	6.0947394393	6.1299926795
17	11932055130427 / 1963244539967	6.0777223048	6.6529208479
18	72355270235687 / 11932055130427	6.0639403225	14.7110136879
19	437946318679747 / 72355270235687	6.0527217610	64.8393305454
20	2646751398406607 / 437946318679747	6.0435521102	96.7174472768
21	15975875822080267 / 2646751398406607	6.0360318811	99.7948677633
22	96332092090684727 / 15975875822080267	6.0298473250	99.9875918848
23	580376738335123987 / 96332092090684727	6.0247496524	99.9992516762
24	3494181358965822047 / 580376738335123987	6.0205399841	99.9999549130
25	21024692798570322907 / 3494181358965822047	6.0170582573	99.9999972854
26	126446180015298890567 / 21024692798570322907	6.0141749146	99.9999998367
27	760167196211178109027 / 126446180015298890567	6.0117845879	99.9999999902
28	4568453757863992482287 / 760167196211178109027	6.0098012393	99.9999999994
29	27447975450168574034347 / 4568453757863992482287	6.0081543789	100.0000000000
30	164874117215934539909207 / 27447975450168574034347	6.0067860930	100.0000000000

The second and third columns of Table 1 show the control results of mathematical calculations in the form of ordinary fractions (the third column is their rational approximation with accuracy of 10 decimal places accepted in the program).

Analyzing the data in Table 1, it can be concluded that up to the 13th term of the sequence, the mathematical and computational results correlate well, and starting from the 14th term, they gradually diverge in different directions: the analytical results with increasing n will go towards the value 6, while the software calculations are also at increased n will freeze at 100.

The Muller's sequence is a kind of unique object because the following original aspects of software calculations take place:

1) The indicated sequence (1), depending on the initial values, can both diverge and converge to one of three values: 5, 6, and 100. Therefore, the incorrect convergence of (1) at the given initial values will not cause suspicion, and it will seem that we got a quite acceptable result.

2) Incorrect convergence to the number 100 is observed not for any initial values, but only in the presence of a specific dependence

$$u(1) = 11 - \frac{30}{u(0)}$$

(and $u(0) = 2,\ u(1) = -4$ just such a case).

3) If we stopped our software calculations at steps 7-16, we would get a more or less correct result with a corresponding relative error in the worst case around 10%. This is too

much, but it is much better than the situation that arises when $n > 16$. An interesting paradox arises: the more terms of the sequence we take into account, the less accurate the result becomes. On the one hand, this is a classic problem of a stable-unstable point in the process of convergence of calculations. But on the other hand, the nature of this instability still remains unknown.

For convenience in the subsequent transformations, we introduce the following vector notations.

Relatively few scientific sources are devoted to the problem of Muller's sequence paradox. The author of [17] deals with various aspects of the issue, but he has not come across an exhaustive study of the causes of computational instability of sequence (1) and similar recurrent sequences in more general forms.

2.2. Mathematical investigation of the problem

Such a case of computational instability warrants further mathematical investigation. We apply an algebraic approach to derive a closed-form expression for the solution of sequence (1). Let us consider the expression

$$\begin{cases} u(n+2) = A + \frac{B}{u(n+1)} + \frac{C}{u(n+1)\cdot u(n)}, \\ u(0) = x,\ u(1) = y. \end{cases} \tag{2}$$

For convenience, we introduce the numbers z_1, z_2, z_3, $|z_1| < |z_2| < |z_3|$ which satisfy the following relations

$$\begin{cases} A = z_1 + z_2 + z_3, \\ B = -z_1 \cdot z_2 - z_1 \cdot z_3 - z_2 \cdot z_3, \\ C = z_1 \cdot z_2 \cdot z_3. \end{cases} \tag{3}$$

Accordingly, z_1, z_2, z_3 are the roots of such characteristic equation

$$z^3 - Az^2 - Bz - C = 0.$$

Let's denote

$$Q_n = \frac{z_3^{n+2}}{(z_3 - z_2)(z_3 - z_1)(z_2 - z_1)} \times$$

$$\times \left[x \left(y \left(z_2 - z_1 - \left(\frac{z_2}{z_3}\right)^{n+2} \cdot (z_3 - z_1) + \left(\frac{z_1}{z_3}\right)^{n+2} \cdot (z_3 - z_2) \right) - \right.$$

$$\left. - \left(z_2^2 - z_1^2 - \left(\frac{z_2}{z_3}\right)^{n+2} \cdot (z_3^2 - z_1^2) + \left(\frac{z_1}{z_3}\right)^{n+2} \cdot (z_3^2 - z_2^2) \right) \right) +$$

$$+ z_2 z_1 (z_2 - z_1) - \left(\frac{z_2}{z_3}\right)^{n+2} \cdot z_1 z_3 (z_3 - z_1) + \left(\frac{z_1}{z_3}\right)^{n+2} \cdot z_2 z_3 (z_3 - z_2) \Big].$$

