Effects of meshing density of 1D structural members with non-uniform cross-section along the length on the calculation of eigenfrequencies

Jakub Rubint[1,*]

[1]Slovak Technical University in Bratislava, Faculty of Civil Engineering, Department of Structural Mechanics, Radlinského 11, 810 05 Bratislava, Slovak Republic

Abstract. Density of division in finite element method does not affect only the accuracy of calculation, but also the necessary calculation time. This is directly influenced by the power of the used hardware, the efficiency of the algorithm used to assemble global stiffness and mass matrices and finally, by the method used to find eigenvalues of matrices for determination of eigenfrequencies. In engineering practice, when commercially available software is used, it is necessary to look for the optimum between the accuracy of the calculation and the length of the calculation. This paper deals with solution of eigenfrequencies of 1D elements with nonuniform cross section using Python 3.7.4 with libraries "numpy 1.18.1" for finding eigenvalues of matrices and "scipy 1.4.1" for finding solution for system of nonlinear equations.

1 Introduction

In finite element analysis, choosing mesh density is the finding optimum between accuracy of solution and complexity of model. In some of the simplest static analysis tasks, it is possible to effectively use the coarsest meshing. More complicated static analysis tasks require a finer meshing of the structure. The issue is discussed in more detail in the articles [1] and [2]. The influence of the meshing density of the structure on the results of modal analysis is discussed in the article [2]. Following chapters describe creation of a finite element method program in Python 3.7.4, analyse influence of dividing density on accuracy of calculation of the eigenfrequencies of beam with non-uniform cross-section. Program was designed to be able simply increasing number of finite elements in discretization of the simply supported beam using a single variable. To verify correctness of the algorithm of composing the global stiffness and mass matrices, the results were compared with an analytical solution of the calculation of the eigenfrequencies of the simple beam.

* Corresponding author: jakub.rubint@stuba.sk

2 Programming of FEM

As it was mentioned in the introduction, for the purpose of analysis in the environment of Python, the FEM program was created. Subsequently, the "eigenvals" commands were called to obtain a vector of eigenvalues whose elements, after square rooting and dividing by τ represent the individual eigenfrequencies of the structure. To verify the functionality of algorithm which assembles global matrices of task, numerical calculations were compared with results of the analytical solution.

2.1 Principles of FEM

The first basic step of finite element analysis of structures is the discretization of the structure to finite elements. Since the task was to monitor the effect of division on the accuracy of the calculation.

The next step is to approximate the deformation variables on each finite element separately. We start from the stiffness matrix of the finite element of the beam (Fig. 1) with two degrees of freedom in both nodes.

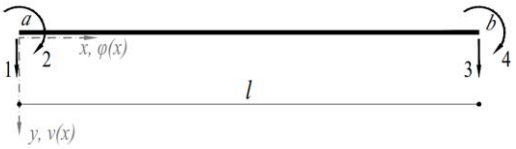

Fig. 1. Node parameters - numbering and sign convention.

As it is shown in Fig. 1 in this case, chosen finite element has two nodes with three unknown deformations in each of them, thus stiffness matrices of elements (1) and their mass matrices (2) as well are 4x4 dimension [3]

$$\mathbf{k}_\mathrm{i} = \frac{EI_i}{{L_i}^3}\begin{bmatrix} 12 & 6L_i & -12 & 6L_i \\ 6L_i & 4{L_i}^2 & -6L_i & 2{L_i}^2 \\ -12 & -6 & 12 & -6L_i \\ 6L_i & 2{L_i}^2 & -6L_i & 4{L_i}^2 \end{bmatrix} \tag{1}$$

where: i is the variable indexing sequence number of the finite element [-]
E is Young modulus of elasticity [Pa]
$\mathbf{k}_\mathrm{i}$ is the stiffness matrix of the i^{th} element
L_i is the length of the i^{th} element [m]
I_i is the second moment of area [m^4]

$$\mathbf{m}_\mathrm{i} = \frac{\mu_i L_i}{420}\begin{bmatrix} 156 & 22L_i & 54 & -13L_i \\ 22L_i & 4{L_i}^2 & 13L_i & -3{L_i}^2 \\ 54 & 13L_i & 156 & -22L_i \\ -13L_i & -3{L_i}^2 & -22L_i & 4{L_i}^2 \end{bmatrix} \tag{2}$$

where: $\mathbf{m}_\mathrm{i}$ is the mass matrix of the i^{th} element
μ_i is the mass per unit length of the i^{th} element

Next step is assembling global matrices of stiffness and mass from element matrices, respecting interconnections between individual members. Procedure of assembling global matrices is well known and it is identical for both, global stiffness matrix and global mass matrix. Scheme of the task is shown in Fig. 2 where the beam is divided into 4 finite elements.

