Solution of Linear Equations using Scilab

P.Rao Mathe

Department of Mathematics, Govt. Degree College, Repalle, Andhra Pradesh, India.

Abstract

Numerical computation packages find extensive usage in both educational and research settings. These packages comprise both licensed and open-source software. One rationale for utilizing such packages is the complexity inherent in mathematical functions, particularly in linear problems. Moreover, there has been an increase in the number of variables within linear or non-linear functions. The objective of this paper was to examine fundamental aspects concerning methodology, pedagogy, and innovative practices in teaching linear equations within higher education. If adopted, this approach could significantly enhance mathematical learning, particularly in areas such as solving simultaneous linear equations, which are crucial for prospective engineers. This research primarily aimed to introduce Scilab as an alternative, cost-effective computational programming tool. Within this paper, Scilab software was proposed for various activities pertaining to mathematical modeling. Subsequently, these methodological routines could be effectively utilized as course materials in teaching.

INTRODUCTION

There exist challenges in transmitting knowledge within the fields of teaching numerical methods and modeling simulations, such as the bottleneck encountered in the elimination process when solving simultaneous linear equations. Many students encounter difficulties due to a lack of comprehension in classical algebra, finding it tedious to translate algebraic formulas into computer programming languages. Furthermore, manual analytical processes for complex issues have inherent limitations. Hence, innovation and diversification in the experiential learning of numerical computation processes are imperative.

In essence, numerical processes offer solutions to analytical mathematical problems, encompassing linear and non-linear equations, calculus derivatives and integrals, differential equations, as well as series and errors. Meanwhile, various methods, including Gaussian elimination, Gauss-Jordan, matrix inversion, Lower-Upper decomposition, Jacobi, Gauss-Seidel, and others, can be employed to tackle linear equations.

Presently, the complexity of linear problems is escalating, accompanied by an increase in the number of variables within linear/non-linear functions. Manual processing is no longer feasible in such scenarios, necessitating computational approaches. Numerous computational software options exist for resolving linear solutions, such as MS Excel, LINDO, MATLAB, and others. However, these software solutions are commercial or proprietary, mandating users to acquire a license for legal usage [1,2].

However, numerous alternatives exist in the form of free software and open source solutions. These software options are also capable of executing numerical computation processes with a high degree of accuracy. In this paper, we advocate for Scilab software as an alternative for addressing complex linear problems. The primary rationale behind this choice is that Scilab software can be developed freely, openly, and without commercial constraints, making it an affordable option [3-5].

Several researchers have explored Scilab as a tool in the field of artificial intelligence (AI). For instance, [6] investigated Scilab for time series forecasting using artificial neural networks (ANN), employing various algorithms such as ANN-GD with extended backpropagation (EBP), ANN-GA with genetic algorithm (GA), and ANN-DE with differential evolution (DE). The analysis involved two time series datasets: one from Hyndman's time series data library spanning January 1961 to October 1975, and another from monthly Wisconsin employment data from January 1962 to December 1975. The findings demonstrated that Scilab serves as a viable alternative tool for forecasting problems, offering ease of implementation and affordability.

Subsequently, Catarino P., Vasco P [3] explored Scilab for teaching linear algebra activities. Experimental outcomes revealed that Scilab commands simplify the process of writing mathematical formulas, particularly in linear algebra. Additionally, students noted that this feature enhances classroom dynamics and engagement.

Furthermore, Yusop N.M.M., Hasan M.K., Rahmat M [8] employed Scilab 5.4.0 Programming to solve mathematical models, specifically ordinary differential equations (ODE) using a modified Euler's method called Harmonic Euler. Results indicated that Scilab can serve as an alternative software programming tool, particularly for numerical methods.

METHODOLOGY

Introducing to Scilab Programming

Scilab is a programming language that offers a comprehensive array of numerical algorithms, addressing various scientific computing challenges. Originating in 1990, Scilab was developed by the French National Research Institution, Institut Nationale de Recherche en Informatique et en Automatique (INRIA). It serves as a free numerical computational package, boasting many similarities with proprietary software such as MATLAB and Minitab. Available for download on Linux, Windows, and Mac OS X operating systems, Scilab can be accessed via the web page at www.scilab.org/products/scilab/download. Additionally, Scilab seamlessly interfaces with LabVIEW, a platform and development environment for visual programming languages. Being open-source software, Scilab is provided under the CeCILL license, eliminating the need for license fees. Furthermore, Scilab offers an integrated editor called Scinotes, facilitating easy script editing, accessible through the console menu under Application > Editor or directly from the console [4].

