A COMPLEMENTARY STEM TEACHING RESOURCE: APPLICABILITY OF ANALYTICAL-NUMERIC METHODS AND PROGRAMMING LANGUAGE FOR THE ANALYSIS OF ISOSTATIC STRUCTURES

Giovani Jordi Bruschi[1], Paulo Roberto Dutra[2]

DOI: 10.37702/REE2236-0158.v39p357-368.2020

ABSTRACT

The STEM teaching resource can be defined as a tool that aids in the teaching of science, technology, engineering, and math (STEM). Most students see the exact sciences curriculum as boring and of the difficult appliance to life outside the theoretical field. Nevertheless, practical activities can enable students to link what they can see and handle with scientific ideas that account for their observations. Basic knowledge of programming, for example, is part of almost all STEM programs curricula, however, several challenges are reported on the connection of computer simulations with real-life applications. Thus, the objective of this research is to illustrate the applicability of two free-source software (SCILAB and Python) in the resolution of structural analysis problems. The results have shown that both programs are suited for the analysis of isostatic structures. Also, the utilization of the software may help the students to observe and evaluate the behavior of systems under different conditions, providing a true understanding of the studied concepts.

Keywords: STEM; structural analysis; open-source software; numerical methods; isostatic structures.

INTRODUCTION

Math learning is of difficult understanding for the majority of students, ranging from elementary school to higher education (KALELIOĞLU, 2015). In the latter case, the situation becomes even more delicate in exact sciences courses, since basic mathematics introduces differential and integral calculus, marked by its most varied aspects and practical applications (ROICK; RINGEISEN, 2018).

Due to the necessity of applied mathematics, basic knowledge of programming is part of almost all exact sciences programs curricula (KALELIOĞLU, 2015). However, both students and instructors are confronted with several challenges, and the programming concepts and language syntax may become barriers to the learning process, inhibiting the motivation of the students (ALI; T SMITH, 2014).

The main challenges in the teaching of programming are (a) the number of concepts and skills to learn in a limited time; (b) the absence of interactive media and instant feedback in instruction; and (c) the lack of mathematical background (MUSSO et al., 2019). Studies reveal that many of these problems originate from the complexity of concepts such as variables, loops, arrays, functions, and syntax in programming languages (REICHERT; NIEVERGELT;

[1] Master's student at the Federal University of Rio Grande do Sul (UFRGS) by the Graduate Program in Civil Engineering (PPGEC), Porto Alegre, RS. Contact: giovani.bruschi@ufrgs.br
[2] Graduated in Civil Engineering at the Regional University of Alto Uruguai and Missions – URI, *Campus* Erechim, RS.

HARTMANN, 2001; SERAFINI, 2011; TSAI, 2019).

Additionally, it is well-known that the utilization of the software has become more important than the coding itself. It is reported that the minority of the effort for software development is dedicated to coding. Rather than learning real problem-solving skills, the students resort to trial-and-error as a resolution system (MÉSZÁROSOVÁ, 2016; WANG et al., 2012).

Software with a source code that can be modified exclusively by the individual or company that maintains control over, is known as a closed-source software (e.g. McAfee anti-virus software) (BITZER; SCHRETTL; SCHRÖDER, 2007). Only the original authors can legally copy, review, and modify it, and in order to utilize this type of software, users must agree to follow the rules instituted by the creators. On the other hand, open-source software (e.g. SCILAB, Phyton) has the source code available to view, copy, share, and alter (VAN ANTWERP; MADEY, 2010).

The education sector, in conjunction with all educational activities, directly influence society through its teaching instruments. To utilize a closed-source software confronts one of the main goals of education, universality. Thus, the use of free-source software can aid students to develop and master the techniques of programming, while directing the future of education towards open-access tools.

In this scenario, the objective of this research is to illustrate the applicability of two distinctive free-source software (SCILAB and Python) in the resolution of structural analysis problems, providing subsidy to students and teachers for a full understanding of the contents taught in the classroom, more varied levels of teaching, as well as everyday situations that require the use of computational tools.

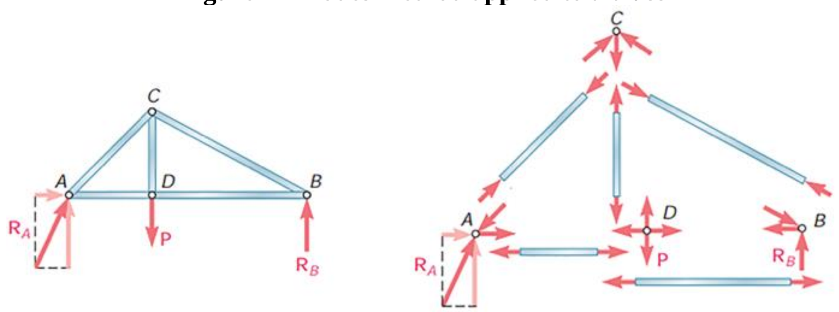
BACKGROUND

Structural components – truss and beam

The structural component known as truss has been adopted as a practical and economical solution for typical engineering structures, especially bridges and roofs. According to Alvarez et al. (2019), a truss is formed by relatively thin elements connected by the extremities, in which the external forces act in a single plane, justifying a two-dimensional analysis.

