


An Educational GUI-Based Software for Dynamic Analysis of Framed Structural Models

Claudio H. B. Resende¹, Pedro C. Lopes², Rafael L. Rangel³, Luis F. Paulo Muñoz¹,
Luiz F. Martha^{1*} 

¹Department of Civil and Environmental Engineering, Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil

²Institute of Computing, Fluminense Federal University, Niterói, Brazil

³International Centre for Numerical Methods in Engineering (CIMNE), Universitat Politècnica de Catalunya, Barcelona, Spain

Email: *lfm@tecgraf.puc-rio.br

How to cite this paper: Resende, C.H.B., Lopes, P.C., Rangel, R.L., Muñoz, L.F.P. and Martha, L.F. (2023) An Educational GUI-Based Software for Dynamic Analysis of Framed Structural Models. *Journal of Software Engineering and Applications*, 16, 265-286.

<https://doi.org/10.4236/jsea.2023.167014>

Received: June 7, 2023

Accepted: July 23, 2023

Published: July 26, 2023

Copyright © 2023 by author(s) and Scientific Research Publishing Inc. This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

Abstract

This paper introduces a new version of the open-source educational software, LESM (Linear Elements Structure Model), developed in MATLAB for structural analysis of one-dimensional models such as frames, trusses, and grillages. The updated program includes dynamic analysis, which incorporates inertial and damping effects, time-dependent load conditions, and a transient solver with multiple time integration schemes. The software assumes small displacements and linear-elastic material behavior. The paper briefly explains the theoretical basis for these developments and the reorganization of the source code using Object-Oriented Programming (OOP). The updated Graphical User Interface (GUI) allows interactive use of dynamic analysis features and displays new results such as animations, envelope diagrams of internal forces, phase portraits, and the response of degrees-of-freedom in time and frequency domain. The new version was used in a structural dynamics course, and new assignments were elaborated to improve students' understanding of the subject.

Keywords

Structural Analysis, Structural Dynamics, Educational Software, MATLAB, Graphical User Interface

1. Introduction

The teaching of structural analysis has faced a dilemma in recent decades: how to balance the use of manual methods and computer programs. It is known that structural mechanics courses that are based on theoretical instruction can benefit greatly from the use of software, especially from those educational-oriented,

as it keeps the course up to date with nowadays resources and, at the same time, more attractive to students. These benefits are potentially even greater when graphical-interactive resources are available. The employment of computer programs can complement the theory by helping students to: model and simulate the behavior of real-world structures, besides simple academic problems; get a physical interpretation of the governing equations by observing how the mechanical system behaves; acknowledge the sensitivity of the system to the input parameters; visualize the results and learn to better criticize them. Moreover, in hands-on programming-involving scenarios, access to the source code can provide a unique experience for students, which goes beyond the use of the software as a “black-box”. This approach allows students to work on the implementation of analysis methods, which is easier when the code is written in a high-level programming language, and it is modular and well documented.

This paper describes the extension of a structural analysis software, LESM (Linear Elements Structure Model), to include dynamic analysis. The goal of this development was to complement a graduate course on structural dynamics at the Department of Civil and Environmental Engineering of the Pontifical Catholic University of Rio de Janeiro (PUC-Rio). This paper also intends to show how the course was adapted to incorporate the use of the software as an educational tool and how such use took place to enhance students’ understanding of the subject.

LESM is an educational project developed at PUC-Rio. The program is open-source, written in the MATLAB script language, and it was initially designed to serve as a complementary tool for undergraduate and graduate Engineering courses at PUC-Rio. The first version of the program was conceived as a non-graphical tool for static linear-elastic analysis of models composed of linear elements, *i.e.* prismatic uniaxial elements with one dimension much larger than the others, such as bars (axial behavior only) and beams (axial, flexural, and torsion behaviors). These models are 2D/3D frames, 2D/3D trusses, and grillages. The focus was to provide a didactic source code to introduce students to the implementation of matrix structural analysis methods [1]. The second version incorporated a user-friendly Graphical User Interface (GUI) with sophisticated mouse modelling capabilities [2], among some new analysis features [3]. That version was introduced in a course of computer graphics for engineering and also increased the usability of the software, allowing not only students but also professional engineers to use the program for more practical purposes, but still limited to static analysis.

The consideration of dynamic effects to analyze the vibration of structures was an immediate demand. This type of analysis is relevant in many engineering applications, and there are only a few educational-oriented programs for this purpose with graphical-interactive features and modelling freedom, *i.e.* geometry and degrees-of-freedom (DOFs) are not fixed to pre-defined templates. **Table 1** provides a list of software developed for educational purposes with dynamic analysis capabilities and graphical resources. Part of these programs are specialized in earthquake engineering [4]-[11] and are often intended to serve as

virtual laboratories. Many of them are restricted to simple predefined models, such as single degree-of-freedom (SDOF) and multiple degrees-of-freedom (MDOF) systems [4] [5] [6] [11] [12] [13] [14] [15], or frames defined by parametric templates [7] [8] [9] [10]. Among those that allow free modelling of frames, some rely on non-graphical inputs [16] [17] [18] [19] [20] and only a few have an interactive interface for efficient pre- and post-processing [21] [22] [23] [24]. However, even for the latter group, the models are usually restricted to 2D and the dynamic analysis options are very limited. In addition, open-source is not a common feature of these softwares. Therefore, the third version of LESM is devoted to filling these gaps.

Table 1. Educational software with dynamic structural analysis capabilities and graphical resources.

Software name	Platform	Open source	Models	Reference
Abel	MATLAB	Yes	3D frame templates	[10]
CAL/CGI	FORTTRAN	Yes	2D frames	[16] [17] [25]
CALSB	Java	No	2D frames	[21]
CALSIDOF	Java	No	SDOF systems	[15]
DINEST	DELPHI	No	SDOF systems	[12]
Dynasoft	Java	Yes	SDOF/MDOF systems, 2D/3D frame templates	[9]
Unnamed	Mathematica	No	SDOF systems	[26]
DYSSOLVE	MATLAB	Yes	SDOF systems, 2D frames	[27]
ENGLTHA	Visual Basic	No	SDOF systems	[11]
Frame of Mind	Java/Flash/ FORTRAN	No	2D frames	[22]
Frame3DD	C	Yes	2D/3D frames	[20]
Ftool	C	No	2D frames	[24]
LAS	Visual C#	Yes	2D frames	[18]
MASTAN2	MATLAB	No	2D/3D frames	[23]
NDOF	MATLAB	No	MDOF systems	[13]
NONLIN	Visual Basic	No	SDOF/MDOF systems	[4]
SDET	Java	No	SDOF systems	[14]
Stabil	MATLAB	Yes	2D/3D frames, Continuous FEM models	[19]
Structural Control Virtual Laboratories	Java	No	2D frame templates	[7] [8]
VSDL	Visual Basic/ Java	No	SDOF/MDOF systems	[5] [6]

In the new version of LESM, inertial and damping forces of bar and beam elements are considered in different ways according to the selected formulation for the mass and damping matrices. Regarding flexural behavior, both Euler-Bernoulli and Timoshenko theories are available. Different solution methodologies can be employed for the transient problem: a numerical time integration with one of the implemented implicit or explicit schemes, or an analytical solution of the homogeneous equation for free vibration through a modal decomposition. Initial conditions, load conditions subjected to customizable time functions, and concentrated masses can be applied to free DOFs. In all cases, small displacements and linear-elastic material behavior are assumed. The available results are animations of the transient response or vibration modes, envelope diagrams of internal forces, phase portraits, and the response of DOFs in time and frequency domain.

These developments took advantage of some of the main features initially proposed for the program. One of them is the code modularity provided by Object-Oriented Programming (OOP). In fact, this work put to the test the extensibility of the analysis module as idealized in its creation [1], and the result is very satisfactory, as only a few modifications to the original code were required, and most of the dynamic analysis features were placed in new classes. Another one is the graphical interactivity provided by MATLAB, as explained in [2]. The dynamic analysis requires much more graphical resources than a simple static analysis since users may want to see animations of the vibrating structures and plots of the transient response. Furthermore, the amount of inputs and outputs is larger, which requires an intuitive GUI to avoid the pre and post-processing becoming confusing. The installation files and the complete source code of the latest release version of the program are available on its website¹. The project is also hosted in a public Git repository².