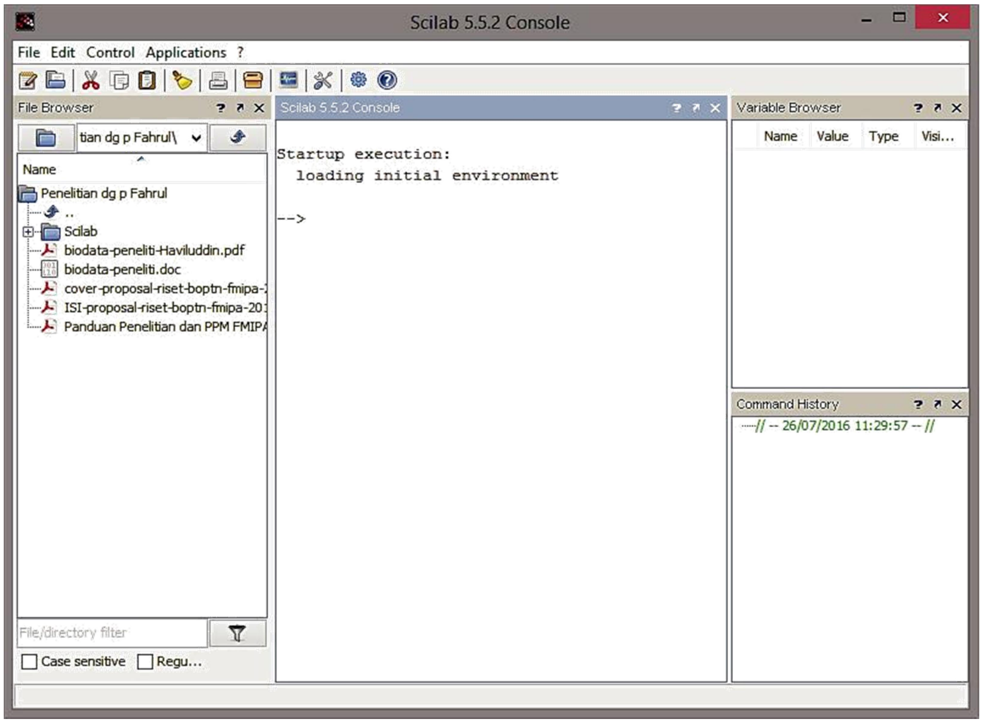

Fig.. The Scilab Console

Numerical Methods

Numerical methods encompass the generation of matrices to solve linear equations, ordinary differential equations (ODEs), and perform numerical integration (NI). They involve the utilization of mathematical modeling techniques [10,11]. This study provides a brief overview of four numerical methods: Gaussian Elimination, Gauss-Jordan, Inverse Matrix, and Lower-Upper Decomposition (LU).

The Gaussian elimination method is widely recognized as a technique for solving simultaneous linear equations, attributed to Karl Friedrich Gauss. This method involves manipulating values within a matrix to simplify it, ultimately transforming it into echelon form through row operations. Additionally, it can be applied to solve linear equations by augmenting the matrix and performing operations [3,12].

Gauss-Jordan Method

The Gauss-Jordan method is an enhancement of the Gaussian elimination method, yielding superior results. It extends the row operations of Gaussian elimination to achieve a matrix in reduced row echelon form [3,12].

Inverse Matrix Method

The inverse matrix method is another approach for solving linear equations. However, it is less efficient than Gaussian elimination due to its increased computational complexity. Nonetheless, it proves effective when dealing with similar matrices featuring different vectors [12]. According to

Matlab-Mathworks, "The inverse is returned as a matrix of the same type as the input matrix. If the matrix is not invertible, then 'fail' is returned. If the input does not evaluate to a matrix, then a symbolic call of inverse is returned" [13].

