Isogeometric analysis with interactive modeling of multi-patches NURBS

João Carlos L. Peixoto[1], Rafael L. Rangel[2], Luiz F. Martha[1]

[1]*Dept. of Civil and Environmental Engineering, Pontifical Catholic University of Rio de Janeiro (PUC-Rio)*
R. Marquês de São Vicente, 225, 22451-900, Gávea, Rio de Janeiro - RJ, Brazil
joaoclpeixoto@gmail.com, lfm@tecgraf.puc-rio.br
[2]*International Centre for Numerical Methods in Engineering (CIMNE)*
Carrer del Gran Capità s/n, UPC Campus Nord, 08034 Barcelona, Spain
rrangel@cimne.upc.edu

Abstract. Isogeometric analysis (IGA) is an innovative method of numerical analysis of structures that seeks to unite design and simulation, allowing the creation of computational models that preserve the exact geometry of the problem. This approach uses mathematical functions known as NURBS (Non-Uniform Rational B-Splines), widely used in CAD systems to model curves and surfaces. In the context of the IGA, these same functions that define the geometry are used to approximate the field variables. This work aims to provide a computational tool for two-dimensional isogeometric analysis, covering the stages of modeling, analysis, and visualization of the results. It is worth mentioning that the visualization of results is still a prototype under development. The system consists of two software: FEMEP (Finite Element Method Educational Computer Program), developed in Python, responsible for geometric modeling, and FEMOOLab (Finite Element Method Object-Oriented Laboratory), a MATLAB software used for analysis and display of results. The proposed tool presents an intuitive graphical user interface (GUI) to facilitate the visualization and manipulation of NURBS curves, providing advanced modeling features. The device also includes features for the automatic intersection of curves and region recognition to simplify and streamline the modeling process. In addition, the modeling software is also responsible for managing attributes such as material properties and natural and essential boundary conditions. A relevant aspect of this work is to provide open-source code, allowing the user and developer community to collaborate and contribute to the continuous development of the tool. This open-source approach encourages innovation and shared growth, fostering an environment for collaboration and mutual learning. This tool aims to make isogeometric analysis accessible even to users with basic programming knowledge, expanding its scope and enabling its application in various projects and engineering studies.

Keywords: Finite Element Method, Isogeometric Analysis, NURBS, Educational Software, Object-oriented Programming.

1 Introduction

When dealing with numerical simulations of real engineering objects, creating geometric models is a crucial task in mesh generation. In the case of planar models, employing a boundary representation (B-rep) [1] to describe the solid's boundary explicitly appears to be a suitable and intuitive approach. Nevertheless, numerical simulations demand further modeling capabilities: for instance, the Finite Element Method (FEM), a widely used numerical technique to solve partial differential equations, requires discrete models such as domain meshes generated based on the geometric information provided by the model's domain description. Various algorithms are available for this purpose, with discrete transfinite mapping [2] being a pertinent example.

Numerical simulations involve other fundamental aspects besides addressing geometric modeling and meshing. These include defining model attributes, such as material properties, displacement constraints as essential boundary conditions, and loads and external solicitations as natural boundary conditions. Defining and creating simulation attributes and associating them with the model's topological and geometric entities are indispensable. The topic has been thoroughly explored in various dissertations and theses within the research group [3-6]. Recently published articles [7-9] also delve into this matter.

In FEM programs, the isoparametric formulation is typically employed. In this approach, the finite element's geometry is interpolated using the shape functions, which are generally polynomial because of their simplicity. These same functions are then employed to interpolate the field variables [10]. The shortcoming of the isoparametric concept is that the shape functions chosen to approximate the unknown solution fields are also used to approximate the known geometry. In this context, the isogeometric formulation emerges as a recent technique, first introduced by Hughes et al. [11]. This approach uses the same class of functions commonly employed for geometry parameterization in Computer-Aided Design (CAD), specifically Non-Uniform Rational B-Splines (NURBS), in the solution space of Partial Differential Equations (PDEs). Therefore, IGA can be understood as a generalization of standard FEM where more sophisticated interpolation functions are used. In the context of IGA, the distinction between geometry and analysis as separate entities no longer exists. Geometric approximations are inexistent, given that the analysis is conducted directly within the original design model. This comprehensive approach eradicates potential mesh approximation errors similar to those encountered in FEM, resulting in a process with higher accuracy and faster convergence.

The primary objective of this work is to provide an educational tool for students and researchers focusing on two-dimensional isogeometric analysis. This tool encompasses the entire spectrum of modeling, analysis, and result visualization. It's important to note that the result visualization component is still in the developmental prototype stage. The system comprises two software components: FEMEP (https://gitlab.com/danilosb/femep), developed in Python and dedicated to the pre-processing (geometric modeling), and FEMOOLab (https://gitlab.com/rafaelrangel/femoolab), a MATLAB-based application designed for analysis and result visualization. Both applications are proficient in conventional finite element analysis but have been expanded to encompass isogeometric analysis, which is the focus of this article. The proposed modeling system features a graphical user interface (GUI) that simplifies NURBS curves visualization and manipulation, offering advanced modeling capabilities. The tool includes functionalities like automatic curves intersection and region recognition, streamlining the modeling process. Furthermore, the modeling software manages crucial attributes such as material properties as well as natural and essential boundary conditions.