The Method of Joints can be applied to determine the internal forces acting on the truss. It consists of ensuring the static equilibrium of the structure by the equilibrium of each node (see Figure 1), with two unknown reactions involved in the problem. The arrangement of pins and bars in a simple truss allows finding a node that is subjected to only two forces of unknown intensity. Once defined, the values can be transferred to the other nodes until all variables are discovered (HIBBELER, 2013).

Figure 1 – Nodes method applied to a truss



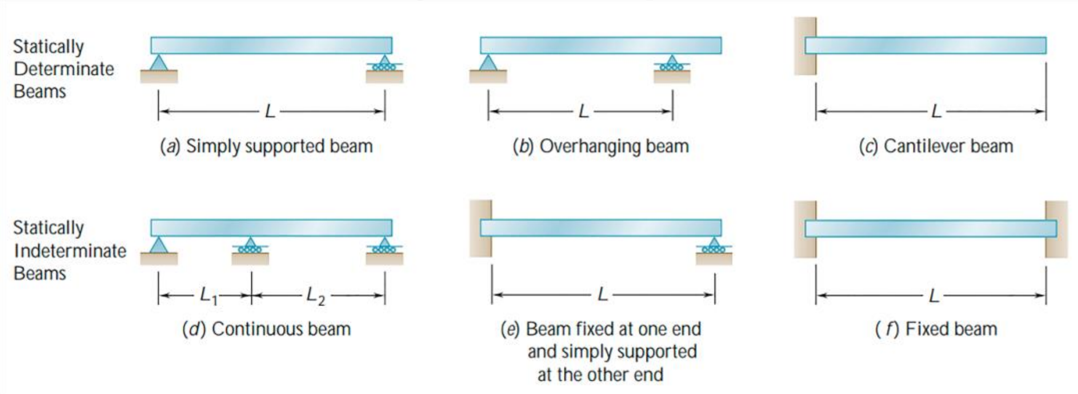
Source: Hibbeler (2013).

The mechanical behavior of steel joints in terms of strength, stiffness, and rotation capacity is a complex phenomenon. To determine this behavior, the joint can be decomposed into different parts, the so-called components (DASGUPTA, 2018).

The mechanical behavior of the components is normally studied separately. When all components of a joint have been characterized in terms of strength, stiffness, and deformation capacity, this behavior can be determined by assembling the individual contributions of the components with the aid of different models (ALVAREZ et al., 2019).

On the other hand, beams are prismatic and straight long bars, designed to withstand loads, especially flexural stresses and shear stresses (HIBBELER, 2013). As Figure 2 illustrates, they are generally classified according to the type of support.

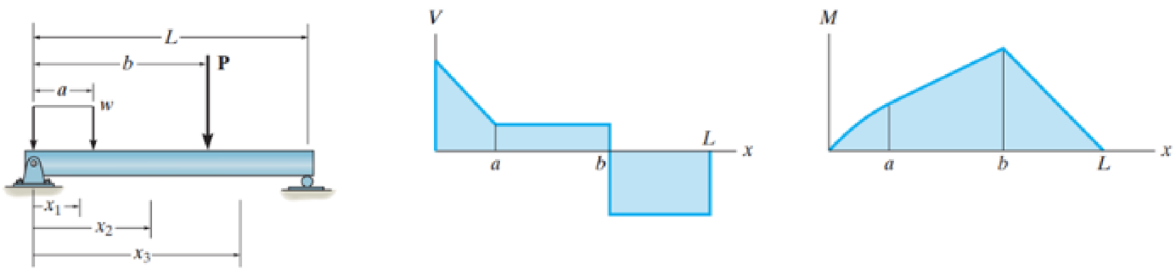
Figure 2 – Beam examples



Source: Hibbeler (2013).

The variations of the flexion bending moment (M) and the shear effort (V) on the extension of the beam can be measured with the use of the Method of Sections (Figure 3), which provides as product bending moment and shear stress diagrams (HIBBELER, 2013).

Figure 3 – Method of Sections



Source: Hibbeler (2013).

Numerical methods

Numerical methods are techniques by which mathematical problems are formulated to be solved through arithmetic operations. Although there are many types of numerical methods, they have one feature in common: a large number of extensive calculations. It is not surprising that, with the development of fast and efficient computers, the role of numerical methods in solving engineering problems has increased in recent years (KILYENI; BARBULESCU; SIMO, 2015).

