Python Program for 3D Linear Static Reticular Structural Analysis Based on Finite Element Method

Gabriel C. Ferreira, Maria S. M. Sampaio

Dept. of Civil Engineering, University of Amazonas State
Darcy Vargas Avenue, 69050-020, Amazonas/Manaus, Brazil
gcarva.ferreira@gmail.com, msampaio@uea.edu.br

Abstract. Structural analysis in civil engineering is the method, in which, the response of a real-world structure before its design and construction is predicted. For any non-simplistic structure, it's hard to conceive the analysis by hand without relying on simplifications, that by themselves, sacrifices precision of the analytical solutions. Therefore, to solve these structures quickly and accurately, a computational tool is needed. These computational tools, usually available in commercial version, come at a high monetary cost, causing inaccessibility for students to have access, even for confirming and/or learning purposes. In this regard, this paper proposes to develop a free and open-source computational tool, able to make structural analysis of reticular structures in 3D linear-static regime. To make it possible, the Finite Element Method for structural discretization and the Python programming language were used. Analytical and numerical examples available in the literature were exhaustively tested. The differences between the solutions obtained with the numerical tool and the reference solutions are in the range of 10^{-7} to 10^{-5} showing the good performance of the implemented tool for the tested examples.

Keywords: python, 3D structural analysis, finite element method, static analysis, reticular finite elements.

1 Introduction

Engineering programs related to structural analysis help the user to find solutions for a given problem. Many engineering problems can be expressed in terms of partial differential derivatives and have analytical solution. However, in real problems, this feature is not observed and it is necessary to use a numerical tool to obtain approximate solutions to the problem. A very widespread method for calculating approximate solutions is the Finite Element Method (FEM). Its main characteristic is to divide a body into finite elements, connected by nodes, and obtain an approximate solution to the analyzed problem (Zienkiewicz and Taylor [1], Hughes [2], Oden and Reddy [3]).

There are different methods to derive the finite element system of equations, among them the variational methods and the weighted residuals method stand out (Zienkiewicz and Taylor [1], Hughes [2], Oden and Reddy [3], Reddy [4]). In this paper an energy method was used to derive the system of equations of the 3D reticular finite element. This method consists of writing the total potential energy functional of the element and applying the Principle of Potential Energy Stationarity. This principle, states that of all displacements fields which satisfy the prescribed constraint conditions, the correct state is that which makes the total energy of the structure at it's minimum. The choice of an energy method is due to the simplicity of the physical concepts of external work and strain energy involved in this formulation when compared to the mathematical refining of other methods.

The finite element developed here has two nodes and six degrees of freedom per node, totaling twelve degrees of freedom per element. The nodal parameters are translations and rotations. A cubic polynomial and a linear polynomial are used, respectively, to approximate the solutions of the problem and describe the loading. The total potential energy functional and the Potential Energy Stationarity Principle are used to obtain the finite element equations system. Python 3.10.7 programming language and Spyder IDE 5.4.4 are used to implement the developed algorithm. After determining the problem unknowns, displacements and rotations, by solving the system of equations gave by Hooke's Law, the internal forces and support reactions are determined in the post processing phase. To validate the implemented code, some examples available in the literature were tested and show that the code was successfully implemented.

There are different computational tools for structural analysis, usually available in commercial version. A similar educational and open-source programs is presented by Rangel and Martha [5]. So, the main purpose of this paper is to develop a free and open-source computational tool in Python language, able to make structural analysis of reticular structures in 3D linear-static regime using the FEM. The code is available in a GitHub repository. The link is presented in the conclusion.

2 Finite Element Formulation for 3D Reticular Linear Static Analysis

Let a frame finite element of length (L), cross-section area (A), Young modulus (E), shear modulus (G) and inertial moments (I_X, I_Y and I_Z), be subjected to a linearly distributed load q(x). Since, by hypothesis, the element is subjected to axial forces, bending forces and torsion, the nodal parameters associated with nodes i and j of this finite element are rotations and translations as shown in Fig. 1.