For convenience in the subsequent transformations, we introduce the following vector notations

$$\begin{cases} \overline{a^{(n)}} = \left(z_2 - z_1;\ -\left(\frac{z_2}{z_3}\right)^{n+2} \cdot (z_3 - z_1);\ \left(\frac{z_1}{z_3}\right)^{n+2} \cdot (z_3 - z_2) \right), \\ \overline{\lambda_1} = (1; 1; 1), \\ \overline{\lambda_2} = (z_1 + z_2;\ z_1 + z_3;\ z_2 + z_3), \\ \overline{\lambda_2} = (z_1 z_2;\ z_1 z_3;\ z_2 z_3). \end{cases} \tag{4}$$

We can then write

$$Q_n = \frac{z_3^{n+2}}{(z_3-z_2)(z_3-z_1)(z_2-z_1)} \cdot \langle \overline{a^{(n)}}, xy\overline{\lambda_1} - x\overline{\lambda_2} + \overline{\lambda_3} \rangle. \tag{5}$$

where $\langle \overline{a}, \overline{b} \rangle$ denotes the scalar product of vectors $\overline{a}$ and $\overline{b}$.

The main result of this paper is the following theorem.

Theorem 1. Adopting notations (4) and (5), the closed-form solution to system (2) is expressed in the following system.

$$\begin{cases} u(0) = x, u(1) = y, \\ u(k+2) = \frac{Q_{k+1}}{Q_k}, \ \ k = \overline{0, \infty} \end{cases} \tag{6}$$

Proof of Theorem 1. The proof will be conducted using mathematical induction.

We will verify if (6) is satisfied for $k = 0$. We obtain the following expression

$$u(2) = \frac{Q_1}{Q_0} = \frac{\frac{z_3^3}{(z_3-z_2)(z_3-z_1)(z_2-z_1)} \cdot \langle \overline{a^{(1)}}, xy\overline{\lambda_1} - x\overline{\lambda_2} + \overline{\lambda_3} \rangle}{\frac{z_3^2}{(z_3-z_2)(z_3-z_1)(z_2-z_1)} \cdot \langle \overline{a^{(0)}}, xy\overline{\lambda_1} - x\overline{\lambda_2} + \overline{\lambda_3} \rangle}.$$

After transforming the expressions in the numerator and denominator and performing cancellation, we obtain

$$u(2) = \frac{xy(z_1 + z_2 + z_3) + x(z_1 \cdot z_2 + z_1 \cdot z_3 + z_2 \cdot z_3) + z_1 \cdot z_2 \cdot z_3}{xy} =$$

$$= A + \frac{B}{u(1)} + \frac{C}{u(1)u(0)}.$$

So (2) is fulfilled and the theorem is satisfied for $k = 0$.

Assume the statement is true for $k = n$, i.e.

$$u(n+2) = \frac{Q_{n+1}}{Q_n}.$$

And now, consider the case $k = n + 1$, which gives the following expression

$$u(n+3) = A + \frac{B}{u(n+2)} + \frac{C}{u(n+2) \cdot u(n+1)} =$$

$$= \frac{A \cdot u(n+2) \cdot u(n+1) + B \cdot u(n+1) + C}{u(n+2) \cdot u(n+1)} =$$

$$= \frac{A \cdot \frac{Q_n}{Q_{n-1}} \cdot \frac{Q_{n+1}}{Q_n} + B \cdot \frac{Q_n}{Q_{n-1}} + C}{\frac{Q_n}{Q_{n-1}} \cdot \frac{Q_{n+1}}{Q_n}} =$$

$$= \frac{A \cdot Q_{n+1} + B \cdot Q_n + C \cdot Q_{n-1}}{Q_{n+1}}.$$

Taking into account notation (3), we can write

$$A \cdot Q_{n+1} + B \cdot Q_n + C \cdot Q_{n-1} =$$

$$= (z_1 + z_2 + z_3) \cdot Q_{n+1} - (z_1z_2 + z_1z_3 + z_2z_3) \cdot Q_n + z_1z_2z_3 \cdot Q_{n-1} =$$