The remainder of this paper is organized as follows. Section 2 explains the theoretical considerations of structural dynamics behind the new developments. Section 3 describes the new data structure of the source code of the analysis module, focusing on the organization of the OOP classes. Section 4 shows the new graphical interface features related to dynamic analysis. Section 5 demonstrates how the program was applied in an academic context. The paper ends with concluding remarks and discusses future developments in Section 6.

2. Theoretical Considerations

In dynamic analysis, inertial and damping forces are considered, in addition to the elastic forces considered in static analysis. In this case, the response of the discretized structural model, expressed as vectors of nodal displacements and rotations (\mathbf{x}), velocities ($\dot{\mathbf{x}}$), and accelerations ($\ddot{\mathbf{x}}$), as well as external loads (\mathbf{f}), are time (t) dependent. The system of dynamic equilibrium equations is giv-

¹<https://web.tecgraf.puc-rio.br/lesm>

²<https://gitlab.com/rafaelrangel/lesm>

en in Equation (1), where \mathbf{M} , \mathbf{C} , and \mathbf{K} are the mass, damping and stiffness matrices, respectively.

$$\mathbf{M}\ddot{\mathbf{x}}(t) + \mathbf{C}\dot{\mathbf{x}}(t) + \mathbf{K}\mathbf{x}(t) = \mathbf{f}(t) \quad (1)$$

2.1. System Assembly

The global mass, damping and stiffness matrices of the structure are assembled with contributions from the local matrices of each element. The local stiffness matrix available in the program considers Euler-Bernoulli and Timoshenko theories. Details about the element stiffness matrix formulation can be found in the previous work [1]. The formulation of the local mass matrix and the global damping matrix, as well as the assembly of the array of nodal loads, are briefly discussed in the sequence.

2.1.1. Mass Matrix

Two formulation types are commonly employed for the local mass matrix of structural elements. They are the lumped and consistent formulation. Both options are available in LESM. The lumped formulation is based on the simplification that the total element mass is equally divided into two concentrated masses at the end nodes. This assumption leads to a diagonal local matrix, which can significantly reduce the computational cost in several numerical integration schemes. Moreover, in LESM, concentrated masses contribute only to translational inertia, which relates to DOFs of displacements, not rotations. On the other hand, the consistent formulation assumes that the element mass is distributed in accordance with the shape functions that describe the displacement and rotation fields. Hence, the local mass matrix, \mathbf{M}_e , is computed through Equation (2), where ρ is the material density, A is the cross-section area, L is the element length, and \mathbf{N} is the vector of shape functions. In this case, the local mass matrix is relative not only to translational, but also to rotational DOFs. The result of the integral expression of Equation (2) for a generic 3D beam element, written in such a way that it works for both Euler-Bernoulli and Timoshenko theories, is presented in the **Appendix**. The matrix coefficients of a generic 3D beam are filtered into the element matrix according to the selected model type, as explained in [1].

$$\mathbf{M}_e = \rho A \int_0^L \mathbf{N}^T \mathbf{N} dx \quad (2)$$

Alternatively, customized mass matrix [28] is also available in LESM. In this approach, the local mass matrix is a linear combination of lumped and consistent formulations, as shown in Equation (3), where μ is a coefficient of proportionality that must be provided by the user so that $0 \leq \mu \leq 1$.

$$\mathbf{M}_e = (1 - \mu) \mathbf{M}_e^{\text{lump}} + \mu \mathbf{M}_e^{\text{consist}} \quad (3)$$

2.1.2. Damping Matrix

In LESM, a classical Rayleigh damping is assumed. In this case, the global

damping matrix is considered as a linear combination of the global mass and stiffness matrices, as expressed in Equation (4), where α and β are coefficients of proportionality. These coefficients can be directly informed by users, or computed from the critical damping factors of the first and second vibrations modes (ξ_1 and ξ_2) according to Equation (5), where ω_1 and ω_2 are the natural angular frequencies associated to modes 1 and 2, respectively. In this case, the critical damping factors must be informed by the user.

$$\mathbf{C} = \alpha \mathbf{M} + \beta \mathbf{K} \quad (4)$$

$$\begin{Bmatrix} \alpha \\ \beta \end{Bmatrix} = \frac{1}{2} \begin{bmatrix} 1/\omega_1 & \omega_1 \\ 1/\omega_2 & \omega_2 \end{bmatrix}^{-1} \begin{Bmatrix} \xi_1 \\ \xi_2 \end{Bmatrix} \quad (5)$$

2.1.3. External Load Vector

External loads can be concentrated forces and moments applied to nodes or distributed forces applied along elements. These load components are assembled into the vector of external loads. In the case of dynamic analysis, the components of the load vector may vary over time. In LESM, only nodal loads are allowed to be time-dependent. A load matrix (\mathbf{F}) is assembled with the load vectors at each time step, as stated in Equation (6). Vector \mathbf{a} stores user-specified amplitudes of the nodal loads, and vector \mathbf{y} holds discrete values of dimensionless time functions that depict how each load varies over time. Note that, as we are mostly interested in not very large problems, memory is not a major concern, thus it is no issue to store \mathbf{F} as a matrix. However, it is straightforward to adapt the solvers to compute a force vector on the fly at each time step to spare RAM usage, should it be a pressing matter in an eventual large simulation.

$$\mathbf{F}(t) = \mathbf{a} \mathbf{y}^T(t) \quad (6)$$

2.2. System Solution

To obtain the solution of the system of dynamic equilibrium equations, LESM makes available two methodologies: numerical time integration schemes, and analytical solutions for free vibration via modal decomposition. Both are briefly described in the sequence.

2.2.1. Numerical Integration

LESM provides different solution algorithms for numerical time integration with both explicit and implicit schemes. Currently, there are four options: Newmark, Wilson- θ , three-step Adams-Moulton, and 4th order Runge-Kutta. Given the appropriate initial conditions and time step size, these algorithms calculate the vectors of nodal displacements/rotations, velocities, and accelerations at each time step. However, with this approach it is not possible to decouple the influence of each vibration mode on the dynamic response.

2.2.2. Modal Decomposition

This methodology allows the identification of significant vibration modes and

their contributions. This is done by decoupling the system of equations, through a basis transformation to modal space, and solving analytically each resultant homogeneous differential equation for free vibration.

Firstly, it is necessary to obtain the natural frequencies and modes of vibration. To do this, undamped and free vibrating conditions are applied to Equation (1), leading to the generalized eigenvalue problem of Equation (7). It yields a solution pair (ω_i^2, φ_i) for each of the n vibration modes, where n is the number of free DOFs, φ_i is the i^{th} eigenvector, which represents the normalized nodal displacements/rotations of the i^{th} vibration mode, and the i^{th} eigenvalue, ω_i^2 , is the squared natural frequency associated to φ_i . The solution of natural vibration frequencies and modes is provided by LESM.

$$\mathbf{K} - \omega^2 \mathbf{M} \boldsymbol{\varphi} = 0 \quad (7)$$

To solve the transient problem by modal decomposition, the system of dynamic equilibrium equations in Equation (1) is transformed as in Equation (8), where Φ is the modal matrix, whose columns are composed of the eigenvectors φ_p and Ω^2 is the spectral matrix, whose diagonal coefficients are composed of the eigenvalues ω_i^2 . Equation (8) represents an uncoupled system, where the resultant ordinary differential equations can be solved individually even by analytical methods for $\mathbf{f}(t) = 0$. Here, each entry of the solution vector represents the time-dependent contribution of a specific vibration mode to the movement of the whole structure. Nodal displacements and rotations (and their derivatives) are computed as an interpolation of the eigenvectors with each of their contributions [29].