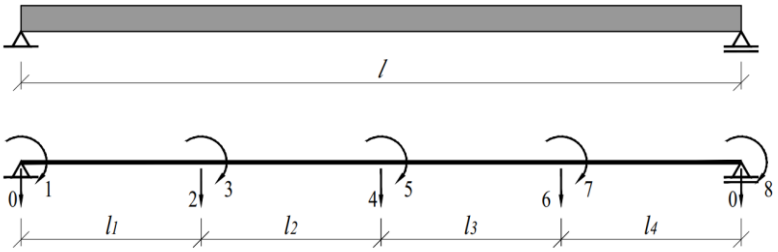

Fig. 2. Scheme of the 4-element simple beam.

To obtain the eigenfrequencies of the structure, an eigenvalue function is called. Eigenvalue function in Python [5] is based on Linear Algebra Package, originally written in FORTRAN 77. In Python code it is necessary to import "numpy" library first. Then it is possible to call function for computing eigenvalues.

3 Eigenfrequencies of a simple beam

For purpose of verifying the functionality of the program, results of the numerical calculation method were compared with the results of the analytical solution. A simple beam, with uniform cross-section along its length and uniformly distributed mass, was solved according to the following scheme (Fig. 3).

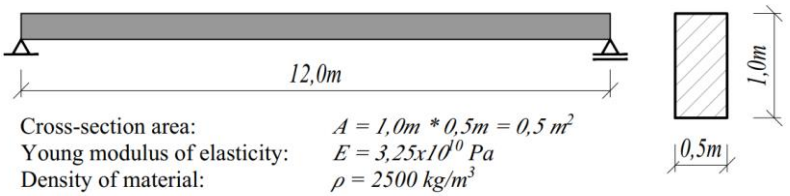

Fig. 3. Scheme of the solved simple beam.

3.1 Analytical solution

Based on [4], the analytical solution for a simple beam with a uniformly distributed mass is based on a partial differential equation describing the transverse damped beam oscillation with the uniformly distributed mass.

$$\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2 v(x,t)}{\partial x^2}\right) + \mu(x)\frac{\partial^2 v(x,t)}{\partial t^2} = 0 \tag{3}$$

where:
- $v(x,t)$ is the function of deflection
- E is the Young modulus of elasticity [Pa]
- $\mu(x)$ is the function of distribution of mass along the length [kg/m']
- $I(x)$ is the function of second moment of area along the length [m^4]

In this case of a simple beam, the cross-section of the beam is uniform, $\mu(x) = \mu$ and $I(x) = I$. Differential equation (3) can be simplified to form

$$\frac{\partial^4 v(x,t)}{\partial x^4} + \frac{\mu}{EI}\frac{\partial^2 v(x,t)}{\partial t^2} = 0 \tag{4}$$

By substituting the boundary conditions of the simple beam into the solution of equation (3), an analytical relation (5) for the calculation of the natural vibration frequencies of the simple prismatic beam is obtained.

$$f_n = \frac{1}{2\pi}\sqrt{\frac{n^4\pi^4 EI}{\mu l^4}} \tag{5}$$

where: n is integer indexing the n^{th} eigenfrequency
l is the span of a simple beam [m]

3.2 Numerical solution

The calculation of the eigenfrequencies of the simple beam was performed on the models with gradual increase in the number of finite elements of the given structure. Specifically, the beam was divided in range of finite elements from 4 to 1000. Results of numerical solution quickly converged to results of analytical solution, thus only results of division up to 100 finite elements are relevant.

Table 1. Results of analytical and numerical solutions.

Modal shape	Analytical solution	Numerical solution				
		4 FE	20 FE	100 FE	500 FE	1000 FE
#	Eigenfrequencies [Hz]					
1	11.3537	11.3566	11.3537	11.3537	11.3536	11.3547
2	45.4149	45.5941	45.4152	45.4149	45.4149	45.4149
3	102.1835	104.0506	102.1870	102.1835	102.1835	102.1835
4	181.6596	201.6272	181.6790	181.6596	181.6596	181.6596
5	283.8431	320.4857	283.9168	283.8432	283.8431	283.8431

3.3 Results comparison

The following chart (Fig. 4) shows the deviation of the numerical solution from the analytical solution for individual eigenmodes. The rapid convergence to the analytical solution in increasing the number of finite elements is evident. Coarse mesh in this relatively simple task is effective for obtaining results of first few eigenfrequencies. The graph shows a clear dependence - the higher the eigenmode shape, the higher the number of elements needed.

