Available online at www.jlls.org

JOURNAL OF LANGUAGE AND LINGUISTIC STUDIES

ISSN: 1305-578X

Journal of Language and Linguistic Studies, *18*(4), 1150-1155; 2022

Didactic Proposal For Learning Finite Element Methods

Wilson Pinzón Casallas[a], Wilson Gordillo Thiriata[b], Orlando García Hurtado[c]

[a] "Universidad Distrital Francisco José De caldas", Bogotá Colombia, *wjpinzonc@udistrital.edu.co*
ORCID: https://orcid.org/0000-0003-0258-6810
[b] "Universidad Distrital Francisco José De caldas", Bogotá Colombia, *wgordillot@udistrital.edu.co*
ORCID: https://orcid.org/0000-0002-3856-4691
[c] "Universidad Distrital Francisco José De caldas", Bogotá Colombia, *ogarciah@udistrital.edu.co*
ORCID: https://orcid.org/0000-0002-4155-4515

APA Citation:

Casallas, W.P., Thiriata, W.G., Hurtado O.G., (2022). Didactic Proposal For Learning Finite Element Methods, *Journal of Language and Linguistic Studies*, *18*(4), 1150-1155

Submission Date: 26/10/2022
Acceptance Date: 23/12/2022

ABSTRACT

The Finite Element Method "FEM" is a numerical method for the solution of differential equations used in various areas of engineering and physics, which discretizes a domain by dividing it into a finite number of geometric elements that approximate the original model, by continuously generating values of the unknowns throughout the entire solution domain instead of isolated points, facilitating the development of a didactic proposal for learning the finite element method using the mathematical software Matlab. An example is presented that reveals the suitability of the approach adopted to reduce the existing gap between the mathematical support of the method and the simplicity of software use. Also, numerical methods have become an indispensable tool for engineers; Already that gives them, in an approximate way, the solution of an application problem, which analytically it would be impossible to solve. Which provide the engineering student with knowledge basic of mathematical applications for the numerical solution of problems, allowing likewise to present the guidelines used for the design of algorithms and the construction of programs that, with the help of the computer, facilitate the work of the Engineer.

Keywords: Finite Element Method, didactic, Matlab

1. Introduction

Archimedes used a method similar to that of the finite element to determine the calculated areas, lengths and volumes of geometric objects, dividing them into simpler ones and then adding them up, the concept of variational approximation is nowhere to be seen. By changing "measure" to energy and "objects" to elements in the above lines, the description approximates the FEM statement "the energy of the system is equal to the sum of the energy of each element". However, Archimedes needed the derivative definitions to perform his energy calculations and the calculus was not invented until 20 centuries later.

EMAIL ID : wjpinzonc@udistrital.edu.co

FEM was originally developed in the 60's for the analysis of structures ranging from vibration analysis to heat transfer, fluid flow or electromagnetic fields, it can also be used in: construction industry, aerospace industry, automotive safety, study of the human body for the biomedical field, among others. FEM is a general numerical method for the approximation of solutions of partial differential equations, which lies in the division of a finite number of geometric elements that approximate the original model of the domain, in which the variable to be calculated is defined and on which the approximation is applied. The procedure involves the characterization and corresponding definition of the behavior of each element separately and the subsequent assembly of these, obtaining an approximate solution of the global behavior of the discretized system.

Similarly, future engineering professionals must be able not only to obtain results by means of commercial software, but also to know the details of the mathematical formulation and numerical calculation tools used by these codes. Students should have the opportunity to create and access the source code, which is not possible in commercial programs, which act as black boxes that do not allow understanding the processes for the solution of the same, or allow them to develop simple finite element codes that serve as a tool for learning the same, allowing the mathematical formulation of their solutions and practical application, enabling the mathematical software Matlab as a suitable tool for this purpose.

Matlab is a very powerful matrix-oriented engineering software with its own programming language (m language). Its ability to manipulate matrices and solve matrix equations makes it an ideal tool for the implementation and development of a finite element code. In this article the finite element code has been created with the help of students and is used to solve a simple structural calculation problem that will serve to present a didactic proposal for learning FEM with the use of MATLAB.