$$\ddot{\mathbf{x}}(t) + [\Phi^T \mathbf{C} \Phi] \dot{\mathbf{x}}(t) + \Omega^2 \mathbf{x}(t) = \Phi^T \mathbf{f}(t) \quad (8)$$

3. Data Structure

The analysis module of the program, responsible for the computation of the results, adopts the Object-Oriented Programming (OOP) paradigm. It provides modularity to the code and improves its extensibility. This is crucial in programming-involving scenarios for teaching structural analysis methods, as students can work on independent OOP classes without affecting the rest of the program. The implementation of the dynamic analysis resources also followed the OOP approach. Accordingly, only small modifications were needed to the original code, as most of the dynamic analysis capabilities were placed in new classes. In order to provide a general overview of the new code organization, **Figure 1** presents the Class Diagram of the program following the UML (Unified Modelling Language) format [30]. The UML is used to formally illustrate the design of an object-oriented code through diagrams. Specifically, the Class Diagram depicts the relationship between instantiated objects of the program's classes, as well as inheritance details about concrete subclasses that implement behaviors laid out by abstract superclasses. The diagram in **Figure 1** also highlights, in grey, the newly added classes with respect to the previous versions, whose diagram was provided in [1]. The purpose of these new classes is described below:

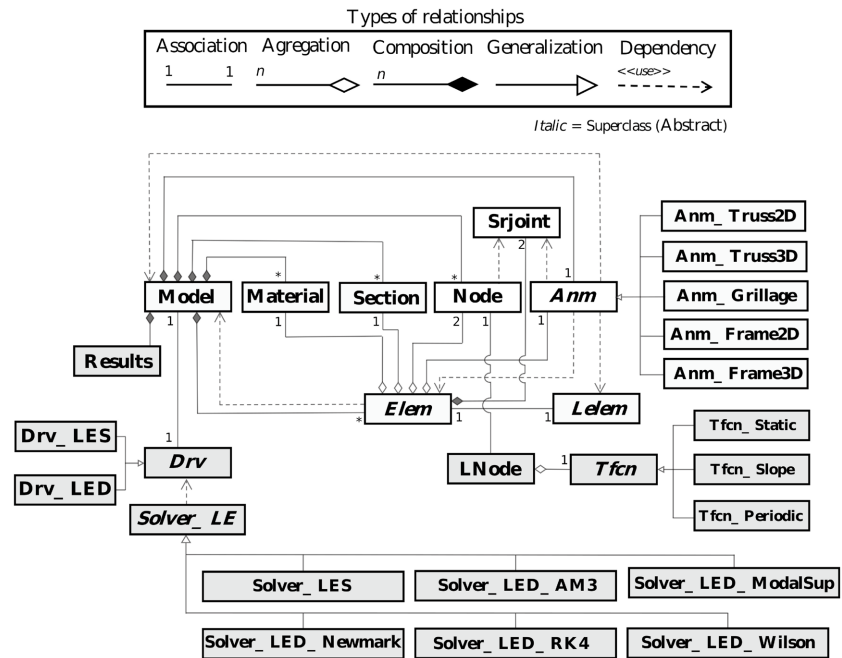


Figure 1. UML class diagram of LESM highlighting the new classes for implementing dynamic analysis.

- *Model*: An object of this class represents the structural model by storing its global properties and pointing to the objects that represent its components. This class was previously called *Drv* and also included the methods for performing linear-elastic static analysis.

- *Drv* (Driver): Superclass responsible for declaring the methods that drive the analysis process, most of them related to the assembly of the system of equations; an object must be instantiated from one of its subclasses, which contain the implementations for linear-elastic static (*Drv_LES*) and dynamic (*Drv_LED*) analysis.

- *Solver_LE*: Superclass responsible for declaring the methods that deal specifically with the solution of the system of equilibrium equations of linear-elastic analyses (currently, the only available option). An object must be instantiated from one of its subclasses, which contain the implementation of the solution for static analysis (*Solver_LES*) or dynamic analysis according to the available algorithms: Modal Superposition (*Solver_LED_ModalSup*), Newmark (*Solver_LED_Newmark*), Wilson- θ (*Solver_LED_Wilson*), Adams-Moulton (*Solver_LED_AM3*), and Runge-Kutta (*Solver_LED_RK4*).

- *Tfcn* (Time Function): Superclass that defines time functions, *i.e.* the evolution on time of a given parameter; an object must be instantiated from one of its subclasses, which contain the implementations for constant (*Tfcn_Static*), linear (*Tfcn_Slope*), and harmonic (*Tfcn_Periodic*) functions.

- *Lnode* (Nodal Load): An object of this class stores the properties of the loads applied to a specific node, including the time function that describes their evolution.

- *Results*: An object of this class is responsible for storing the results as the analysis advances on time.

4. Graphical User Interface

To accommodate the new analysis features, the GUI of LESM went through some modifications with respect to previous versions [2]. It includes the addition of auxiliary dialogs for setting dynamic analysis options, creating time functions, and plotting graphs. New result options were also added to display animations and envelope diagrams of internal forces. Moreover, small changes were made to the layout, in order to improve its organization and keep it simple and intuitive to be used by inexperienced students or engineers. **Figure 2** shows the main interface of LESM highlighting new buttons added to open auxiliary dialogs related to dynamic analysis and a demo 3D frame model.

The dialog for setting dynamic analysis options is shown in **Figure 3**. It allows users to opt for purely modal analysis (natural vibration frequencies and modes) or to include the transient response. In the case of transient analysis, the solution algorithm must be selected, as one of the previously described methodologies, together with the time discretization. The mass matrix formulation and damping coefficients must also be informed. In addition, initial conditions of displacement/rotation and velocity can be prescribed to free DOFs, and they are displayed in a table.

The creation and edition of time functions is done through the dialog shown in **Figure 4**. For each time function created, users can add several components from predefined function types (constant, linear, or harmonic) or from an imported table of values. These added components are superimposed to form the desired time function. The created time functions can then be assigned to nodal loads in the dialog of nodal load insertion (not shown). The load components provided are multiplied by the value of the selected time function at each time step. In the dialog of nodal load insertion, it is also possible to add a concentrated mass to each node.

After dynamic analysis is performed, it is possible to plot graphs for specific results with the resources made available in the dialog shown in **Figure 5**. A specific DOF is selected to show its displacement, velocity and acceleration history throughout the simulation, while a phase portrait diagram provides these three responses plotted against each other. It is also possible to visualize results in the frequency domain, computed with MATLAB's built-in FFT function. In addition, the results can be filtered to show only the response due to free or forced vibration, besides the total response as the sum of both. Furthermore, if the solution algorithm for the analysis was set as modal superposition, each vibration mode can be displayed individually, so that their individual contributions can be studied. All these data can be exported as images, text files, or spreadsheets.

Finally, in the results panel of the main dialog (which can be seen in **Figure 2**), where users can select which result type they want to display in the model,

new options were added for dynamic analysis. These are the animation of the structure's motion or a particular vibration mode, and also envelope diagrams of internal forces. Controls for speed and amplitude of motion are available for the animations. The envelope diagrams inform the extreme values throughout the transient analysis of the selected internal force at discrete positions within each element.