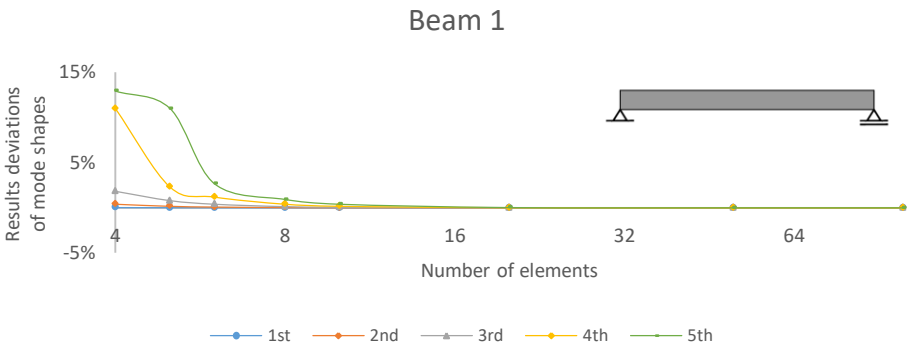

Fig. 4. Deviation of results of numerical solution compared to analytical solution.

Results of numerical solution quickly converge to results of analytical solution. Division of beam to more than 100 finite elements is practically noneffective, thus unnecessary.

4 Eigenfrequencies of non-uniform beams

Following subchapters deal with solutions of simply supported beams with variable cross-section height along its length. All beams have the same span of 12 m. Since exact solution of equation (3) is known only for prismatic simple beam, the numerical finite element method was used for the analysis of the beams. Again, the same finite element divisions were used as in Chapter 3.2. This time the results could not be compared with the calculated eigenfrequencies, thus values when beam was divided to 100 finite elements were used as a reference.

For beams with variable height of cross-section along its length, the function of height must be defined. The cross-sectional characteristics and uniform mass divided into nodes are calculated based on the average cross-sectional height between nodes of finite element. For each type of variable cross-sectional height, two beams with different steepness of the function defining height along its length were considered. The Young modulus of elasticity with value of 32.5 GPa was considered in all cases.

4.1 Variable height with the shape of a linear function

Variable cross-section height was defined by function (6) in case of beam 2a and by function (7) in case of beam 2b.

$$h(x) = 30^{-1}x + 0.8 \tag{6}$$

$$h(x) = 15^{-1}x + 0.8 \tag{7}$$

In the following charts (Fig. 5) dependence of results deviations on meshing density of first five eigenfrequencies are shown.

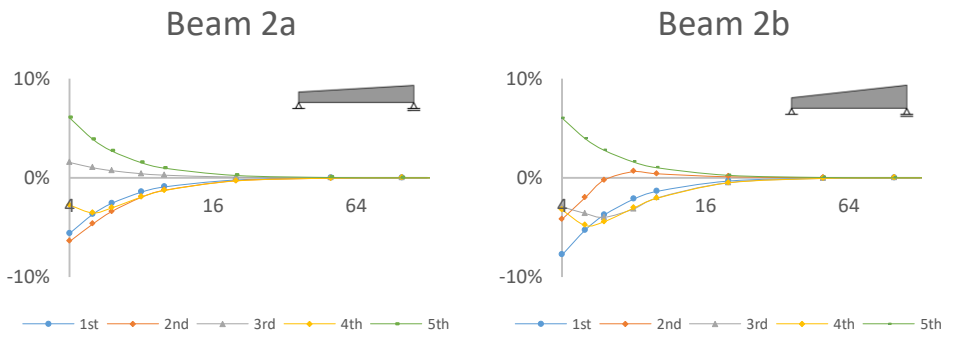

Fig. 5. Deviation of results depending on the division density of beams 2a and 2b.