These methods are extremely powerful tools in problem-solving. They are capable of handling a large number of equations, nonlinearities, and complicated geometries, common in engineering practice, and, in general, impossible to solve analytically (ZLATEV et al., 2020).

Numerical methods are also an efficient way to learn how to use computers. It is known that one of the bests forms to learn how to program is to write a computer program. Since numerical methods are, for the most part, designed for implementation on computers, they are ideal for this purpose (MÉSZÁROSOVÁ, 2016).

Furthermore, they are especially suited to illustrate the power and the limitations of computers. When numerical methods are successfully implemented on a computer and then applied to solve otherwise intractable problems, there is access to a dramatic demonstration of how computers can help professional development (KILYENI; BARBULESCU; SIMO, 2015). At the same time, one learns to identify and control errors in approximations, which are an essential part of large-scale numerical calculations. Generally, they can be used for the resolution of linear and nonlinear systems, problems involving function zeroes, optimization, ordinary differential equations, among other applications (ZLATEV et al., 2020).

Linear equation system

The Gauss elimination, despite not being a numerical method, is one of the most usual techniques for the resolution of linear systems. Also known as row reduction, the method is presented as an algorithm in linear algebra for solving a system of linear equations. It is usually understood as a sequence of operations performed on the corresponding matrix of coefficients. This method can also be used to find the rank of a matrix, to calculate the determinant, and to calculate the inverse of an invertible square matrix (ABBASBANDY; EZZATI; JAFARIAN, 2006).

To perform row reduction on a matrix, one uses a sequence of elementary row operations to modify it until the lower left-hand corner of the matrix is filled with zeros, as much as possible. There are three types of elementary row operations (a) swapping two rows; (b) multiplying a row by a nonzero number; and (c) adding a multiple of one row to another row (WANG; REN; DUAN, 2019).

Using these operations, a matrix can be transformed into an upper triangular matrix. Once all of the leading coefficients (the leftmost nonzero entry in each row) are 1, and every column containing a leading coefficient has zeros elsewhere, the matrix is said to be in reduced row echelon form (WANG; REN; DUAN, 2019). This final form is unique; in other words, it is independent of the sequence of row operations used, as exemplified in Figure 4.

Figure 4 – Simplified Gauss elimination algorithm

$$\left(\begin{array}{ccc|c} a_{11}, & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \end{array}\right) = \left(\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & b_1 \\ 0 & a'_{22} & a'_{23} & b'_2 \\ 0 & 0 & a"_{33} & b"_3 \end{array}\right)$$

$$x_3 = b"_3/a"_{33}$$
$$x_2 = (b_2 - a_{13} * x_3/a'_{22}$$
$$x_1 = (b_1 - a_{12} * x_2 - a_{13} * x_3)/a_{11}$$

Source: Wang, Ren e Duan (2019).

Besides Gaussian Elimination, LU factorization (or decomposition) has also been widely utilized. In numerical analysis and linear algebra, lower–upper decomposition factors a matrix as the product of a lower triangular matrix and an upper triangular matrix (ABBASBANDY; EZZATI; JAFARIAN, 2006).

In this case, the final solution is given as a function of the resolution of smaller linear systems (Ly = b Ux = y), which becomes useful when the coefficient matrix remains the same while the matrix of constants b is changed, as shown in Figure 5 (ABBASBANDY; EZZATI; JAFARIAN, 2006).

Figure 5 – Simplified LU factorization algorithm

$$\begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} * \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & l_{33} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Source: Abbasbandy, Ezzati e Jafarian (2006).

LU decomposition can be viewed as the matrix form of Gaussian Elimination. Computers usually solve square systems of linear equations using LU decomposition, and it is also a key-step when inverting a matrix or computing the determinant of a matrix (WANG; REN; DUAN, 2019).

SCILAB and programming logic

SCILAB can be defined as an open-source software for numerical analysis that provides a computing environment for scientific and engineering applications. It is similar in operation to MATLAB and other existing numerical/graphic environments, and it can be utilized in different operating systems (e.g. Windows, Linux) (MAGYAR; ŽÁKOVÁ, 2012).

The language utilized in SCILAB arranges an interpreted programming environment, with matrices as the main data type. When compared to other traditional languages, the mathematical problems solved in SCILAB can be expressed in contracted code lines, due to the use of matrix-based computation, dynamic typing, and automatic memory management (STILBS, 2019).

This feature concedes the fast construction of models for several numerical problems. While the language provides simple matrix operations such as multiplication, the SCILAB package also provides a library of high-level operations such as correlation and complex multidimensional arithmetic. The software can be used for signal processing, statistical analysis, image enhancement, fluid dynamics simulations, structural analysis, and numerical optimization (STILBS, 2019).