By means of linear finite elements, the stress and displacement fields of a bridge pier are obtained, whose section is widened following an exponential law towards the upper end in order to adequately support the deck in all its width. The code has been implemented in Matlab version R2022b and in order to facilitate a better understanding and reuse of it, the codes corresponding to some approaches are shown and the most relevant aspects of them are detailed.

2. Statement of the problem

A bridge pier of height h = 20m, whose section is widened following an exponential law towards the upper end in order to adequately support the deck over its full width. The area at the base is 2 m^2 while at the upper end it is 4 m^2. The modulus of elasticity is equal to 2.5×10^7 kN/m^2. Find the deformations along the structure under the action of a concentrated load at the end of 3000kN, corresponding to the weight of the deck. The effect of the column self-weight is neglected.

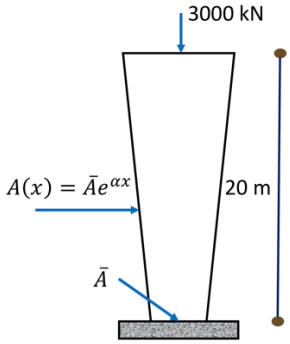

Figure 1. Bridge abutment, whose section is widened following an exponential law towards the upper end in order to adequately support the deck over its entire width.

3. Elementos lineales

The member is discretized by five linear bar-type elements in order to expose the matrix assembly characteristic of the method and the displacement and stress fields calculated by means of the FEM are compared a posteriori with the analytical solution.

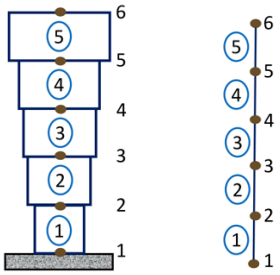

Figure 2. Discretization of bridge abutment

This example allows to understand the matrix formulation of the FEM and to become familiar with the pre-processing, processing and post-processing steps intrinsic to the method and to programming in Matlab. It is necessary to comment on some of the aspects related to the programming of the code.

The first thing that is done is the programming of a window that allows to enter the length of the element, the number of elements in which it is discretized and the force that is exerted.

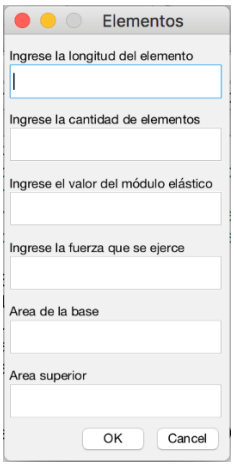

Figure 3. Data entry

```
prompt = {'Ingrese la longitud del elemento',...
    'Ingrese la cantidad de elementos',...
    'Ingrese el valor del módulo elástico',...
    'Ingrese la fuerza que se ejerce',...
     'Área de la base','Área superior'};
dlg_title = '  Elementos  ';
num_lines = 1;
answer = inputdlg(prompt,dlg_title,num_lines);
L = str2num(answer{1});
NumElem = str2num(answer{2});
Me = str2num(answer{3});
Long = L/NumElem;
F1 = str2num(answer{4});
Abase = str2num(answer{5});
Asupe = str2num(answer{6});
```

By means of a for/end loop, the connection matrix is calculated, which defines the connections between elements, where each row has a correspondence with an element and where the first column refers to the element number, column two is the initial point of the element and the third column is the end point of the element, within this same loop the stiffness matrices of each element are generated and assembled in the global stiffness matrix. To obtain the stiffness matrix of each element.

```
for i=1:NumElem
    Longi=Long*i;
    Area(i,:) = Area_sec*exp(alfha*Longi);
    Longitud(i,:) = Long;
    E(i,:) = Me;
    PI = i;
    PF = i + 1;
    Inicio(i,:) = PI;
    Final(i,:) = PF;
```

```
   Num_Elem(i,:) = i;
   conexion =[Num_Elem Inicio Final];
   EAL = E(i)*Area(i)/Longitud(i);
   K = EAL [1 -1;
       -1 1];
   Kelem(:,:,i) = K;
end
```

The displacement vector (Displacements), the force vector (Force) and the stiffness matrix (Stiffness) are defined. All of them are also initialized by means of the Matlab zeros function, which allows speeding up the program calculation process in the loops.