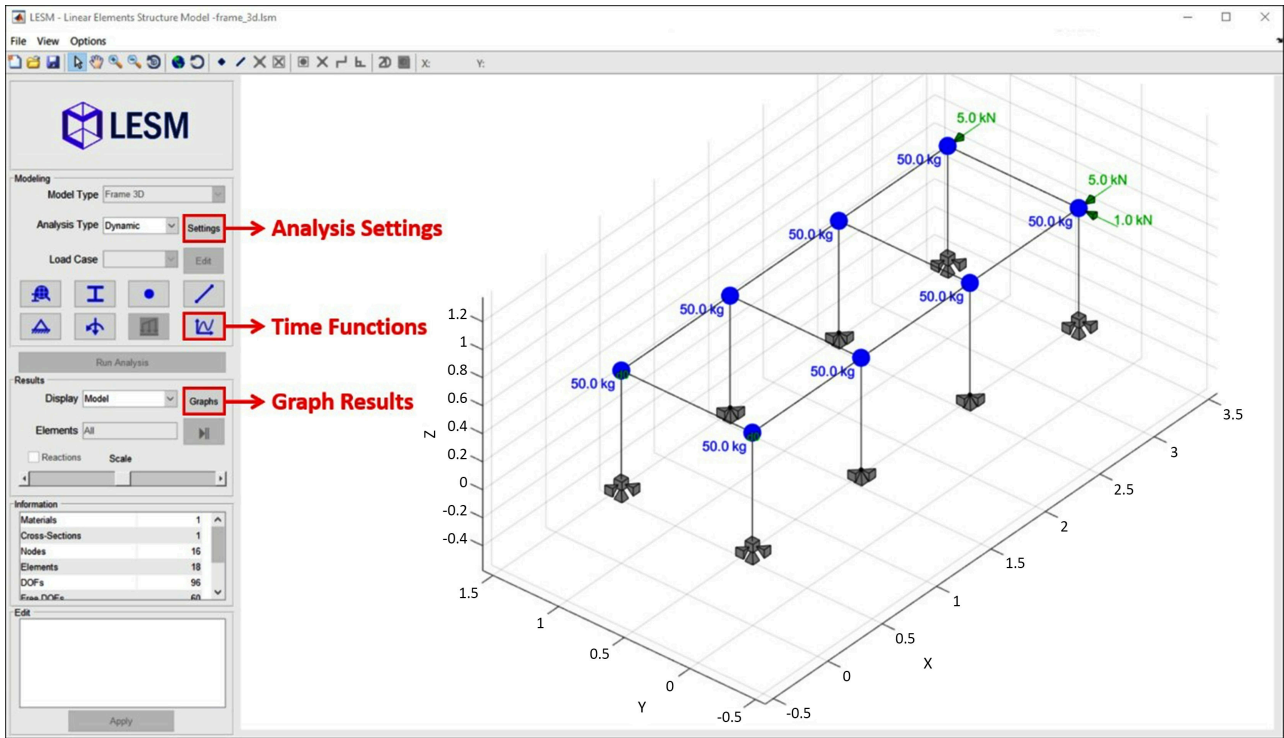


Figure 2. Main interface of LESM highlighting new options for dynamic analysis.

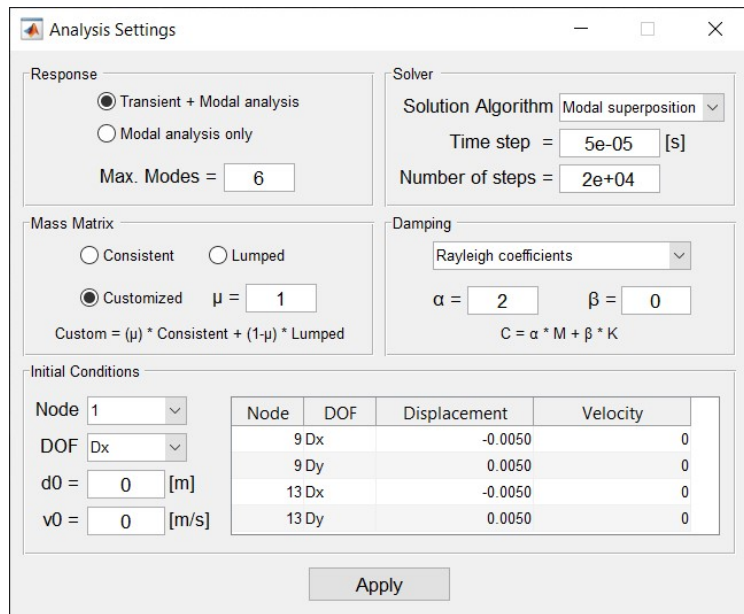


Figure 3. Auxiliary dialog for setting dynamic analysis options.

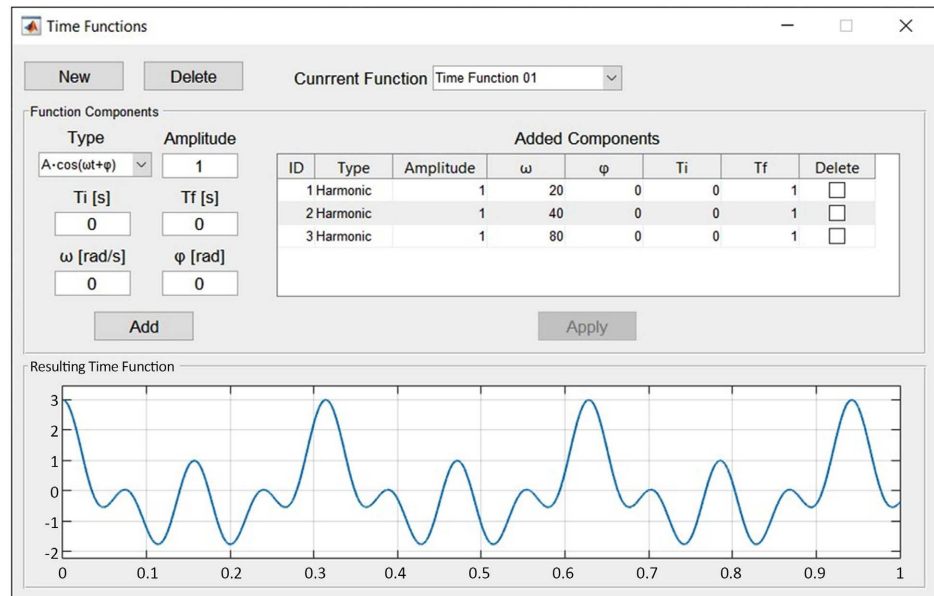


Figure 4. Auxiliary dialog for creating time functions.

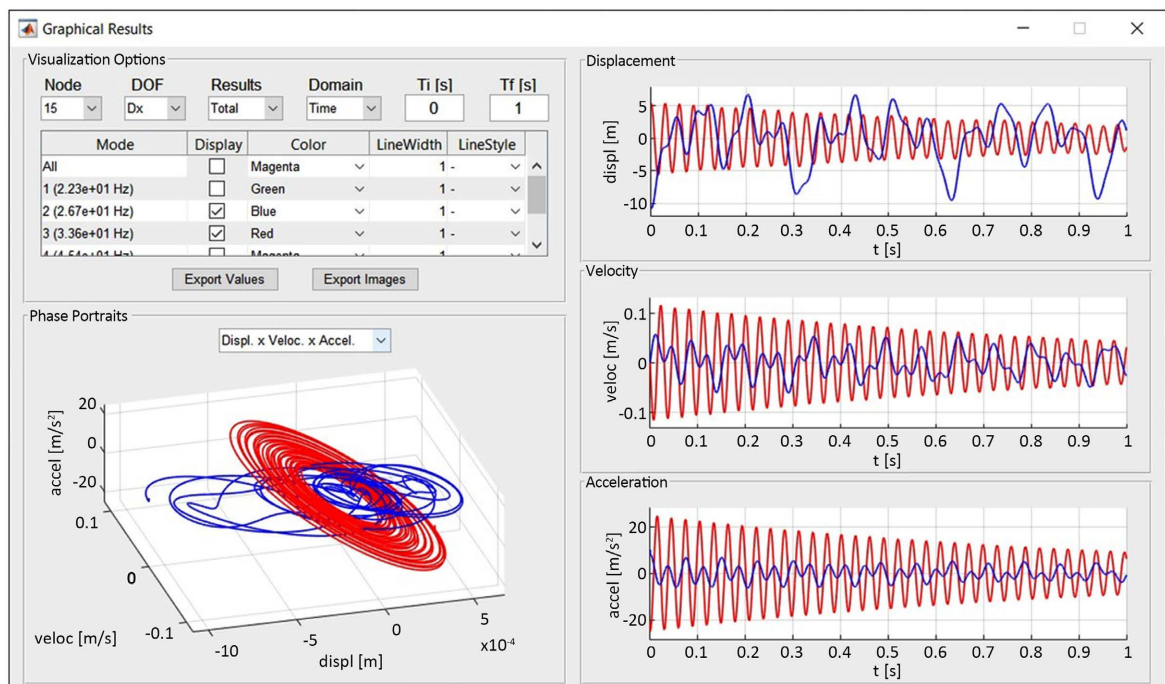


Figure 5. Auxiliary dialog for plotting graph results.

