Matrix structural analysis of beams on elastic supports: implementation in Python

Maysa A. R. Curvelo[1], Felipe A. V. Bazán[1], Jesaías S. Silva[1], Paulo C. O. Queiroz[1], George F. Azevedo[1]

[1]*Dept. of Civil Engineering, Federal University of Maranhão*
Av. dos Portugueses, 1966, Vila Bacanga, 65080-805, São Luís/MA, Brazil
maysa.ar@discente.ufma.br, felipe.vargas@ufma.br, jesaias.santos@discente.ufma.br, pco.queiroz@ufma.br, gf.azevedo@ufma.br

Abstract. Technological resources have made it possible to model natural phenomena and engineering problems more accurately, thus allowing better results. This article presents the computational implementation of a procedure for matrix structural analysis of beams. Rigid and elastic supports, and continuous elastic base were considered to represent the contact between the structural model and the external environment. Internal hinges are also addressed. A computational code was created in Python. The code calculates stiffness matrices and force vectors of beam elements, which are assembled into a global stiffness matrix and a global force vector, and the equilibrium equation system is then solved. Results such as nodal displacements and slopes, support reactions, and internal forces are obtained. It is shown that the presence of elastic supports and elastic foundations can significantly affect the overall structural behavior of the system. By incorporating these effects into the analysis, more accurate predictions of the structural response are achieved, leading to safer and more economical solutions. The article provides a basis for further investigation of the behavior of complex structural systems. Python was chosen as it provides a versatile and efficient platform for conducting numerical analyses and is suitable for advances in structural engineering.

Keywords: beam element, elastic support, matrix structural analysis, finite element method, Python.

1 Introduction

It is of crucial importance for the development of civil construction that the modeling of problems is more aligned with real events, so that mistakes are avoided and the consumption of raw materials and labor is reduced, resulting in cleaner and more efficient constructions. In current times, the search for excellence has become intense, due to economic, social, and environmental factors. Technological advancement is crucial to guarantee continuous improvement in comparison to the use of older methods. The use of computational tools makes it easier to perform complex calculations, such as those ones involved in numerical methods to solve engineering problems. With this in mind, this paper proposes the analysis of beam structural models using the matrix structural analysis method [1] along with a computer program written in Python [2] programming language. Different cases such as beams on a continuous elastic base and beams with internal hinges are considered. This allows us to obtain more accurate results and more realistic models considering physical phenomena such as the variability of soil stiffness.

2 Matrix analysis of beam elements

The concepts of matrix analysis of structures were used to develop the code, discretizing the structural beam elements into smaller elements connected by nodes. The elements are analyzed separately and considered as

characteristics of the materials and applied actions. Each element will have its element stiffness matrix [K_{el}], eq. (1), elastic base stiffness matrix [K_{be}], elastic bond contribution stiffness matrix [K_{ve}], element equivalent force vector [F_{el}], eq. (2), and point load vector [F_P], composing the global stiffness matrix [K_G] and global force vector [F_G] that represent the structure as a whole. To arrive at the global matrix and vector, the addressing is done through $2i - 1$; $2i$; $2j - 1$; $2j$; where i is the initial node and j is the final node. The corresponding row and column values in the element matrices are added to the global matrix, the same is done with the rows in the element and global force vectors, point forces are considered only in a single element. Since these are beam elements, they have only two degrees of freedom, which represent the moment and the shear, hence the matrices having four rows and four columns [3].

$$[K_e] = \begin{bmatrix} \frac{12\,E\,I}{L^3} & \frac{6\,E\,I}{L^2} & -\frac{12\,E\,I}{L^3} & \frac{6\,E\,I}{L^2} \\ \frac{6\,E\,I}{L^2} & \frac{4\,E\,I}{L} & -\frac{6\,E\,I}{L^2} & \frac{2\,E\,I}{L} \\ -\frac{12\,E\,I}{L^3} & -\frac{6\,E\,I}{L^2} & \frac{12\,E\,I}{L^3} & -\frac{6\,E\,I}{L^2} \\ \frac{6\,E\,I}{L^2} & \frac{2\,E\,I}{L} & -\frac{6\,E\,I}{L^2} & \frac{4\,E\,I}{L} \end{bmatrix} \tag{1}$$

$$\{F_e\} = \begin{Bmatrix} \frac{7}{20}\,L\,q_i + \frac{3}{20}\,L\,q_j \\ \frac{1}{20}\,L^2\,q_i + \frac{1}{30}\,L^2\,q_j \\ \frac{3}{20}\,L\,q_i + \frac{7}{20}\,L\,q_j \\ -\frac{1}{30}\,L^2\,q_i - \frac{1}{20}\,L^2\,q_j \end{Bmatrix} \tag{2}$$

The boundary conditions in this work are based on the "zeros and ones technique" [1,4], where the row and column representing the constraints are set to zero, with the unitary value only on the main diagonal. The nodal displacements are verified using the notation found in eq. (3). This formulation is based on Hooke's Law and through its manipulation, the values of the internal forces and support reactions [5] are also found.

$$[K_G]\{U_G\} = \{F_G\} \tag{3}$$

In analyzing elements with internal hinges, the matrices used for calculations undergo variations. The stiffness matrices can be found in Martha [1], for the elastic base matrices and force vectors the deductions were made by the authors.