The stiffness matrix of a linear member element is a function only of its geometry (L, A) and its mechanical properties (E). So in the present example, where Young's modulus and cross-section of the member are constant, the stiffness matrix of each element corresponds to the following expression:

$$K = EAL\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

The stiffness matrices of each element, which have been calculated according to (K) and have size 2x2, are distributed to form the global stiffness matrix according to the GDLelem vector. The assembly process is carried out by the following line of code:

```
Kglobal = zeros(NumElem+1,NumElem+1);
for i=1:NumElem
GDLelem = GDL(i,:);
Kglobal(GDLelem,GDLelem) = Kelem(:,:,i)+Kglobal(GDLelem,GDLelem);
end
```

So for this case, where the member has been discretized by four finite elements, the global stiffness matrix takes the following form:

$$K = \begin{bmatrix} (EAL)^{(1)} & -(EAL)^{(1)} & 0 & 0 & 0 \\ -(EAL)^{(1)} & \left[(EAL)^{(1)} + (EAL)^{(2)}\right] & -(EAL)^{(2)} & 0 & 0 \\ 0 & -(EAL)^{(2)} & \left[(EAL)^{(2)} + (EAL)^{(3)}\right] & -(EAL)^{(3)} & 0 \\ 0 & 0 & -(EAL)^{(3)} & \left[(EAL)^{(3)} + (EAL)^{(4)}\right] & -(EAL)^{(4)} \\ 0 & 0 & 0 & -(EAL)^{(4)} & (EAL)^{(4)} \end{bmatrix}$$

Once the global stiffness matrix has been calculated, it is possible to solve the global system of equations intrinsic to the finite element method: $K \cdot f = a$. Where K is the global stiffness matrix, f the global force vector and a the nodal displacements.

If a linear system is considered such that $A \cdot X = B$, the solution vector X can be obtained in Matlab by employing the inverse slash (\): X=A\B. Consequently, the same system is applied to obtain the displacements at each node of the bar.

```
Desp = Kglobal(GDL(:,2),GDL(:,2))\Fuerza(GDL(:,2))';
Desplazamientos = zeros(size(conexion,1),1);
Desplazamientos(GDL(:,2)) = Desp;
```

From the values of the nodal displacements it is possible to calculate other parameters of interest, such as the stresses in each element, which will be stored in the Stress vector. These are calculated from the deformations, which in turn are calculated from the nodal displacements.

Once all the information needed from the nodal displacements has been obtained, the calculation or processor stage ends and gives way to the visualization or post-processor stage, where the results obtained are displayed in detail. This phase is very important when solving complex geometries and commercial finite element software includes powerful tools to visualize the results obtained. In the present example, being a simple one-dimensional configuration, the analytical solution is known and consequently there is the possibility of establishing comparisons with it. Thus, by means of the plot tool, the displacements calculated by the FEM are represented graphically.

The plots obtained for the displacement field and the stress field are shown in Figures 4 and 5 respectively. The results have been represented in a simple and functional way in order to eliminate obstacles in teaching, but MATLAB offers a wide range of tools to improve and complement the aesthetics and detail of the graphs. As can be seen, the values obtained for the displacements (Figure 4) and stresses (Figure 5) from the problem data are plotted as a function of the position on the member. The solution obtained by means of the FEM is reproduced by means of a dashed blue line, where the position of the nodes is indicated by small circles of the same color, as described in the legend of both images.

Figure 4 shows that the nodal displacements vary linearly, approximating the analytical solution reasonably well.

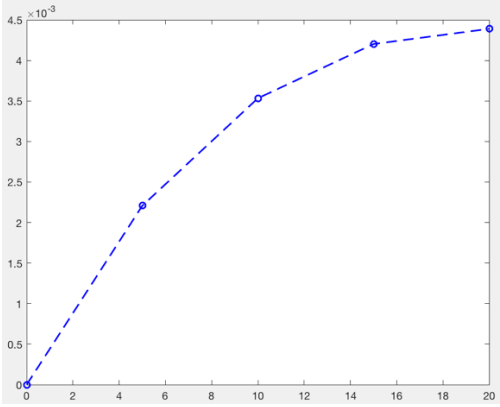

Figure 4. Displacements obtained as a function of position

Students can see how compatibility is always satisfied at the nodes and consequently there is a continuous displacement field, which can be computed with a larger number of nodes.