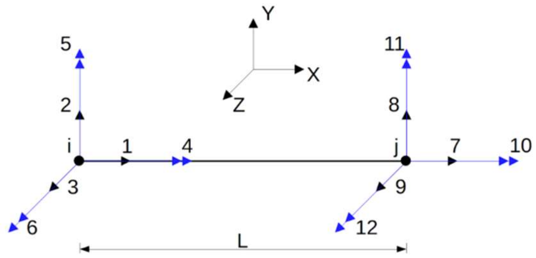

Figure 1. Finite element nodal degrees of freedom.

The governing equation of the problem can be obtained by the balance of forces or alternatively by the stationarity condition of the functional energy, which for static problems is given by Eq. (1).

$$\Pi = U - W \tag{1}$$

where U is the strain energy and W is the work done by external loads.

For linear elastic bodies under triaxial stress and strain, the strain energy U in Eq. (1) is given by Eq. (2).

$$U = \frac{1}{2}\int_V (\sigma_x \varepsilon_x + \sigma_y \varepsilon_y + \sigma_z \varepsilon_z + \tau_{xy}\gamma_{xy} + \tau_{xz}\gamma_{xz} + \tau_{yz}\gamma_{yz})dV \tag{2}$$

For the element shown in Fig. 1, the strain energy U in Eq. 1, has the contribution of axial forces, bending forces and torsion as shown in Eq. (3).

$$U = \pi_a + \pi_b^{x-y} + \pi_b^{x-z} + \pi_t \tag{3}$$

The strain energy for axial forces contribution is given by Eq. (4).

$$\pi_a = \frac{1}{2}\int_V (\sigma_x \varepsilon_x)dV = \frac{1}{2}\int_0^L EA\left(\frac{du_a}{ds}\right)^2 dx \tag{4}$$

The strain energy for bending contributions in x-y and x-z plane is given by Eq. (5).

$$\pi_b = \frac{1}{2}\int_V (\sigma_x \varepsilon_x)dV = \frac{1}{2}\int_0^L EI\left(\frac{d^2u_b}{ds^2}\right)^2 dx \tag{5}$$

The strain energy for torsion contribution is given by Eq. (6).

$$\pi_t = \frac{1}{2}\int_V (\tau_{yz}\gamma_{yz})dV = \frac{1}{2}\int_0^L GJ\left(\frac{d\theta_t}{ds}\right)^2 dx \tag{6}$$

According to the order of the derivatives present in the energy functional, polynomials of different order are chosen to approximate the solutions. The coeficientes in these polinomials are constants, obtained from the boundary conditions of the finite element. After the replacement of the coefficients in the aproximations, the resulting equation is derived according to the order of the derivative in the energy functional under analysis, and after all the necessary operations, the functional is minimized by differentiating with respect to the unknowns of the problem.

This is then done for each contribution, that is, axial, x-y and x-z bendings and torsion. The terms obtained can be organized into a single matrix, called the stiffness matrix, or **K** for short, from the Hooke's Law. Respecting the numbering of the degrees of freedom of the element shown in Fig. 1. The resulting **K** matrix, is given in Eq. 7.

$$\mathbf{K}=\begin{bmatrix}
\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{12EI_Z}{L^3} & 0 & 0 & 0 & \frac{6EI_Z}{L^2} & 0 & -\frac{12EI_Z}{L^3} & 0 & 0 & 0 & \frac{6EI_Z}{L^2} \\
0 & 0 & \frac{12EI_Y}{L^3} & 0 & -\frac{6EI_Y}{L^2} & 0 & 0 & 0 & -\frac{12EI_Y}{L^3} & 0 & -\frac{6EI_Y}{L^2} & 0 \\
0 & 0 & 0 & \frac{GI_X}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{GI_X}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_Y}{L^2} & 0 & \frac{4EI_Y}{L} & 0 & 0 & 0 & \frac{6EI_Y}{L^2} & 0 & \frac{2EI_Y}{L} & 0 \\
0 & \frac{6EI_Z}{L^2} & 0 & 0 & 0 & \frac{4EI_Z}{L} & 0 & -\frac{6EI_Z}{L^2} & 0 & 0 & 0 & \frac{2EI_Z}{L} \\
-\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{12EI_Z}{L^3} & 0 & 0 & 0 & -\frac{6EI_Z}{L^2} & 0 & \frac{12EI_Z}{L^3} & 0 & 0 & 0 & -\frac{6EI_Z}{L^2} \\
0 & 0 & -\frac{12EI_Y}{L^3} & 0 & \frac{6EI_Y}{L^2} & 0 & 0 & 0 & \frac{12EI_Y}{L^3} & 0 & \frac{6EI_Y}{L^2} & 0 \\
0 & 0 & 0 & -\frac{GI_X}{L} & 0 & 0 & 0 & 0 & 0 & \frac{GI_X}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_Y}{L^2} & 0 & \frac{2EI_Y}{L} & 0 & 0 & 0 & \frac{6EI_Y}{L^2} & 0 & \frac{4EI_Y}{L} & 0 \\
0 & \frac{6EI_Z}{L^2} & 0 & 0 & 0 & \frac{2EI_Z}{L} & 0 & -\frac{6EI_Z}{L^2} & 0 & 0 & 0 & \frac{4EI_Z}{L}
\end{bmatrix} \tag{7}$$