5. Examples of Application

In 2021 and 2022, LESM was introduced as an educational tool in the course “Dynamic of Structures—Part I”, which is part of the curriculum of the Civil Engineering Graduate Program at PUC-Rio. The course is offered in the first semester of each academic year and has a total duration of 45 hours, distributed in 15 weekly classes. Although there is no specific requirement to enroll in the course, most attendees are in the first year of their master’s studies and have no

experience in structural dynamics. Therefore, it addresses the basics of structural dynamics, by dividing the subject into three parts. The first and second parts involve, respectively, the study of SDOF and MDOF systems idealized as mass-spring-dashpot models, while the third part deals with continuous systems focusing on frame and truss models. Each part covers the derivation of the equations of motion, and the study of free and forced, and damped and undamped, vibrations.

The incorporation of LESM into the course aims to allow students to:

I) Become familiar with the use of software for dynamic analysis of structures with resources similar to what they would find in commercial programs during professional activities.

II) Compare the results obtained manually from simplified mass-spring-dashpot models with those provided by the program for continuous frame models.

III) Perform complete analyses of large structures.

IV) Investigate dynamic properties of structural systems, such as damping and mass distribution, by changing model parameters.

V) Explore the effects of using different numerical integration methods and get some calibration experience in terms of stability and efficiency.

To introduce LESM to students, a one-hour demonstration was held after the basic concepts of structural dynamics were presented. The homework assignments were updated so that LESM can be used as exemplified in the following sections. The theoretical exams remained the same as in previous years. However, it was observed that the average score, considering the two exams, increased by 6% in the courses in which LESM was used (8.4/10.0 with standard deviation of 1.14) compared to the previous 5 years (7.9/10.0 with standard deviation of 1.08). Although sampling of students who have used the program is still not statistically significant (13 students in total), and the atypical online format of the course in 2021 due to COVID-19 restrictions, this result can be interpreted as an indication of the increased interest of students in the subject.

5.1. Simple Academic Frame

The first assignment using LESM was to allow students to practice its use, in terms of modelling and verifying results. It aims to fulfil objective I, as described above. Students were required to simulate academic examples found in textbooks and compare the results. One of the examples is a 2D frame from [31] (**Figure 6**). It consists of two 2.54-meter long Euler-Bernoulli beam elements, one of them inclined at 45° . The material has a Young's modulus of 68,948 MPa and a density of 7,480,000 kg/m³, whilst the cross-sections have an area of 38.7 cm² and a moment of inertia of 4162 cm⁴. The joint node is subjected to a 445 kN horizontal force that is suddenly applied, *i.e.* a Heaviside step function. The other two nodes are completely fixed. The system is undamped and the mass matrix is assembled with the consistent formulation. The envelope diagrams of internal forces are shown in **Figure 7**. The transient response of the joint node is given in **Figure 8**, comparing the reference results with those obtained with LESM.

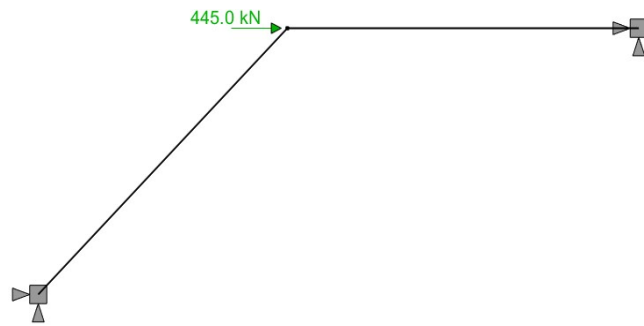
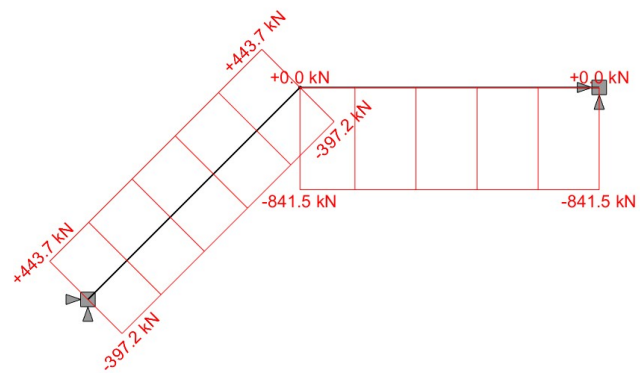
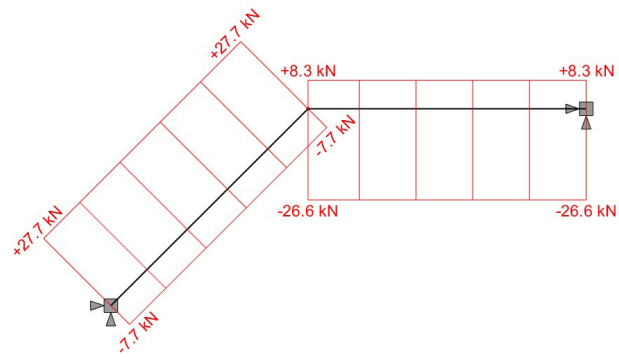


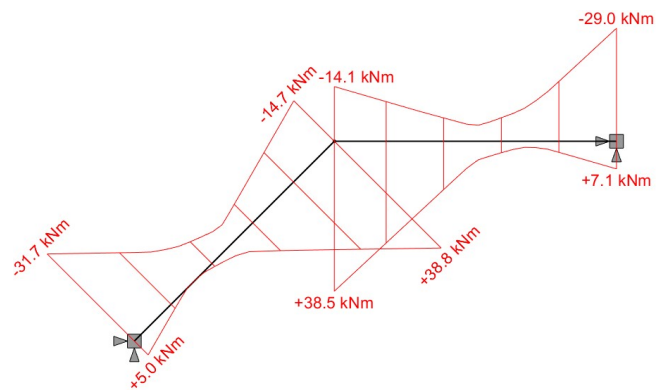
Figure 6. Frame 2D model [31] to introduce LESM to students.



(a)



(b)



(c)

Figure 7. Envelope diagrams of (a) axial force, (b) shear force, and (c) bending moment.

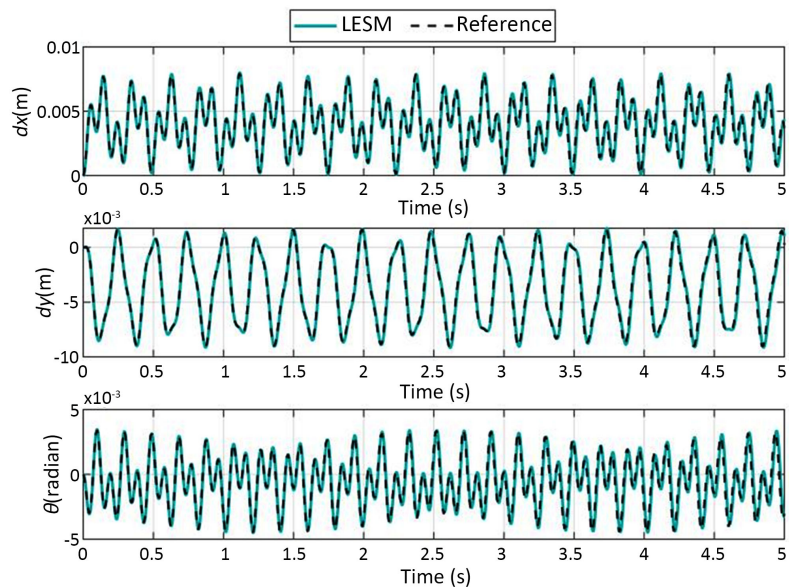


Figure 8. Time evolution of the horizontal displacement (dx), vertical displacement (dy), and rotation (θ) of joint node.