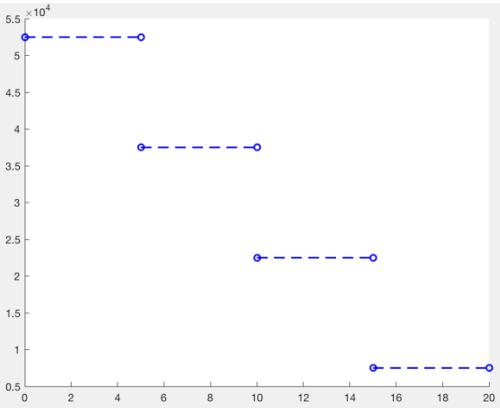

Figure 5. Stresses as a function of position

However, the approximation of the stress field in Figure 5 has differences between the exact solution and the one obtained by FEM, therefore it is convenient to discretize the member with a larger number of elements to reduce the error in the axial stress calculation. The student can observe that in the FEM,

generally, there is no stress equilibrium at the nodes. The stresses at the nodes are obtained in each element from the values of the displacements at its nodes and consequently, the stresses at a node common to several elements can take different values. Thus, from the example presented, the student quickly obtains the deformations and stresses from the derivatives of the displacement field.

In addition, the programming code allows to modify the meshing of the problem easily, requiring only to modify the variables element length (L), number of elements (NumElem), elastic modulus (Me), force exerted (F1), area of the base (Abase), upper area (Asupe). This will allow the student to obtain the solution by means of the FEM for different discretizations, observing how the solution obtained gets closer and closer to the exact solution as the number of elements used increases. But above all, the exercise will allow the student to become familiar with the basic stages of the FEM. That is:

- Discretize the problem into a series of finite elements connected together at the nodes (meshing).
- Compute the stiffness matrix ($K^{(e)}$) and the vector of nodal forces ($f^{(e)}$) for each element of the system.
- Assemble and solve the global equilibrium matrix equation ($K \cdot f = a$) to, once the boundary conditions are imposed, calculate the values of the displacements at the nodes.
- Calculate the parameters of interest in each case from the nodal displacements obtained.
- Clearly visualize the results in order to make the right decisions regarding the design of the analyzed component.

Correct programming of the example by the student will help him/her understand the basic structure of the method and this will allow him/her to develop the code to answer more complex problems.

4. Conclusions

By means of MATLAB mathematical software, an intelligible finite element code has been developed, capable of solving simple structural problems by gradually addressing the most relevant features of the FEM. It has been used to obtain the displacement and stress fields in a bridge pier, whose section is widened following an exponential law towards the upper end in order to adequately support the deck over its full width. The solved problem, the order of the examples and the sequence of operations developed in the process of solving them obey exclusively to purely didactic purposes, with the objective that the student faces in a progressive way the particularities inherent to the FEM.

The structure of the code has been programmed with the help of the students so that it can be extended, without the need for major modifications, to respond to more complex problems, either because of their dimension (2D, 3D) or because of the type of finite element to be used (beam, truss, etc.). In any case, the example shown in this article has proved to be more than enough to introduce the FEM to engineering students, serving as a link between the mathematical formulation of the method and the commercial software. Because of its simplicity of use and programming, the mathematical software MATLAB has proved to be a very valuable teaching tool when teaching students the basic fundamentals of the FEM.

References

Martínez-Pañeda, E., & Betegón, C., (2015). Modeling damage and fracture within strain- gradient plasticity. International Journal of Solids and Structures, 59, 208-215.

Pañeda, E. M. (2016). MATLAB: A tool for the didactics of the Finite Element Method. Unión - iberoamerican journal of mathematics education., 12(45).

Rahman, T., & Valdman, J. (2013). Fast MATLAB assembly of FEM matrices in 2D and 3D: Nodal elements. Applied Mathematics and Computation, 219 (13), 7151-7158.