$$= \frac{z_3^{n+2}}{(z_3-z_2)(z_3-z_1)(z_2-z_1)} \times \Big[z_3 \cdot (z_1 + z_2 + z_3) \cdot \langle \overline{a^{(n+1)}}, xy\overline{\lambda_1} - x\overline{\lambda_2} + \overline{\lambda_3} \rangle -$$

$$-(z_1z_2 + z_1z_3 + z_2z_3) \cdot \langle \overline{a^{(n)}}, xy\overline{\lambda_1} - x\overline{\lambda_2} + \overline{\lambda_3} \rangle + z_1z_2z_3 \cdot \langle \overline{a^{(n-1)}}, xy\overline{\lambda_1} - x\overline{\lambda_2} + \overline{\lambda_3} \rangle \Big] =$$

$$= \frac{z_3^{n+2}}{(z_3 - z_2)(z_3 - z_1)(z_2 - z_1)} \times$$
$$\times \left[z_3 \cdot (z_1 + z_2 + z_3) \cdot \overline{a^{(n+1)}} - (z_1 z_2 + z_1 z_3 + z_2 z_3) \cdot \overline{a^{(n)}} + z_1 z_2 z_3 \cdot \overline{a^{(n-1)}} \right].$$

By applying such simple identity

$$z_3 \cdot (z_1 + z_2 + z_3) \cdot \left(\frac{z_i}{z_3}\right)^2 - (z_1 z_2 + z_1 z_3 + z_2 z_3) \cdot \left(\frac{z_i}{z_3}\right) + z_1 z_2 z_3 = \frac{z_i^3}{z_3}, \ \ i = \overline{1,3}$$

we get

$$Q_{n+2} = (z_1 + z_2 + z_3) \cdot Q_{n+1} - (z_1 z_2 + z_1 z_3 + z_2 z_3) \cdot Q_n + z_1 z_2 z_3 \cdot Q_{n-1} =$$
$$= A \cdot Q_{n+1} + B \cdot Q_n + C \cdot Q_{n-1}.$$

Thus, we conclude that

$$u(n+3) = \frac{Q_{n+2}}{Q_{n+1}}.$$

This completes the proof.

Further since we have established the validity of Theorem 1 and since the following statement

$$\lim_{n\to\infty} \frac{Q_n}{z_3^{n+2}} = \frac{xy - x(z_2 + z_1) + z_2 z_1}{(z_3 - z_2)(z_3 - z_1)}$$

is fulfilled, we can write that if $y \neq (z_2 + z_1) - \frac{z_2 z_1}{x}$ then

$$\lim_{n\to\infty} \frac{Q_{n+1}}{Q_n} = \lim_{n\to\infty} u(n+2) = z_3.$$

Thus, we have obtained the sought value of the initial value

$$y = (z_2 + z_1) - \frac{z_2 z_1}{x},$$

which disrupted the correctness of the calculations. By direct substitution of the specified value into Q_n, we get

$$u(n+2) = z_2 \cdot \left(\frac{z_1}{z_2} + \frac{\left(\frac{z_1}{z_2} - 1\right)(x - z_1)}{\left(\frac{z_1}{z_2}\right)^{n+1}(x - z_2) - x + z_1} \right),$$

therefore if $y = (z_2 + z_1) - \frac{z_2 z_1}{x}$ then we get

$$\lim_{n\to\infty} u(n+2) = z_2.$$

Hence, we can generalize the convergence outcome of the sequence (2).

Corollary 1. *Let's consider given the sequence (2) and the numbers* z_1, z_2, z_3, $|z_1| < |z_2| < |z_3|$. *The sequence (2) is convergent, with the following possible limit values:*

$$\begin{gathered} 1.\ If\ \ u(0) = u(1) = z_1, \quad then\ \ \lim_{n\to\infty} u(n) = z_1; \\ 2.\ If\ \ u(1) = z_1 + z_2 - \frac{z_1 z_2}{u(0)}, \ \ then\ \lim_{n\to\infty} u(n) = z_2; \\ 3.\ If\ \ u(0) = x, u(1) = y\ \ and\ y \neq z_1 + z_2 - \frac{z_1 z_2}{u(0)}, \ then\ \lim_{n\to\infty} u(n) = z_3. \end{gathered} \tag{7}$$

So now we can analyse Muller's sequence (1). We can note that the numbers $z_1 = 5$, $z_2 = 6$, $z_3 = 10$, are roots of the equation

$$z^3 - 111z^2 + 1130z - 30000 = 0$$

and given initial values $u(0) = 2$, $u(1) = -4$ correspond to the second case (7). Thus the sequence (1) should converge to the value $z_2 = 6$.