Phyton and programming logic

Python is an open-source language interpreted within a strong dynamic typing and rich standard library. Like other languages, a program in Python is organized around select and repeat structures. However, Python distinguishes itself by working with lists and dictionaries, mutable data sets that can be accessed at any time. In particular, it allows procedural and objects oriented programming (WAGNER et al., 2017).

It supports introspection allowing, for example, dynamically consultation of the composition of a class, an object, or a code file. This facilitates the programming of a component-oriented approach to develop modular software where the modules can be inspected and loaded dynamically (DOTSENKO et al., 2019).

For structural analysis, this tool assists in obtaining the internal requesting efforts, as well as in the graphical representation of the results. Thus, one can proceed to the dimensioning of the structure (WAGNER et al., 2017).

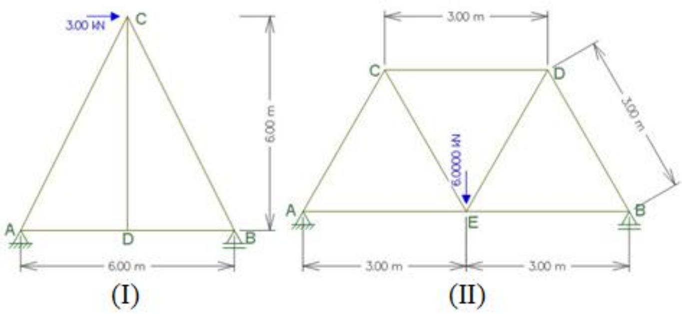
METHOD

This research was structured in two distinct phases: i) analysis of two isostatic truss and an isostatic beam, utilizing SCILAB and Phyton software respectively; and ii) utilization of Ftool, an engineering software for structural analysis, for the validation of the results provided by SCILAB and Phyton.

Isostatic truss analysis

Aiming to evaluate the applicability of the methods, two different types of truss (Figure 6) were proposed, with different degrees of complexity.

Figure 6 – Analyzed truss



Source: Authors own elaboration.

Firstly, the Method of Joints was applied to discover the internal forces acting on model I. Without the previous calculation of the support reactions, a system of linear equations of order equal to the number of members of the truss was defined, whose resolution alluded to the Elimination of Gauss.

Regarding model II, Scilab interface was utilized to generate the results, through direct resolution, applying the left division operator \ (Ax = B, then x = A \ b) and LU factorization, through the function LU, which returns the L and U factors of Gaussian elimination. Similar to what was previously delimited, a characteristic linear system was assigned to the structure.

Isostatic beam analysis

The analysis of the beam (Figure 7) was granted in a more simplified way. Initially, the shear and bending moment diagrams were calculated along the entire length of the beam, with the aid of Python software. Then, the IDLE was used to graphically represent the internal forces acting on the structure.

Figure 7 – Analyzed beam

Source: Authors own elaboration.

RESULTS AND DISCUSSION

The Elimination of Gauss was applied for the resolution of truss I, resulting in a system of linear equations as shown in Figure 8.

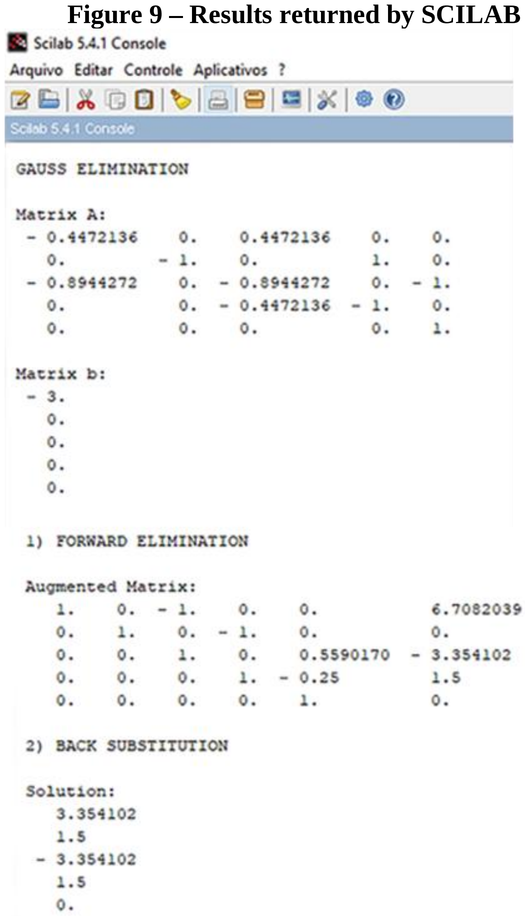
Figure 8 – Linear system obtained for truss I

$$\begin{bmatrix} -0.4472136 & 0 & 0.4472136 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ -0.8944272 & 0 & -0.8944272 & 0 & -1 \\ 0 & 0 & -0.4472136 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} FAC \\ FAD \\ FBC \\ FBD \\ FCD \end{bmatrix} = \begin{bmatrix} -3 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Source: Authors own elaboration.