3 Rotation of Axes in Three Dimensions

The space frame element may have its principal axes in general directions then, there are various ways in which the orientation of theses axes can be defined in Weaver and Gere [6]. In the present formulation, the rotation matrix R has the form shown in Eq. (8).

$$\mathbf{R}=\begin{bmatrix} \bar{\mathbf{R}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \bar{\mathbf{R}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \bar{\mathbf{R}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{R}} \end{bmatrix} \tag{8}$$

In Eq. (8), $\bar{\mathbf{R}}$ is a 3x3 matrix given by Eq. (9).

$$\bar{\mathbf{R}}=\begin{bmatrix} Cx & Cy & Cz \\ -(Cx\cdot Cy\cdot\cos\delta+Cz\cdot sen\delta)/Cxz & Cxz\cdot\cos\delta & (-Cy\cdot Cz\cdot\cos\delta+Cx\cdot sen\delta)/Cxz \\ (Cx\cdot Cy\cdot sen\delta-Cz\cdot sen\delta)/Cxz & -Cxz\cdot sen\delta & (Cy\cdot Cz\cdot sen\delta+Cx\cdot\cos\delta)/Cxz \end{bmatrix} \tag{9}$$

Se $Cxz \neq 0$, $\mathbf{0}$ is a 3x3 matrix, δ is the principal axes angle, $Cx=(x_j-x_i)/L$, $Cy=(y_j-y_i)/L$, $Cz=(z_j-z_i)/L$, $Cxz=\sqrt{Cx^2+Cz^2}$ and $L=\sqrt{(x_j-x_i)^2+(y_j-y_i)^2+(z_j-z_i)^2}$.

On the other hand, if $Cxz=0$ the matrix $\bar{\mathbf{R}}$ is given by Eq. (10).

$$\bar{\mathbf{R}}=\begin{bmatrix} 0 & Cy & 0 \\ -Cy\cdot\cos\delta & 0 & sen\delta \\ Cy\cdot sen\delta & 0 & \cos\delta \end{bmatrix} \tag{10}$$

4 Python Computational Implementations Aspects

Python is an object-oriented and open source programming language often used for rapid application development. Having simple syntax, with an emphasis on readability, reduces the cost of maintaining the program, while its vast library of functions encourages reuse and extensibility. Because of these characteristics, the 3D reticular finite element formulation was implemented in Python 3.10.7 and Spyder IDE 5.4.4. The proposed algorithm can be subdivided into three phases, that is, pre-processing, processing and post-processing. In the pre-processing phase, the discretization and the physical and geometric properties of the problem are provided and then, used by the processing phase to determine the unknowns of the problem, that is, the displacements. Finally, in the post-processing phase, the nodal displacements, are used to determine the internal forces and support reactions. Numpy library 1.25.1, for optimized vector and matrices routines, was used. It is important to mention that the contributions to the global matrix are effected through the incidence rules that relate the nodes of a given element with its final position in the global system of the structure. For more details, the authors suggest to consult Weaver and Gere [6].

5 Numerical Results and Validation

Analytical and numerical examples available in the literature were exhaustively tested and the obtained results showed the good performance of the implemented tool for the tested examples. To demonstrate its good performance, three examples available in the literature are presented in this section.