Lower-Upper Decomposition (LU) Method

The Lower-Upper Decomposition (LU) method is a variation of the Gaussian elimination method, wherein certain steps are omitted but are subsequently incorporated into the LU decomposition process [12]. According to Matlab-Mathworks, "The LU function expresses a matrix A as the product of two essentially triangular matrices, one of them a permutation of a lower triangular matrix and the other an upper triangular matrix. The factorization is often called the LU, or sometimes the LR, factorization. A can be rectangular" [13].

RESULTS AND DISCUSSIONS

In the present study, the general that involved in engineering solving and basic conceptual using Scilab programming was implemented. Number of numerical methods such as Gaussian Elimination, Gauss-Jordan, Inverse Matrix, and Lower-Upper Decomposition (LU) were used as test cases. Fahrul Agus and Haviluddin performed an experiment, using four numerical methods such as Gaussian Elimination, Gauss-Jordan, Inverse Matrix, and Lower-Upper Decomposition (LU). The results of this experiment showed that a routine or procedure in numerical methods have been created and explored by using Scilab procedures [14]. The codes for the above four numerical methods can be obtained from the work of Agus and Haviluddin [14]. The codes for other numerical methods can be obtained from the authors on request.

The routine of numerical methods based on the proposed ones by Fahrul Agus and Haviluddin were improved and tested for various numerical methods, all most all, used in Numerical Analysis and found that Scilab that could be used as a teaching material course for Mathematics and Statistics.

CONCLUSION

Numerical methods can be effectively demonstrated through Scilab, an open-source computational programming software widely recognized for its user-friendly interface, particularly in solving simultaneous linear equations. Many researchers have endorsed Scilab for its simplicity and effectiveness. Essentially, Scilab serves as a valuable alternative for teaching and learning numerical methods.

REFERENCES

1. Salleh Z., Yusop M.Y.M., Ismail S.B. Basic of Numerical Computational Using Scilab Programming. in the 2nd International Conference on Mathematical Applications in Engineering (ICMAE2012). pp. 1-8 (2012).
2. Baudin M., Couvert V. Optimization In Scilab. The Scilab Consortium - Digiteo / INRIA (2010).
3. Catarino P., Vasco P. Scilab in Linear Algebra. Applied Mathematical Sciences, vol. 8 2014, No. 28 pp. 1391-1399 (2014).
4. Baudin M., Steer S. Optimization with Scilab. present and future. pp. 1-8 (2009).

5. Wang Y., Liu L., Cao J., Feng K., Fu J., Li C. Random Forests Toolbox with Scilab and its Application. in The 7th International Conference on Computer Science & Education (ICCSE 2012) July 14-17 2012., Melbourne, Australia, pp. 1082-1085 (2012).
6. Panigrahi S., Karali Y., Behera H.S. Time Series Forecasting using Evolutionary Neural Network. Int J Comput Appl T (0975 – 8887), vol. 75– No.10, August 2013, pp. 13-17, (2013).
7. Singh A. Implementation of Back-Propagation Neural Network using Scilab and its Convergence Speed Improvement. Int. Journal of Electrical & Electronics Engg., vol. 2, Spl. Issue 1 (2015), pp. 192-194, (2015).
8. Yusop N.M.M., Hasan M.K., Rahmat M. Comparison New Algorithm Modified Euler in Ordinary Differential Equation Using Scilab Programming. Lecture Notes on Software Engineering. 3(3): 199-202, (2015).
9. Desmaële D., Renaud L., Tingry S. A wireless sensor powered by a flexible stack of membraneless enzymatic biofuel cells. Sensors and Actuators B, 220: 583–589 (2016).
10. Rao S. Applied Numerical Methods for Engineers and Scientist 3rd Edition (2002).
11. Salleh Z. Fundamental of Numerical Methods for Scientists and Engineers, 1st Edition (2011).
12. Urroz G.E. Matrix Operations with SCILAB. (2001).
13. MathWorks. LU Matrix Factorization. (2016).
14. Fahrul Agus and Haviluddin AIP Conference Proceedings 1813, 040005 (2017)