This linear system was then transformed in an algorithm and executed in the SCILAB console. The results are illustrated in Figure 9.

Figure 9 – Results returned by SCILAB



```
Scilab 5.4.1 Console
Arquivo Editar Controle Aplicativos ?
Scilab 5.4.1 Console

GAUSS ELIMINATION

Matrix A:
 - 0.4472136    0.     0.4472136    0.    0.
   0.         - 1.     0.           1.    0.
 - 0.8944272    0.   - 0.8944272    0.  - 1.
   0.           0.   - 0.4472136  - 1.    0.
   0.           0.     0.           0.    1.

Matrix b:
 - 3.
   0.
   0.
   0.
   0.

 1) FORWARD ELIMINATION

 Augmented Matrix:
    1.    0.  - 1.    0.    0.            6.7082039
    0.    1.    0.  - 1.    0.            0.
    0.    0.    1.    0.    0.5590170   - 3.354102
    0.    0.    0.    1.  - 0.25          1.5
    0.    0.    0.    0.    1.            0.

 2) BACK SUBSTITUTION

 Solution:
    3.354102
    1.5
  - 3.354102
    1.5
    0.
```

Source: Authors own elaboration.

As for truss model II, the structure was analysed through the Method of Joints and presented as a result the system shown in Figure 10.

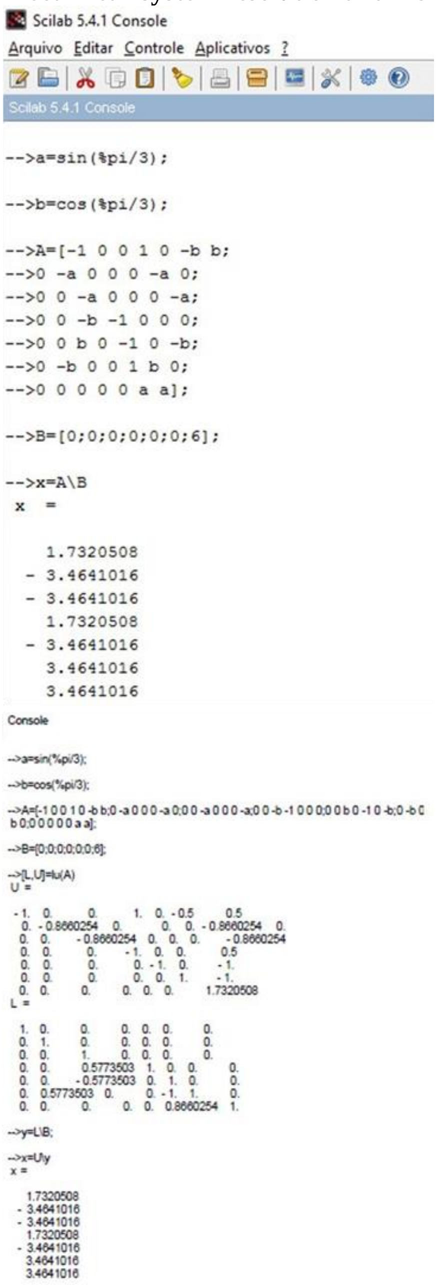
Figure 10 – Linear system obtained for truss II

$$\begin{bmatrix} -1 & 0 & 0 & 1 & 0 & -0.5 & 0.5 \\ 0 & -0.8660254 & 0 & 0 & 0 & -0.8660254 & 0 \\ 0 & 0 & -0.8660254 & 0 & 0 & 0 & -0.8660254 \\ 0 & 0 & -0.5 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & -1 & 0 & -0.5 \\ 0 & -0.5 & 0 & 0 & 1 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.8660254 & 0.8660254 \end{bmatrix} * \begin{bmatrix} FAE \\ FAC \\ FBD \\ FBE \\ FCD \\ FCE \\ FDE \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 6 \end{bmatrix}$$

Source: Authors own elaboration.

This system was calculated in the SCILAB console with a direct linear system resolution and LU factorization. The results are presented in Figure 11.