5.1 Example 1: Space Frame presented by Soriano [7] (2014)

In this example, the space frame shown in Fig. 2, presented by Soriano [7], is analyzed. The space frame has three members and all member have the same cross-sectional properties, in which it's values, are all unitary (cross-section area, and inertias). The numerical values, in SI units, can be seen in Fig. 2. The support point A is fixed and a uniformly distributed load q = 20 kN/m is acting in the vertical direction on members BC and CD. For the numerical analysis, each member, that is, AB, BC and CD were discretized in one finite element. Tables 1-2 compares the solution obtained for reactions values and internal forces with the reference solution presented by Soriano [7], respectively. Analyzing the obtained results, it is possible to verify that the implemented formulation is returning consistent solutions.

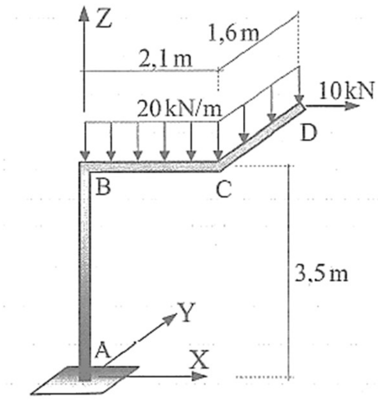

Figure 2. Space frame by Soriano [7].

Table 1. Example 1 Reactions values (R_i in kN and M_i in kN·m).

Reaction	Node A	
	Soriano (2014)	Author (2023)
R_x	-10	-10
R_y	0	0
R_z	74	74
M_x	25,6	25,6
M_y	-146,3	-146,3
M_z	16	16

Table 2. Example 1 Internal nodal forces (F_i in kN and M_i in kN·m).

		Element AB		Element BC		Element CD	
Node	Internal Force	Soriano (2014)	Author (2023)	Soriano (2014)	Author (2023)	Soriano (2014)	Author (2023)
i	F_x	74	74	-10	-10	0	0
	F_y	-10	-10	-74	-74	-32	-32
	F_z	0	0	0	0	10	10
	M_x	16	16	25,6	25,6	0	0
	M_y	25,6	25,6	-16	-16	-16	-16
	M_z	-146,3	-146,3	-111,3	-111,3	-25,6	-25,6
j	F_x	-74	-74	10	10	0	0
	F_y	10	10	32	32	0	0
	F_z	0	0	0	0	-10	-10
	M_x	-16	-16	-25,6	-25,6	0	0
	M_y	-25,6	-25,6	16	16	0	0
	M_z	111,3	111,3	0	0	0	0

5.2 Example 2: Grid Structure presented presented by Soriano [7] (2014)

The grid structure shown in Fig. 3 is in a horizontal plane (X-Y plane) and carries a uniformly distributed load q = 20 kN/m acting in the vertical direction, Soriano [7]. The grid structure has three members and all member have the same cross-sectional properties, the values being the same as the first example. The numerical values, in SI units, can be seen in Fig. 3. For the numerical analysis, each member, was discretized in one finite element. Tables 3-4 compares the solution obtained for internal forces and reaction values with the reference solution, respectively. The differences between the solutions obtained with the numerical tool and the reference solutions are practically non-existent and show the good performance of the implemented tool for the tested examples. It is important to mention that in Tables 2 and 3, the results already consider the differences between the coordinate systems adopted by the Reference and by the present formulation.

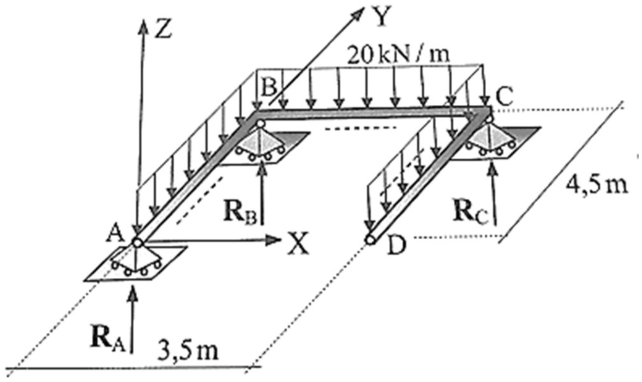

Figure 3. Grid structures presented by Soriano [7].