However, due to the known characteristics of computational processes [16,18] we understand that due to the nature of floating-point errors, the value of the expression will not be strictly equal to 0 (if calculated in *float* format for Python or *float* or *double* for C++), and consequently degenerates to the third case of convergence from (7). That is why we get values close to 100 in the last column of Table 1 starting from the twentieth term of the sequence.

3. Conclusions and prospects for further research

To summarize, the reliability and accuracy of computer-based numerical calculations continue to challenge and shape the field of computational mathematics. Muller's method, as a representative iterative scheme, stands as both a beneficiary and a casualty of advances in floating-point arithmetic, error analysis, algorithmic innovation, and verification techniques. Contemporary research has created a panoply of root-finding algorithms—leveraging memory, derivative-free techniques, and adaptive parameterization—to strike a delicate balance between convergence acceleration, cost reduction, and error control [3-7]. Nonetheless, the ultimate reliability of these methods rests on a systematic understanding and proactive management of numerical error sources, as elucidated in foundational and modern studies alike [1,2,6,8,9].

The authors plan to take into account the results mentioned in this paper, particularly in the computer modeling of stochastic and natural processes, which has been explored in recent studies [19-25]. Furthermore, they also intend to utilize these results in the application of computational methods for addressing applied problems, such as those found in their recent studies concerning combinatorial games [26], economic models [27], election statistics [28], and Monte Carlo techniques [29].

Author Contributions: All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by grants from the Simons Foundation (PDUkraine-00010584, Krykun I. H. and SFI-PD-Ukraine-00017674, Krykun I. H.)

Acknowledgments: The authors express their heartfelt gratitude to the brave soldiers of the Ukrainian Armed Forces who protect the lives of the authors and their families from Russian bloody murderers since 2014.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] Higham, N. The Accuracy of Floating Point Summation. *SIAM J. Sci. Comput.* **1993**, *Vol. 14*, pp. 783-799.
[2] Cody, W. Static and Dynamic Numerical Characteristics of Floating-Point Arithmetic. IEEE *Transactions on Computers C-22* **1973**, pp. 598-601.
[3] Qureshi, S.; Soomro, A.; Naseem, A.; Gdawiec, K.; et al., From Halley to Secant: Redefining root finding with memory-based methods including convergence and stability. *Mathematical Methods in the Applied Sciences* **2024**, *Vol. 47*, pp. 5509-5531.
[4] Naseem, A.; Rehman, M.A.; and Abdeljawad, T. A Novel Root-Finding Algorithm With Engineering Applications and its Dynamics via Computer Technology. *IEEE Access* **2022**, *Vol. 10*, pp. 19677-19684.
[5] Özyapici, A.; Sensoy, Z.B.; and Karanfiller, T. Effective Root-Finding Methods for Nonlinear Equations Based on Multiplicative Calculi. *Journal of Mathematics* **2016**.
[6] Shams, M.; Kausar, N.; Araci, S.; and Kong, L. On the stability analysis of numerical schemes for solving non-linear polynomials arises in engineering problems. *AIMS Mathematics* **2024, *Vol. 9(4)*,** pp. 8885-8903.
[7] Nedzhibov, G. Dynamic Mode Decomposition via Polynomial Root-Finding Methods. *Mathematics* **2025**, *Vol. 13(5),* pp. 1-18.