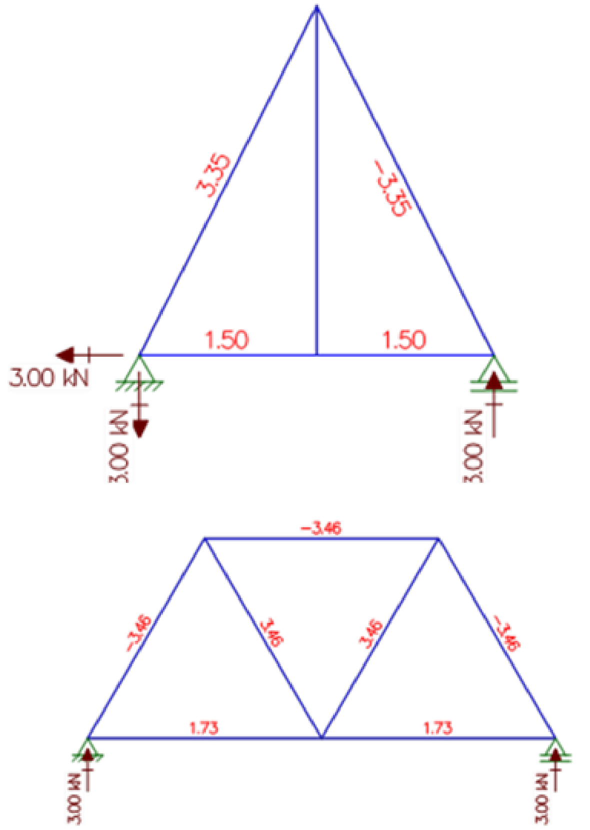
Figure 11 – Direct linear system resolution and LU factorization



Source: Authors own elaboration.

The Ftool software was utilized to compare the results with the ones provided by the SCILAB software. The diagrams of the Ftool, displaying the values of the acting stresses, are presented in Figure 12.

Figure 12 – Values obtained by the Ftool software



Source: Authors own elaboration.

An excellent agreement between the results provided by SCILAB and Ftool was identified, ratifying the viability of the structural analysis through linear systems. In accordance with Meguro and Tagel-din (2000), the linear structural analysis is based on two fundamental assumptions, namely, (a) material linearity – i.e., the structures are composed of linear elastic material, and (b) geometric linearity, implying that the structural deformations are so small that the equations of equilibrium can be expressed in the undeformed geometry of the structure.

The advantage of making these assumptions is that the equations relating applied loads to the resulting structural deformations become linear which can be solved conveniently (MEGURO; TAGEL-DIN, 2000). Another important advantage of linear load-deformation relations, in consonance with Hibbeler (2013), is that the principle of superposition can be used to simplify the analysis.

Linear analysis generally yields accurate results for most common structures under service loading conditions. However, because of its inherent limitations, the analysis cannot predict structural response in the large deformation and/or failure range, nor it can detect instability conditions of structures (HIBBELER, 2013; MEGURO; TAGEL-DIN, 2000).

In order to explore a diverse extension of open-source software, the structural beam was analysed through the Phyton software. Thus, an algorithm (as shown in Figure 13) was created to calculate the acting stresses on the structure.

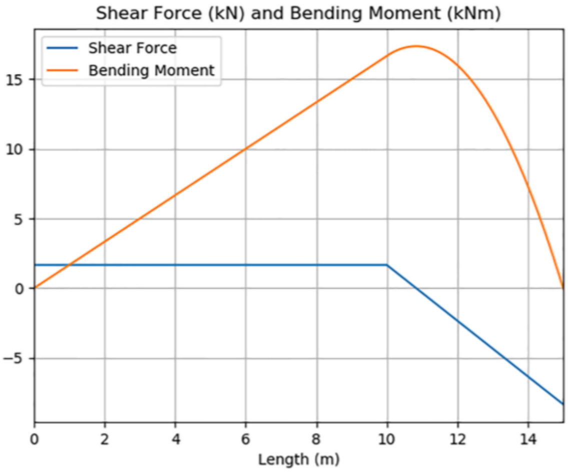
Figure 13 – Algorithm used to generate the graphical results in Python

```
import matplotlib.pyplot as plt

x1=[];x2=[] #Sections
V1=[];V2=[] #Shear force along the sections
M1=[];M2=[] #Bending moment along the sections

a=0;b=10;c=15 #Start and end of sections
x=a #Section 1 start at x equal to 0

while x<=b: #Section 1
    x1.append(x)
    x=x+0.1

for i in x1:
    V1.append(5/3*i**0) #Shear Force Equation
    M1.append(5/3*i**1) #Bending Moment Equation

while x>=b and x<=c: #Section 2
    x2.append(x)
    x=x+0.1

for i in x2:
    V2.append(5/3-2*(i-10)) #Shear Force Equation
    M2.append(5/3*i-(i-10)**2) #Bending Moment Equation

shear=V1+V2 #Total Shear Force
bending=M1+M2 #Total Bending Moment
x=x1+x2 #X Axis

plt.plot(x,shear,label='Shear Force') #Plot the shear force along the beam
plt.plot(x,bending,label='Bending Moment') #Plot the bending moment along
the beam
plt.title("Shear Force (kN) and Bending Moment (kNm)")
plt.xlabel("Length (m)") #Beam Length
plt.xlim(0,15) #Beam length: from 0 to 15 meters
plt.legend(loc='best') #Show the Legend
plt.grid(True) #Show the Grid
plt.show() #Show the Diagrams
```

Source: Authors own elaboration.