5.2. Shear Building

It is common practice to simplify structural models by reducing the number of DOFs, usually by considering that some elements are inextensible or infinitely rigid. The behavior of these simplified models can be reproduced with good accuracy by mass-spring-dashpot systems. Therefore, to fulfill the aforementioned objective II, the results of frame and truss models obtained with LESM were compared to equivalent mass-spring-dashpot systems, whose equations were manually deduced by the students. A typical example is the multi-story shear building, a frame model commonly used in earthquake engineering to study the lateral motion of buildings subjected to seismic loads. The model assumes that the horizontal beams are rigid elements and the columns are inextensible. As a consequence, the nodes present no rotation or vertical displacement, and the horizontal displacement of the two nodes on each floor is the same. Therefore, the simplified model has only one DOF per story, which is the horizontal displacement of the floor. It can be simulated as a mass-spring system in series, where the spring stiffness is the flexural stiffness of the two columns of each story.

The shear building model considered in this example (**Figure 9(a)**) has three stories, each one with height and width of 3 m. The base nodes are completely fixed. All other nodes have the vertical displacement constrained to account for the inextensibility of columns, and a concentrated mass of 5000 kg. The material of the horizontal beams has a very high Young's modulus of 10^9 MPa to simulate rigid elements, while 10^5 MPa is considered for the material of the columns. A density of 8000 kg/m^3 is assigned to the material of all elements. All cross-sections have an area of 20 cm^2 and a moment of inertia of 1500 cm^4 . All elements are Euler-Bernoulli type and their masses are neglected. The first three

modes and their respective frequencies of natural vibration are given in **Figures 9(b)-(d)**. Furthermore, to perform a transient analysis, initial conditions were given to the horizontal displacements of nodes: 3 mm to the two nodes of the third floor, 2 mm to the mid-floor, and 1 mm to the first floor. The results provided by LESM are shown in **Figure 10**. The results of the modal and transient analyses from LESM were compared with those obtained from the equivalent mass-spring system. The students proceeded with a parametric analysis to report the effects of mass distribution on the system.

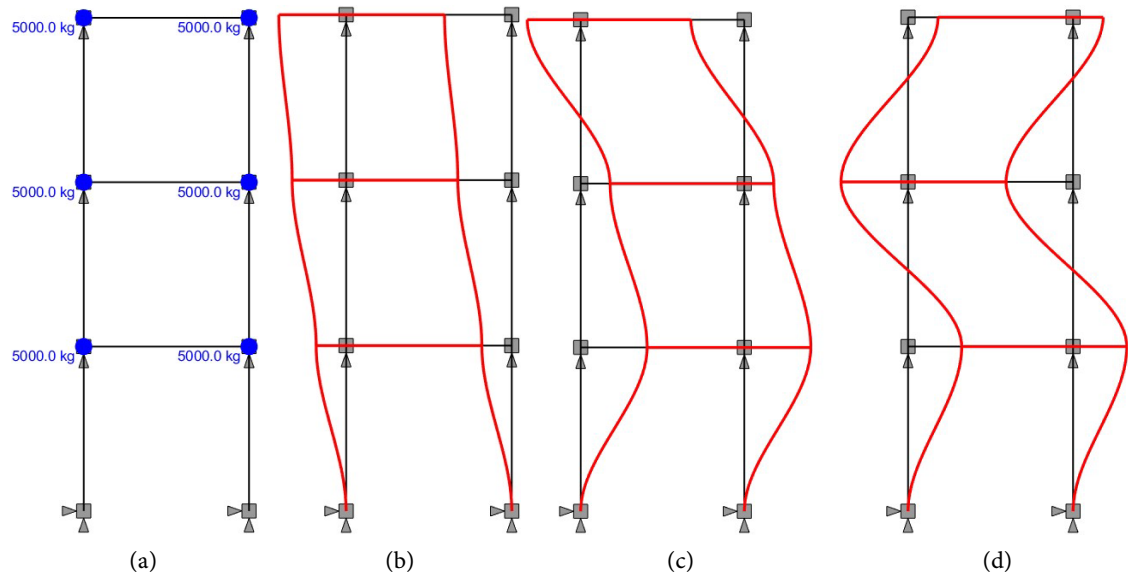


Figure 9. Shear building showing (a) model and natural vibration modes and frequencies: (b) mode 1 - 1.15 Hz, (c) mode 2 - 3.22 Hz, and (d) mode 3 - 4.66 Hz.

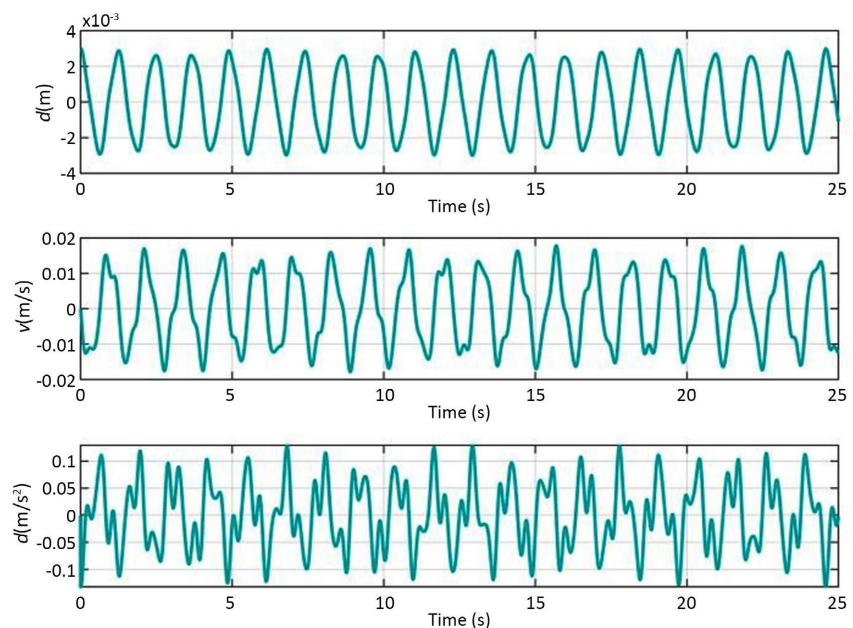


Figure 10. Time evolution of the displacement (d), velocity (v) and acceleration (a) of the horizontal motion of top right node.

5.3. Truss Bridge

Simple academic models are useful for the demonstration of theoretical concepts. However, they are not sufficient to prepare engineering students for the situations encountered in professional life. Therefore, LESM was used to allow students to perform complete analyses of more complex structures, such as bridges, buildings, and transmission towers subjected to live and wind loads. An example is the trussed bridge model presented in **Figure 11**. This 3D truss is made up of 206 bar elements hinged at their ends. The adopted material has a Young's modulus of 200 GPa and a density of 7850 kg/m³, and the cross-sections have an area of 36 cm². The nodes of the upper chords are subjected to forces with an amplitude of 10 kN. The time functions for the application of these forces replicate a load train moving across the bridge, as depicted in **Figure 12**. Other types of load conditions were also simulated, such as explosions and the lateral impact of a boat. Students were asked to prepare reports on modal contributions under different loads and bracing configurations, as well as assessing the effects of damping.

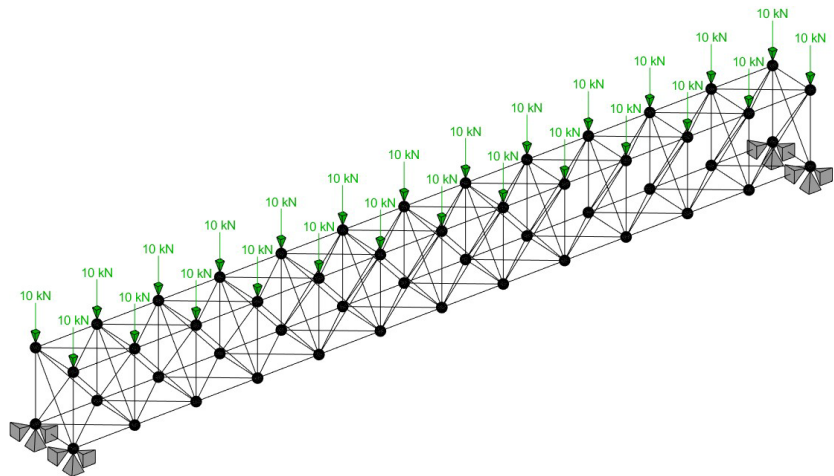


Figure 11. 3D truss model of a bridge subjected to live load.