[8] Tisseur, F. Newton's Method in Floating Point Arithmetic and Iterative Refinement of Generalized Eigenvalue Problems. *SIAM J. Matrix Anal. Appl*., **2000,** *Vol. 22*, pp. 1038-1057.
[9] Ozaki, K.; Terao, T.; Ogita, T.; and Katagiri, T. Verified Numerical Computations for Large-Scale Linear Systems. *Applications of Mathematics* **2021**, *Vol. 66*, pp. 269-285.
[10] Ryssens, W.; Heenen, P.; and Bender, M. Numerical accuracy of mean-field calculations in coordinate space. *Physical Review C* **2015**, Vol. 92, 064318.
[11] Kopec, D. *Classic Computer Science Problems in Python*, Manning Publications: Shelter Island, NY, USA, 2019. - 224 p.
[12] Kopec, D. *Classic Computer Science Problems in Java*, Manning Publications: Shelter Island, NY, USA, 2020. - 264 p.
[13] Muller, J.-M.; Brisebarre, N.; de Dinechin, F.; Jeannerod, C.-P; et al., *Handbook of Floating-Point Arithmetic*, Birkhäuser: Basel, Switzerland, 2018. - 627 p. DOI: 10.1007/978-3-319-76526-6
[14] Patriot Missile Defense: Software Problem Led to System Failure at Dhahran, Saudi Arabia: IMTEC-92-26, 1992. Available online: https://www.gao.gov/assets/220/215614.pdf
[15] Disasters due to rounding error. Available online: https://web.ma.utexas.edu/users/arbogast/misc/disasters.html
[16] Bilous, R.V.; Krykun, I.H. A Problem of Accuracy of Computer Calculations. *Universal Journal of Computer Sciences and Communications* **2022**, *Vol. 1, No. 1*, pp. 35-40. DOI: 10.31586/ujcsc.2022.531
[17] Kahan, W. How futile are mindless assessments of round off in floating-point computation? 2006. Available online: http://http.cs.berkeley.edu/~wkahan/Mindless.pdf
[18] Boldo, S.; Jeannerod, C.-P.; Melquiond, G.; and Muller, J.-M. Floating-point arithmetic. *Acta Numerica* **2023**, *Vol. 32,* pp. 203-290. DOI: 10.1017/S0962492922000101
[19] Krykun, I.H.; Makhno, S.Ya. The Peano phenomenon for Itô equations. *Journal of Mathematical Sciences* **2013**, *Vol. 192, Issue 4*, pp. 441-458. DOI: 10.1007/s10958-013-1407-5
[20] Krykun, I.H. Functional law of the iterated logarithm type for a skew Brownian motion. *Theory of Probability and Mathematical Statistics* **2013**, *Vol. 87*, pp. 79-98. DOI: 10.1090/S0094-9000-2014-00906-0
[21] Krykun, I.H. Convergence of skew Brownian motions with local times at several points that are contracted into a single one. *Journal of Mathematical Sciences* **2017**, *Vol. 221, Issue 5*, pp. 671-678. DOI: 10.1007/s10958-017-3258-y
[22] Krykun, I.H. The Arc-Sine Laws for the Skew Brownian Motion and Their Interpretation. *Journal of Applied Mathematics and Physics* **2018**, *Vol. 6, No. 2*, pp. 347-357. DOI: 10.4236/jamp.2018.62033
[23] Krykun, I.H. The Arctangent Regression and the Estimation of Parameters of the Cauchy Distribution. *Journal of Mathematical Sciences* **2020**, *Vol. 249, Issue 5*, pp. 739-753. DOI: 10.1007/s10958-020-04970-3
[24] Krykun, I.H. New Approach to Statistical Analysis of Election Results. *International Journal of Mathematical, Engineering, Biological and Applied Computing* **2022**, *1, No. 2*, pp. 68-76. DOI: 10.31586/ijmebac.2022.466
[25] Krykun, I.H. On weak convergence of stochastic differential equations with irregular coefficients. *Journal of Mathematical Sciences* **2023**, *Vol. 273, Issue 3*, pp. 398-413. DOI: 10.1007/ s10958-023-06506-x
[26] Cherniichuk, H.P.; Krykun, I.H. Notes about Winning Strategies for Some Combinatorial Games. *Journal of Mathematics Letters* **2022**, *Vol. 1, No. 1*, pp. 1-9. DOI: 10.31586/jml.2022.496
[27] Krykun, I.H. The Black-Scholes Exotic Barrier Option Pricing Formula. *Journal of Mathematics Letters* **2023**, *Vol. 1, No. 1*, pp. 10-18. DOI: 10.31586/jml.2023.604
[28] Krykun, I.H.; Pavlov, M.S. Statistics of Electoral Systems and Methods of Election Manipulation. *Journal of Social Mathematical & Human Engineering Sciences* **2023**, Vol. 1, No. 1, pp. 11-21. DOI: 10.31586/jsmhes.2023.610
[29] Krykun, I. H.; Lukhverchyk, S. A. Some Software Application of the Monte Carlo Method. *Universal Journal of Computer Sciences and Communications* **2023**, *Vol. 1, No. 1*, pp. 41-50. DOI: 10.31586/ujcsc.2023.704