Applying the created algorithm in the software, it was possible to directly generate a diagram of the internal stresses of the beam (shear force and bending moment). This diagram is shown in Figure 14.

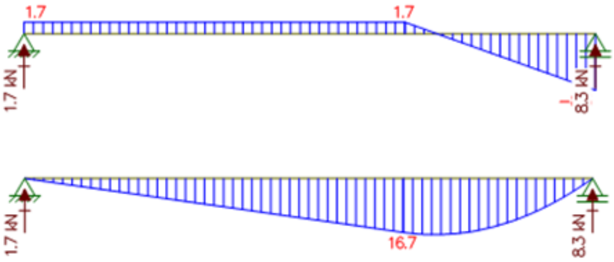
Figure 14 – Effort diagram obtained through Python



Source: Authors own elaboration.

Once again, the Ftool software was utilized as a comparative parameter for the discovered results. The values provided by the Ftool are illustrated in Figure 15.

Figure 15 – Values obtained by Ftool software for the analyzed beam



Source: Authors own elaboration.

Both results showed extremely high agreement. It can be seen that Python offers rich mechanisms and tools for the rapid development of modular and high-performance applications. Furthermore, the program depicts a large number of libraries covering numerous problems treated in the signal and processing of images.

The capabilities of current technologies demonstrate the possibility to create a more interactive resolution system for engineering problems. For example, structural analysis is not obvious from mathematical representations. However, if the time response of equation solutions is observed, scrutinizing every resolution step, the understanding becomes clearer.

The direct utilization of structural analysis programs, such as Ftool, may lead to rather mechanistic solutions. In this scenario, the students only learn how to insert the dataset on the program, without the comprehension of the resolution mechanism acting on the system. On the other hand, SCILAB and Python software offers a variety of visualization aids to better match the different cognitive styles of the students, and the interactive component requires students to control their exploration in ways that the predominantly reading of a text and direct resolution programs cannot.

Students in engineering courses may especially benefit from these techniques. The topics are complicated, multifaceted, and interconnected. The students are often so overwhelmed with facts and information that they may fail to understand the broader context of their education, and thus fail to develop the ability to absorb new data and observations in context, a skill that is needed during a professional career.

By utilizing programming language in conjunction with analytic methods, students can develop an intuitive understanding of digital functioning, as well as the ability to analyze new technologies and locate them into specific contexts. This leads to the full comprehension of concepts, both theoretically and practically.

CONCLUDING REMARKS

There are many uses for analytical-numeric methods in engineering. However, these applications are not always conveyed to students. The approach of practical problems through numerical and computational methods is a very feasible alternative, which justifies its adoption as a support tool for the analytical resolution of mathematical problems. In more complicated situations, the analytical method can easily be replaced by a computational approach with adequate accuracy.

Regarding the analysis of the proposed truss, the vast range of functionalities offered by SCILAB stands out, which allows it to be adapted to a large number of problems. As a result, an excellent agreement between the data provided by SCILAB and Ftool was identified. The same behavior was evidenced in the analysis of the beam, in which the Phyton software worked as an interpreted, object-oriented, high-level programming language with dynamic semantics. Phyton was also capable of generating a diagram that represented the internal stresses of the beam. Supplementary, the utilization of free-source programs, such as SCILAB and Python, contributes to the universalization of the education.

The most common learning environment for exact sciences courses, with the traditional combination of texts, discussions, and laboratories presents limitations that may restrict student success. Supplemental STEM teaching resource such as numerical simulations and programming allow the student to observe and evaluate the behavior of systems under different conditions. This helps with the true understanding of concepts and the development of a basis for more advanced studies.

Finally, there is still the possibility of using different programs to solve problems, such as roots of equations, optimization, differential equations, as well as problems of a simpler nature. Thus, this research shows practical reasons for the popularization of different computational methods as useful tools for teaching and learning in Exact Sciences, while allowing the student access to new educational resources.

REFERÊNCIAS

ABBASBANDY, S.; EZZATI, R.; JAFARIAN, A. LU decomposition method for solving fuzzy system of linear equations. **Applied Mathematics and Computation**, v. 172, n. 1, p. 633-643, jan. 2006.

ALI, A.; T SMITH, D. Teaching an Introductory Programming Language in a General Education Course. **Journal of Information Technology Education: Innovations in Practice**, v. 13, p. 57-67, 2014.

ALVAREZ, R. et al. Nonlinear cyclic Truss Model for analysis of reinforced concrete coupled structural walls. **Bulletin of Earthquake Engineering**, v. 17, n. 12, p. 6419-6436, dez. 2019.