4.2 Variable height with the shape of a quadratic function

Variable cross-section height was defined by function (8) in case of beam 3a and by function (9) in case of beam 3b. Results are shown in the following charts (Fig. 6).

$$h(x) = \sqrt{623.1338 - (x - 6.4038)^2} - 23.3273 \quad (8)$$

$$h(x) = \sqrt{273.5985 - (x - 6.2568)^2} - 14.5118 \quad (9)$$

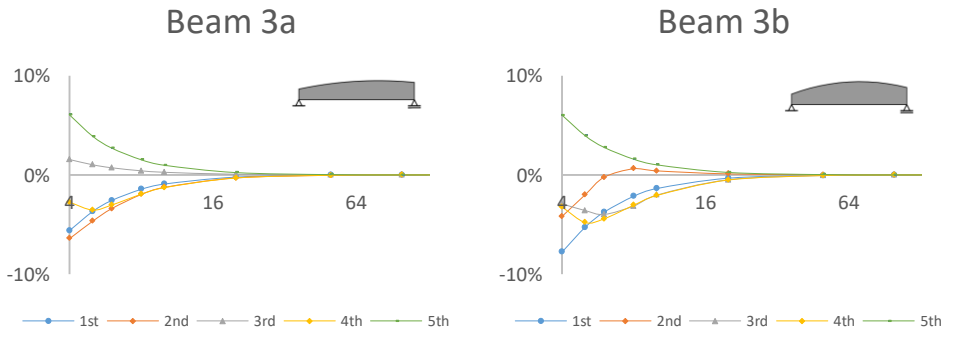

Fig. 6. Deviation of results depending on the division density of beams 3a and 3b.

4.3 Variable height with the shape of a cubic function

Variable cross-section height was defined by function (10) in case of beam 4a and by function (11) in case of beam 4b. Again, results are shown in the following charts (Fig. 7).

$$h(x) = 0.002604x^3 - 0.047x^2 + 0.196x + 0.7 \quad (10)$$

$$h(x) = 0.00651x^3 - 0.119x^2 + 0.496x + 0.9 \quad (11)$$

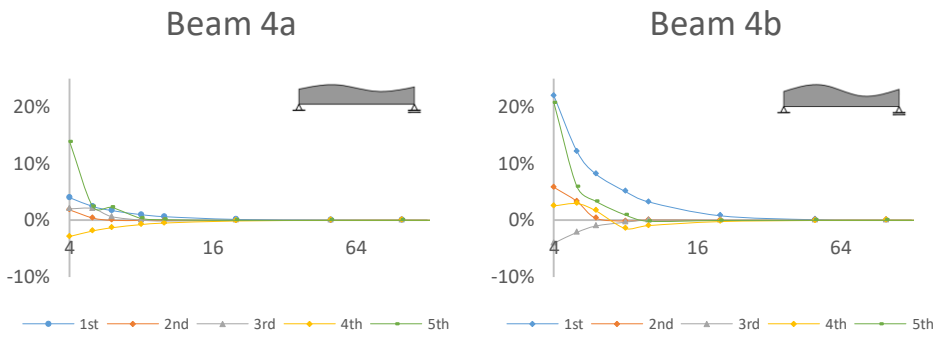

Fig. 7. Deviation of results depending on the division density of beams 4a and 4b.

5 Conclusion

As it was mentioned in the introduction, computational time is, among the others, affected by the complexity of algorithm which assembles global matrices. In the case of the third-degree complexity of the algorithm, the computational time increases cubically with the number of finite elements. As it can be seen in the graphs in Chapter 4, increasing the dividing density of beams with variable cross-section is only effective to some extent. From these graphs it can also be seen that the steepness of the function describing the cross-sectional height along the length of the beam influences the accuracy of the calculation. It can be stated that the division into 20 finite elements proved to be sufficiently precise and at the same time effective with respect to the necessary calculation time.

This paper was written with the support of Slovak Grant Agency VEGA 1/0412/18 and KEGA 025STU-4/2019.

References

1. Dutt, Aman, Effect of Mesh Size on Finite Element Analysis of Beam. SSRG -IJME. 2. 10.14445/23488360/IJME-V2I12P102 (2015)
2. Liu, Yucheng, Effects of Mesh Density on Finite Element Analysis. SAE Technical Papers. 2. 10.4271/2013-01-1375 (2013)
3. P. Marton, *Dynamics of Structures (in Slovak)* (STU Bratislava, 2008)
4. M. Sokol, K. Tvrdá, *Dynamics of Structures (in Slovak)* (STU Bratislava, 2016)
5. https://github.com/numpy/numpy/blob/v1.17.0/numpy/linalg/linalg.py#L1324-L1459