3 Methodology

3.1 Creation

Code was developed using a text editor called Brackets [6]. The input file is in JSON format [7] and the entire program reads input data from this file. The information indicated in the input file is divided into three blocks: the first block contains the number of elements and the number of nodes; the second one, information about each element; and the third one, information about each node. The required element information is the element number, its initial and final nodes, the modulus of elasticity, the moment of inertia, the initial and final values of the distributed load, the stiffness coefficient of the elastic base, if it exists, and, in the case of hinged element, the specification about the position of the hinge. The required node information is the node number; nodal coordinates; constraints (vertical, and rotational); applied loads (vertical, and moment); and stiffness coefficients of the elastic supports (vertical, and rotational).

There is an outer counting loop that reads and extracts the information contained in each node and stores it

in a dictionary. This information is stored from the entire analyzed structure. The storage is done according to the variable type and placed in the dictionary using the same addressing scheme of matrices and vectors, with their keys also in string format. A second, inner counting runs through the elements, subsequently calculating the matrices and force vectors associated with each element. From there, the program executes the following steps: obtaining the global stiffness matrix; obtaining the global force vector; contribution of forces and specific moments in the nodes; contribution of elastic supports; boundary conditions; calculation of nodal displacements; calculation of internal forces; calculation of rigid support reactions; calculation of elastic support reactions; creation of the output file in text format.

3.2 Execution

To perform the calculation of the stiffness matrix and force vector of the element, the program first checks the existence of a hinge in the element and its location. Accordingly, the program calculates the stiffness matrix of the appropriate element. If an elastic base exists, the elastic base matrix is added to the element stiffness matrix. The program will execute this function for each existing element and store the results in dictionaries (whose key is the index in the global stiffness matrix) and lists. The calculation of the element force vector also varies according to the existence or not of a hinge and is done together in this first step.

After obtaining stiffness matrices and force vectors of all elements, the global stiffness matrix and the global force vector are obtained from the created addressing keys, adding values at equal keys. The program then adds the initially stored information about the nodes, applied loads, and elastic supports. Next, the program organizes the matrix in list format, so that the boundary conditions are applied. Before the application of the boundary conditions, copies of the lists are made, which is required for future calculations. Finally, the global displacement vector is calculated by solving eq. (4).

To calculate the internal forces, the individual values of the stiffness matrices and force vectors of each element are accessed in the lists and dictionaries where they were stored individually, and the formula is manipulated based on the results obtained from the displacements. The computation of fixed support reactions also follows the formula mapping, using the values from the matrix and global force vector without any constraints. The elastic support reactions are obtained by multiplying the elastic support constant by the corresponding displacement. All results obtained are added to a formatted text file, and implemented in the final part of the program.

4 Application and Results

The tests were carried out with different beam structures, using all implemented resources, such as elastic supports, elastic bases, and hinges. Fig. 1 shows an example of a beam on an elastic base. The example was adapted from Fernandes and Correia [8]. These authors considered discrete elastic supports, whereas a continuous elastic base is considered in the present work. The input data was deduced from those used by the mentioned work in order to make a consistent comparison. The results obtained, with discretization into 10 elements, are shown in Tab. 1, comparable to the reference results.

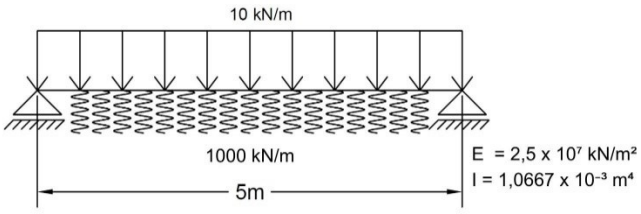

Figure 1. Example of a beam on an elastic base reproduced in AutoCAD [9]

Table 1. Results obtained by the authors with the program created for a beam on an elastic base