Table 3. Example 2 Internal nodal forces (F_i in kN and M_i in kN·m).

		Element AB		Element BC		Element CD	
Node	Internal Force	Soriano (2014)	Author (2023)	Soriano (2014)	Author (2023)	Soriano (2014)	Author (2023)
i	F_x	0	0	0	0	0	0
	F_y	0	0	0	0	0	0
	F_z	90	90	35	35	90	90
	M_x	0	0	-202,5	-202,5	0	0
	M_y	0	0	0	0	-202,5	-202,5
	M_z	0	0	0	0	0	0
j	F_x	0	0	0	0	0	0
	F_y	0	0	0	0	0	0
	F_z	0	0	-35	-35	0	0
	M_x	0	0	-202,5	-202,5	0	0
	M_y	202,5	202,5	0	0	0	0
	M_z	0	0	0	0	0	0

Table 4. Example 2 Reactions values (R_i in kN and M_i in kN·m).

	Node A		Node B		Node C	
Reaction	Soriano (2014)	Author (2023)	Soriano (2014)	Author (2023)	Soriano (2014)	Author (2023)
R_x	0	0	0	0	0	0
R_y	0	0	0	0	0	0
R_z	90	90	35	35	125	125
M_x	0	0	0	0	0	0
M_y	0	0	0	0	0	0
M_z	0	0	0	0	0	0

5.3 Example 3: Space Frame presented by Weaver and Gere [6] (1990)

In this example, the space frame shown in Fig. 4, presented by Weaver and Gere [6], is analyzed. The space frame has three members and all member have the same cross-sectional properties. The numerical values, in SI units, are: $L = 3\ m$, $A_x = 0.01\ m^2$, $I_x = 2\times10^{-3}\ m^4$, $I_y = I_z = 1\times10^{-3}\ m^4$, $E = 200\times10^6\ kN/m^2$, $G = 80\times10^6\ kN/m^2$ and $P = 60\ kN$. The support points C and D have six constraints each. For the numerical analysis, the members AC and BD were discretized in one finite element and the member AB was discretized in two finite elements. The joint loads on the frame are a 2P force in the positive x direction at point A, a P force in the negative y direction at point B and a PL moment in the negative z sense at B. A 4P force in the positive z direction is applied at the midlength of the member AB. Tables 5-7 compares the solution obtained for displacements, internal forces and reaction values with the reference solution, respectively.

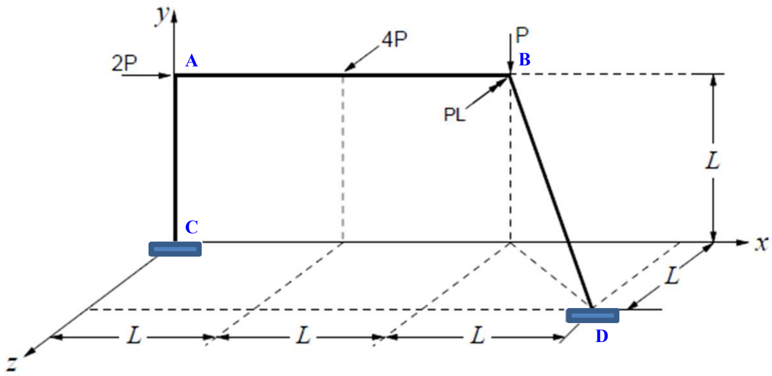

Figure 4. Space frame adapted from Weaver and Gere [6].

Table 5. Example 3 Displacements (U_i in m and θ_i in rad).

Displacement	Node A: Weaver & Gere (1990)	Node A: Author (2023)	Node B: Weaver & Gere (1990)	Node B: Author (2023)
U_x	-85,941E-05	-85,941E-05	-11,761E-04	-11,7605E-04
U_y	57,764E-06	57,764E-06	32,532E-04	32,5316E-04
U_z	50,076E-04	50,076E-04	52,555E-04	52,5552E-04
θ_x	23,933E-04	23,933E-04	12,884E-04	12,8843E-04
θ_y	-16,232E-04	-16,2317E-04	17,209E-04	17,2094E-04
θ_z	68,133E-05	68,1331E-05	-77,147E-05	-77,1469E-05

Table 6. Example 3 Internal nodal forces (F_i in kN and M_i in kN·m).