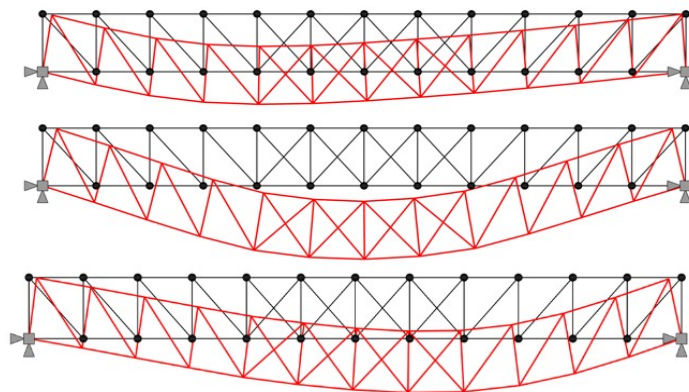


Figure 12. Deformed configurations of the trussed bridge at different times of the passage of a load train from left to right.

6. Conclusions

This paper presented a new version of LESM that includes several features to study the vibration of framed structural models, keeping the simplicity and intuitiveness of its GUI. The result is a structural analysis software that, to the best of the authors' knowledge, is the only with educational purposes, graphical-interactive resources, free-modelling capabilities, dynamic analysis of 2D and 3D frames and trusses, while being open-source. It started to be used as an educational tool in an introductory course of structural dynamics. The software could be added to the scope of the course without major changes to its program, but with small adaptations, which include a demonstration of its use and the modification of some homework assignments. The adoption of LESM during classes also enhanced the demonstration of basic theoretical concepts. In addition, the better performance of students on exams since LESM was adopted indicates that their interest in the subject has increased. Due to the successful use of LESM at PUC-Rio, partner institutions have already shown interest in adopting the program in courses related to dynamics of structures.

It is important to mention that, although commercial software allows students to perform the same types of analyses as LESM, having an in-house educational software has several advantages besides easiness of use and price. For example, the software can be customized according to the needs of the course, and include options that would not make sense in a commercial software, such as inefficient or sub-optimal solution algorithms in order to allow students to make comparisons of results and get insights into numerical stability. A useful implementation that is on-hold in LESM is the possibility of assigning a time function to prescribed support displacements to simulated seismic loads. Also, it is intended to formally consider rigid and inextensible elements instead of resorting to the workarounds presented in this work. Furthermore, as reported in this paper, the students of structural dynamics used LESM as users, not developers. Therefore, it is planned that, in future versions of the course, students will practice the implementation of dynamic analysis methods in an open-source environment using version-control systems, such as Git. It is an authors' hope that practices like this may also expand the interest of students in software development, even for those with little prior programming experience. Among other efforts that are considered for future work that go beyond structural dynamics in LESM, the addition of stability and nonlinear analyses are next objectives. These developments would enable LESM to be used in advanced courses of structural analysis.

Acknowledgements

This study was funded in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES)—Finance Code 001, and the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)—Grant 308884/2021-3.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

- [1] Rangel, R.L. and Martha, L.F. (2019) LESM—An Object-Oriented Matlab Program for Structural Analysis of Linear Element Models. *Computer Applications in Engineering Education*, **27**, 553-571. <https://doi.org/10.1002/cae.22097>
- [2] Lopes, P.C., Rangel, R.L. and Martha, L.F. (2021) An Interactive User Interface for a Structural Analysis Software Using Computer Graphics Techniques in MATLAB. *Computer Applications in Engineering Education*, **29**, 1505-1525. <https://doi.org/10.1002/cae.22406>
- [3] Marques, I.R., Lopes, P.C., Rangel, R.L. and Martha, L.F. (2019). Implementação de conexão semirrígida em modelos reticulados no contexto da programação orientada a objetos. *Proceedings of the XL Ibero-Latin-American Congress on Computational Methods in Engineering (CILAMCE)*, Natal, 11-14 November 2019. <https://cilamce.com.br/anais/arearestrita/apresentacoes/101/6551.pdf>
- [4] Charney, A.F. and Barngrover, B. (2004) NONLIN: Software for Earthquake Engineering Education. In: Blandford, G.E., Ed., *Structures 2004: Building on the Past, Securing the Future*, American Society of Civil Engineers, Reston, 1-12. [https://doi.org/10.1061/40700\(2004\)177](https://doi.org/10.1061/40700(2004)177)
- [5] Kumar, A., Babu, B.R., and Ramancharla, P.K. (2005) Virtual Structural Dynamics Laboratory. https://www.researchgate.net/profile/Pradeep-Ramancharla-2/publication/265656531_VIRTUAL_STRUCTURAL_DYNAMICS_LABORATORY/links/54c6f5510cf238bb7d0a1877/VIRTUAL-STRUCTURAL-DYNAMICS-LABORATORY.pdf
- [6] Munipala, A., Pasupuleti, A.D.K. and Ramancharla, P.K. (2012) Structural Dynamics Virtual Laboratory: A Learning Tool Kit for Young Engineers and Practicing Professionals. *Proceeding of 15th World Conference on Earthquake Engineering*, Lisbon, 24-28 September 2012. https://web2py.iit.ac.in/research_centres/publications/download/inproceedings.pdf_82d9ca034c80a240.57434545323031325f333230322e706466.pdf
- [7] Gao, Y., Yang, G., Spencer Jr., B.F. and Lee, G.C. (2005) Java-Powered Virtual Laboratories for Earthquake Engineering Education. *Computer Applications in Engineering Education*, **13**, 200-212. <https://doi.org/10.1002/cae.20050>
- [8] Sim, S.H., Spencer Jr., B.F. and Lee, G.C. (2009) Virtual Laboratory for Experimental Structural Dynamics. *Computer Applications in Engineering Education*, **17**, 80-88. <https://doi.org/10.1002/cae.20162>
- [9] Panagiotopoulos, C.G. and Manolis, G.D. (2016) A Web-Based Educational Software for Structural Dynamics. *Computer Applications in Engineering Education*, **24**, 599-614. <https://doi.org/10.1002/cae.21735>
- [10] Katsanos, E.I., Taskari, O.N. and Sextos, A.G. (2014) A Matlab-Based Educational Tool for the Seismic Design of Flexibly Supported RC Buildings. *Computer Applications in Engineering Education*, **22**, 442-451. <https://doi.org/10.1002/cae.20568>
- [11] Clarke, R.P. (2011) ENGLTHA: An Educational Tool for Earthquake Nonlinear and General Linear Dynamics. *Computer Applications in Engineering Education*, **19**, 97-106. <https://doi.org/10.1002/cae.20295>
- [12] da Silva, J.G.S, da Silva Vellasco, P.C.G. and de Almeida, N.N. (2002) DINEST: An Educational Software for Structural Dynamic Design and Behavior. *Session International Conference on Engineering Education*, Manchester, 18-21 August 2002. <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=0027ed5efac5807db2cbeac58fd07b47c60b8083>
- [13] Brownjohn, J.M.W and Pavic, A. (2002) NDOF: A Matlab Gui for Teaching and