Nodal displacements			Reactions of rigid supports	
Node	Vertical (m)	Rotation (rad)	Vertical (kN)	Moment (kN.m)
1	0	-1.579700e-03	21.06299	0
2	-7.743561e-04	-1.488629e-03	0	0
3	-1.463197e-03	-1.244850e-03	0	0
4	-2.000709e-03	-8.899912e-04	0	0
5	-2.341145e-03	-4.628089e-04	0	0
6	-2.457586e-03	0	0	0
7	-2.341145e-03	4.628089e-04	0	0
8	-2.000709e-03	8.899912e-04	0	0
9	-1.463197e-03	1.244850e-03	0	0
10	-7.743561e-04	1.488629e-03	0	0
11	0	1.579700e-03	21.06299	0

Internal forces				
Element	QI (kN)	QF (kN)	MI (kN.m)	MF (kN.m)
1	21.06299	16.258476	0	9.314204
2	16.258476	11.822943	9.314204	16.320185
3	11.822943	7.696312	16.320185	21.188785
4	7.696312	3.790675	21.188785	24.05343
5	3.790675	0	24.05343	24.99867
6	0	-3.790675	24.99867	24.05343
7	-3.790675	-7.696312	24.05343	21.188785
8	-7.696312	-11.822943	21.188785	16.320185
9	-11.822943	-16.258476	16.320185	9.314204
10	-16.258476	-21.06299	9.314204	0

Figure 2 illustrates an example consisting of a Gerber beam model, and the results are shown in Tab 2. This example was taken from Machado Júnior [10] and the results obtained with the developed program were the same as the reference results.

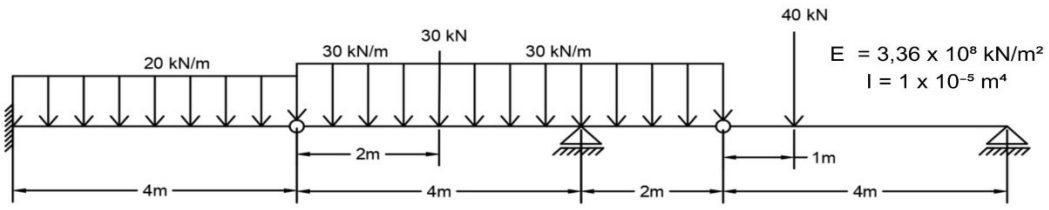

Figure 2. Gerber beam example reproduced in AutoCAD [9]

Table 2. Results obtained by the authors for Gerber beam with the program created

Nodal displacements			Reactions of rigid supports	
Node	Vertical (m)	Rotation (rad)	Vertical (kN)	Moment (kN.m)
1	0	0	125	340
2	-4.761905e-01	1.101190e-01	0	0
3	-2.440476e-01	1.250000e-01	0	0
4	0	1.041667e-01	195	0
5	1.666667e-01	7.440476e-02	0	0
6	1.160714e-01	-4.761905e-02	0	0
7	0	-3.422619e-02	10	0

Internal forces				
Element	QI (kN)	QF (kN)	MI (kN.m)	MF (kN.m)
1	125	45	-340	0
2	45	-15	0	30
3	-45	-105	30	-120
4	90	30	-120	0

5	30	30	0	30
6	-10	-10	30	0

5 Conclusions

The program is based on the study of FEM in a more introductory way, facilitating the understanding of the subject for people with little contact with the numerical analysis method, since when working with elements of a single dimension the concepts become easier to understand. The use of the Python language, due to its high level, facilitates the development of codes by people without in-depth knowledge of programming. It also has vast libraries that facilitate implementation.

Based on the references and results obtained, the program meets the needs for which it was designed. Regarding future developments, the objectives are the automated discretization of the structural model, as well as making the input file more intuitive for the user.

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] L. F. Martha. *Análise de estruturas: conceitos e métodos básicos*. Elsevier, 2010.
[2] G. Van Rossum and F. L. Drake, "Python reference manual". Centrum voor Wiskunde en Informatica Amsterdam, 1995.
[3] J. B. Paiva. *Introdução ao método dos elementos finitos*. EESC/University of São Paulo, 2012.
[4] H. L. Soriano. *Análise de estruturas: formulações clássicas*. Livraria da Física, 2016.
[5] A. Alves Filho. *Elementos Finitos: a base da tecnologia CAE*. Érica, 2013.
[6] Brackets Software, Adobe Systems Incorporated, 2023.
[7] JavaScript Object Notation, available at https://www.json.org/json-en.html accessed on 17 May 2023.
[8] A. V. B. Fernandes and V. C. Correia, "Análise de vigas sobre base elástica considerando a interação solo-estrutura", *Cadernos de Graduação: Ciências Exatas e Tecnológicas*, vol. 6, n. 1, pp. 71-92, 2020.
[9] AutoCAD® autodesk®, 2024.
[10] E. F. Machado Junior. *Introdução à isostática*. EESC/University of São Paulo, 2012.