Node	Internal Force	Member AB: Weaver & Gere (1990)	Member AB: Author (2023)	Member AC: Weaver & Gere (1990)	Member AC: Author (2023)	Member BD: Weaver & Gere (1990)	Member BD: Author (2023)
i	F_x	105,548	105,548	-38,509	-38,509	183,623	183,623
	F_y	-38,509	-38,509	14,452	14,452	9,192	9,192
	F_z	-126,013	-126,013	-126,013	-126,013	5,967	5,967
	M_x	29,464	29,464	86,569	86,569	-21,404	-21,404
	M_y	86,569	86,569	348,575	348,575	46,703	46,703
	M_z	-67,1	-67,1	-23,744	-23,7438	-32,181	-32,181
j	F_x	-105,548	-105,548	38,509	38,509	-183,623	-183,623
	F_y	38,509	38,509	-14,452	-14,452	-9,192	-9,192
	F_z	-113,987	-113,987	126,013	126,013	-5,967	-5,967
	M_x	-29,464	-29,464	-86,569	-86,569	21,404	21,404
	M_y	-50,491	-50,491	29,464	29,464	-77,711	-77,711
	M_z	-163,954	-163,954	67,1	67,1	79,946	79,946

Table 7. Example 3 Reactions values (R_i in kN and M_i in kN·m).

Reaction	Node C		Node D	
	Weaver & Gere (1990)	Author (2023)	Weaver & Gere (1990)	Author (2023)
R_x	-14,452	-14,452	-105,548	-105,548
R_y	-38,509	-38,509	98,509	98,509
R_z	-126,013	-126,013	-113,987	-113,987
M_x	-348,575	-348,575	-75,898	-75,898
M_y	86,569	86,569	-75,808	-75,808
M_z	-23,744	-23,744	37,163	37,163

The differences between the solutions obtained with the numerical tool and the reference solutions are practically non-existent and show the good performance of the implemented tool for the tested examples.

6 Conclusions

A 3D finite element formulation in Python language for the reticular structural linear analysis was successfully implemented as it is possible to conclude from the obtained results. It is important to mention that in this first moment the focus was on the development of the first functions of the program, that is, the development of a tool for the analysis of 3D reticular structures in a linear elastic regime. Future works will aim to develop a graphical interface that will enable a more user-friendly environment for input data, currently done through a text file or directly in the programming environment. It concludes, with a free and open-source Python computation code, open for all to use, and such, the foundations for future engineers to further expand its functionalities is available. The source code is available in the GitHub repository under GNU v3 license. The GitHub link is: https://github.com/gCarvalhoFerreira/FEM-Python.

Acknowledgements. The authors would like to thank the State of Amazonas Research Support Foundation (FAPEAM) for the financial support granted through the Scientific Initiation Program (PAIC): 2022-2023 Edition/ SISPROJ Project Number 37452.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method: Basic Formulation and Linear Problems*, 4th Edition, vol. 1. McGraw-Hill, 1989.
[2] T.J.R. Hughes, *The Finite Element Method – linear static and dynamic finite element analysis.* Prentice-Hall International, Inc. Englewood Cliffs, New Jersey, 1987.
[3] J.T. Oden and J.N. Reddy, *An introduction to the mathematical theory of finite elements*. John Wiley & Sons, New York, 1976.
[4] J.N. Reddy, *Energy and variational methods in applied mechanics*. John Wiley & Sons, New York, 1984.
[5] R.L. Rangel and L.F. Martha, "*LESM — An object-oriented MATLAB program for structural analysis of linear element models*". Computer Applications in Engineering Education, vol. 27, n. 3, pp. 553-571, 2019.
[6] W. Weaver and J.M. Gere, *Matrix Analysis of Framed Strucutres* – 3rd Edition. New York: Van Nostrand Reinhold, 1990.
[7] H. L. Soriano, *Estática das Estruturas* – 4ª edição revisada. Rio de Janeiro: Editora Ciência Moderna Ltda., 2014.