BITZER, J.; SCHRETTL, W.; SCHRÖDER, P. J. H. Intrinsic motivation in open source software development. **Journal of Comparative Economics**, v. 35, n. 1, p. 160-169, mar. 2007.

DASGUPTA, G. **Finite Element Concepts**. New York, NY: Springer New York, 2018.

DOTSENKO, A. V. et al. Python Package for the Network Operator Implementation and its Application for Group Control System Synthesis Problem. **Procedia Computer Science**, v. 150, p. 679-686, 2019.

HIBBELER, R. C. **Resolução resistência dos materiais**. 8. ed. [s.l: s.n.], 2013.

KALELIOĞLU, F. A new way of teaching programming skills to K-12 students: Code.org. **Computers in Human Behavior**, v. 52, p. 200-210, nov. 2015.

KILYENI, S.; BARBULESCU, C.; SIMO, A. POWERNUM. Power Engineering Numerical Methods Educational Software Library. **Procedia - Social and Behavioral Sciences**, v. 191, p. 1783-1789, jun. 2015.

MAGYAR, Z.; ŽÁKOVÁ, K. SciLab Based Remote Control of Experiments. **IFAC Proceedings Volumes**, v. 45, n. 11, p. 206-211, 2012.

MEGURO, K.; TAGEL-DIN, H. Applied element method for structural analysis: Theory and application for linear materials. **Structural Engineering/Earthquake Engineering**, v. 17, n. 1, 2000.

MÉSZÁROSOVÁ, E. Is Python an Appropriate Programming Language for Teaching Programming in Secondary Schools? **International Journal of Information and Communication Technologies in Education**, v. 4, n. 2, p. 5-14, 2016.

MUSSO, M. F. et al. Individual differences in basic cognitive processes and self-regulated learning: their interaction effects on math performance. **Learning and Individual Differences**, v. 71, n. March, p. 58-70, abr. 2019.

REICHERT, R.; NIEVERGELT, J.; HARTMANN, W. Programming in schools – why, and how? **Enseigner l'informatique**, n. October, p. 143-152, 2001.

ROICK, J.; RINGEISEN, T. Students' math performance in higher education: Examining the role of self-regulated learning and self-efficacy. **Learning and Individual Differences**, v. 65, n. September 2016, p. 148-158, jul. 2018.

SERAFINI, G. Teaching Programming at Primary Schools: Visions, Experiences, and Long-Term Research Prospects. In: **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**. [s.l: s.n.]. v. 7013 LNCS, p. 143-154, 2011.

STILBS, P. **Diffusion and Electrophoretic NMR**. [s.l.] Walter de Gruyter GmbH, 2019.

TSAI, C.-Y. Improving students' understanding of basic programming concepts through visual programming language: The role of self-efficacy. **Computers in Human Behavior**, v. 95, p. 224-232, jun. 2019.

VAN ANTWERP, M.; MADEY, G. **The Importance of Social Network Structure in the Open Source Software Developer Community**. 2010 43rd Hawaii International Conference on System Sciences. **Anais**...IEEE, 2010.

WAGNER, M. et al. Performance Analysis of Parallel Python Applications. **Procedia Computer Science**, v. 108, p. 2171-2179, 2017.

WANG, P.; REN, W.; DUAN, Z. Distributed Algorithm to Solve a System of Linear Equations With Unique or Multiple Solutions From Arbitrary Initializations. **IEEE Transactions on Control of Network Systems**, v. 6, n. 1, p. 82-93, mar. 2019.

WANG, Y. et al. Assessment of programming language learning based on peer code review model: Implementation and experience report. **Computers & Education**, v. 59, n. 2, p. 412-422, set. 2012.

ZLATEV, Z. et al. Advanced numerical methods for complex scientific and engineering problems: Editorial introduction. **Journal of Computational and Applied Mathematics**, v. 372, p. 112596, jul. 2020..

DADOS BIOGRÁFICOS DOS AUTORES

Giovani Jordi Bruschi – Master's student at the Federal University of Rio Grande do Sul (UFRGS) by the Graduate Program in Civil Engineering (PPGEC). Graduated in Civil Engineering at the Regional University of Alto Uruguai and Missions - URI *Campus* Erechim. Scientific Initiation Scholarship from the National Council for Scientific and Technological Development (CNPq) at the Construction Technology Laboratory of the Regional University of Alto Uruguai and Missions - URI *Campus* Erechim during 2016.

Paulo Roberto Dutra – Graduated in Civil Engineering at the Regional University of Alto Uruguai and Missions - URI *Campus* Erechim. Scientific Initiation Scholarship from the National Council for Scientific and Technological Development (CNPq) at the Construction Technology Laboratory of the Regional University of Alto Uruguai and Missions - URI *Campus* Erechim during 2016.