- Simulating Structural Dynamics. *IMACXXVI*, Orlando, 4-7 February 2008.
https://www.researchgate.net/profile/Aleksandar-Pavic-2/publication/266181526_NDOF_A_MATLAB_Gui_for_teaching_and_simulating_structural_dynamics/links/550fe8b10cf2752610a178fb/NDOF-A-MATLAB-Gui-for-teaching-and-simulating-structural-dynamics.pdf
- [14] Sonparote, R.S. and Mahajan, S.K. (2018) An Educational Tool to Improve Understanding of Structural Dynamics through Idealization of Physical Structure to Analytical Model. *Computer Applications in Engineering Education*, **26**, 1270-1278.
<https://doi.org/10.1002/cae.22006>
- [15] Mahajan, S.K. and Sonparote, R.S. (2018) Implementation of Comparative Visualization Pedagogy for Structural Dynamics. *Computer Applications in Engineering Education*, **26**, 1894-1902. <https://doi.org/10.1002/cae.22024>
- [16] Paultre, P., Léger, P. and Proulx, J. (1990) Computer Graphics for Computer Assisted Learning of Structural Analysis. *Computers & Structures*, **36**, 1159-1166.
[https://doi.org/10.1016/0045-7949\(90\)90225-Q](https://doi.org/10.1016/0045-7949(90)90225-Q)
- [17] Paultre, P., Léger, P. and Proulx, J. (1991) Computer-Aided Education in Structural Dynamics. *Journal of Computing in Civil Engineering*, **5**, 374-390.
[https://doi.org/10.1061/\(ASCE\)0887-3801\(1991\)5:4\(374\)](https://doi.org/10.1061/(ASCE)0887-3801(1991)5:4(374))
- [18] Paultre, P., Lapointe, E., Carbonneau, C. and Proulx, J. (2016) LAS: A Programming Language and Development Environment for Learning Matrix Structural Analysis. *Computer Applications in Engineering Education*, **24**, 89-100.
<https://doi.org/10.1002/cae.21675>
- [19] François, S., et al. (2021) Stabil: An Educational Matlab Toolbox for Static and Dynamic Structural Analysis. *Computer Applications in Engineering Education*, **29**, 1372-1389. <https://doi.org/10.1002/cae.22391>
- [20] Gavin, H.P. (2022) Frame3DD. <http://frame3dd.sourceforge.net>
- [21] Yuan, X.F. and Teng, J.G. (2002) Interactive Web-Based Package for Computer-Aided Learning of Structural Behavior. *Computer Applications in Engineering Education*, **10**, 121-136. <https://doi.org/10.1002/cae.10020>
- [22] Barretto, S.F.A., Piazzalunga, R. and Ribeiro, V.G. (2003) A Web-Based 2D Structural Analysis Educational Software. *Computer Applications in Engineering Education*, **11**, 83-92. <https://doi.org/10.1002/cae.10040>
- [23] Ziemian, R.D. and McGuire, W. (2022) MASTAN2 v.35.
<http://www.mastan2.com/about.html>
- [24] Lopes, P.C., Rangel, R.L., and Martha, L.F. (2020) Ftool 5.0: Nonlinear, Stability and Natural Vibration Analyses. *Proceedings of the XLI Ibero-Latin-American Congress on Computational Methods in Engineering, ABMEC/UNILA*, Foz do Iguaçu, 16-19 November 2020.
<http://webserver2.tecgraf.puc-rio.br/~lfm/papers/RangelMartha-CILAMCE2020-CODE7805.pdf>
- [25] Wilson, E.L. (1979) CAL—A Computer Analysis Language for Teaching Structural Analysis. *Computers & Structures*, **10**, 127-132.
[https://doi.org/10.1016/0045-7949\(79\)90079-8](https://doi.org/10.1016/0045-7949(79)90079-8)
- [26] Turker, H.T., Coskun, H. and Mertayak, C. (2016) Innovative Experimental Model and Simulation Method for Structural Dynamic Concepts. *Computer Applications in Engineering Education*, **24**, 421-427. <https://doi.org/10.1002/cae.21720>
- [27] Wang, B.P. and Apte, A. (2022) Dyssolve.
<https://sites.google.com/site/dyssolve/>

- [28] Felippa, C.A. (2004) Introduction to Finite Element Methods. University of Colorado, Boulder.
- [29] Bathe, K.J. (2006) Finite Element Procedures. Prentice Hall, Hoboken.
- [30] Booch, G., Rumbaugh, J. and Jacobson, I. (2005) The Unified Modeling Language User Guide. Pearson Education, Upper Saddle River.
- [31] Paz, M. and Kim, Y.H. (2012) Structural Dynamics: Theory and Computation. Springer Science & Business Media, Berlin.

Appendix

The local mass matrix of a 3D beam element is implemented according to Equation (9) to Equation (14), where E is the material Young's modulus, G is the material shear modulus, A_s is the cross-section effective shear area, I is the cross-section moment of inertia, and J is the cross-section polar moment of inertia. The matrix is decomposed into an Euler-Bernoulli component, M_e^{EB} , and a Timoshenko component, M_e^{Tim} . The latter accounts for the effects of shear deformation when the element is Timoshenko type and depends on the Timoshenko parameter Ω , given in Equation (12). In the case of Euler-Bernoulli element, in which shear deformation is neglected ($GA_s \rightarrow \infty$), the Timoshenko parameter vanishes and, consequently, all coefficients of the Timoshenko matrix become null, remaining only the classical Euler-Bernoulli mass matrix of Equation (10).

$$M_e = M_e^{EB} + M_e^{Tim} \tag{9}$$

$$M_e^{EB} = \frac{\rho AL}{420} \begin{bmatrix} 140 & 0 & 0 & 0 & 0 & 0 & 70 & 0 & 0 & 0 & 0 & 0 \\ 156 & 0 & 0 & 0 & 22L & 0 & 54 & 0 & 0 & 0 & 0 & -13L \\ & 156 & 0 & 22L & 0 & 0 & 0 & 54 & 0 & -13L & 0 & \\ & & 140J/A & 0 & 0 & 0 & 0 & 0 & 70J/A & 0 & 0 & \\ & & & 4L^2 & 0 & 0 & 0 & 13L & 0 & -3L^2 & 0 & \\ & & & & 4L^2 & 0 & 13L & 0 & 0 & 0 & -3L^2 & \\ & & & & & 140 & 0 & 0 & 0 & 0 & 0 & \\ & & & & & & 156 & 0 & 0 & 0 & -22L & \\ & & & & & & & 156 & 0 & -22L & 0 & \\ & & & & & & & & 140J/A & 0 & 0 & \\ & & & & & & & & & 4L^2 & 0 & \\ & & & & & & & & & & 4L^2 & \\ \text{sym.} & & & & & & & & & & & \end{bmatrix} \tag{10}$$

$$M_e^{Tim} = \frac{\rho AL}{420} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & \gamma_1 & 0 & 0 & 0 & -\gamma_2 & 0 & \gamma_3 & 0 & 0 & 0 & \gamma_2 \\ & & \gamma_1 & 0 & -\gamma_2 & 0 & 0 & 0 & \gamma_3 & 0 & \gamma_2 & 0 \\ & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & \gamma_4 & 0 & 0 & 0 & -\gamma_2 & 0 & \gamma_4 & 0 \\ & & & & & \gamma_4 & 0 & -\gamma_2 & 0 & 0 & 0 & \gamma_4 \\ & & & & & & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & \gamma_1 & 0 & 0 & 0 & \gamma_2 \\ & & & & & & & & \gamma_1 & 0 & \gamma_2 & 0 \\ & & & & & & & & & 0 & 0 & 0 \\ & & & & & & & & & & \gamma_4 & 0 \\ & & & & & & & & & & & \gamma_4 \\ \text{sym.} & & & & & & & & & & & \end{bmatrix} \tag{11}$$

$$\begin{aligned} \gamma_1 &= 24 \left(\frac{\mu\theta - \Omega}{\mu^2} - 1 \right) & \gamma_2 &= 6L \left(\frac{\Omega(9\mu + 2)}{\mu^2} \right) \\ \gamma_3 &= 36L \left(\frac{\Omega(5\mu + \theta)}{\mu^2} \right) & \gamma_4 &= \frac{L^2}{2} \left(\frac{1}{\mu^2} - 1 \right) \end{aligned} \tag{12}$$

$$\mu = 1 + 12\Omega \quad \theta = 1 + 4\Omega \quad (13)$$

$$\Omega = \frac{EI}{GA_s} \frac{1}{L^2} \quad (14)$$