The present article is structured into five additional sections. Section 2 intends to offer a theoretical overview of B-Splines and NURBS. Section 3 provides a concise review of isogeometric analysis applied to linear elasticity, delving into the parametric spaces of IGA and fundamental concepts. In Section 4, we discuss the design of the modeling software, elucidating the code structure and class organization. Section 5 presents a numerical example of a problem modeled and analyzed using the developed tool. Lastly, Section 6 encapsulates the article with conclusive remarks followed by discussions on potential future developments.

2 B-Splines and NURBS concepts

B-Splines are a popular choice in engineering for accurately and smoothly representing curves and surfaces through piecewise-defined polynomial functions. To build a B-Spline, a knot vector is required. This vector contains non-negative coordinate values in the parametric space and must be arranged in ascending order. The knot vector can be represented in the form $\Xi = [\xi_1, \xi_2, \dots, \xi_{n+p+1}]$, where n is the number of basis functions, p is the polynomial degree, and ξ is the parametric coordinate of index $i = 1, 2, ..., n+p+1$. From the knot vector, the base functions of a B-Spline can be defined for a given degree p. Initially, for $p = 0$ the basis functions are determined by piecewise constants according to eq. (1).

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi \leq \xi_{i+1} \\ 0 & \text{Otherwise} \end{cases} \tag{1}$$

For $p > 0$, the basis functions are defined by the recursive Cox-de Boor formula:

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \tag{2}$$

The knots can assume repeated values; when this happens, we say there is multiplicity. The multiplicity of the parametric coordinates directly influences the continuity of a B-Spline. A vector of knots is said to be open if its first and last values have multiplicity $p+1$. Open knot vectors are standard in CAD systems.

From base functions $N_{i,p}$, built from a knot vector $\Xi = [\xi_1, \xi_2, \dots, \xi_{n+p+1}]$ and a set of control points $\boldsymbol{P}_i$, a B-Spline curve is defined by eq. (3).

$$C(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\boldsymbol{P_i} \tag{3}$$

Given two knot vectors $\Xi^1 = [\xi_1, \xi_2, \dots, \xi_{n+p+1}]$ and $\Xi^2 = [\eta_1, \eta_2, \dots, \eta_{m+q+1}]$ and a control net $\boldsymbol{P}_{i,j}$ a B-Spline tensor product surface is defined by eq. (4).

$$S(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) \boldsymbol{P_{i,j}} \tag{4}$$

B-Splines have a significant limitation of not being able to mathematically represent conic shapes, such as circles and ellipsoids, accurately. This gap in representation inspired the development of NURBS (Non-Uniform Rational B-Splines), which established themselves as an extension of B-Splines. NURBS are formed by rational functions of B-Splines, thus overcoming this limitation and enabling the accurate representation of more complex shapes. Univariate and bivariate NURBS basis functions are shown respectively in eq. (5) and eq. (6).

$$R_{i,p}(\xi) = \frac{N_{i,p}(\xi) w_i}{W(\xi)} = \frac{N_{i,p}(\xi) w_i}{\sum_{\hat{i}=1}^{n} N_{\hat{i},p}(\xi) w_{\hat{i}}} \tag{5}$$

$$R_{i,j}^{p,q}(\xi, \eta) = \frac{N_{i,p}(\xi) M_{j,q}(\eta) w_{i,j}}{\sum_{\hat{i}=1}^{n} \sum_{\hat{j}=1}^{m} N_{\hat{i},p}(\xi) M_{\hat{j},q}(\eta) w_{\hat{i},\hat{j}}} \tag{6}$$

Where w_i or $w_{i,j}$ are weights associated with each control point. A given weight determines the influence of that point on the final curve or surface. When all weights assume the same value, a NURBS reduces to a B-Spline. We can therefore say that NURBS are generalizations of B-Splines. Analogously, NURBS-type curves and surfaces are defined by eq. (7) and eq. (8), respectively.

$$C(\xi) = \sum_{i=1}^{n} R_{i,p}(\xi)\boldsymbol{P_i} \tag{7}$$

$$S(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} R_{i,j}^{p,q}(\xi, \eta) \boldsymbol{P_{i,j}} \tag{8}$$

The books A Practical Guide to Splines [12] and The NURBS Book [13] are recommended for a more in-depth discussion on this topic.

3 Isogeometric analysis: bidimensional linear elasticity

This section begins with an explanation of spaces intrinsic to isogeometric analysis. The expressions for the mapping between these spaces are presented, followed by the isogeometric formulation of finite elements for two-dimensional linear elasticity problems, which uses the NURBS basis functions. This approach unifies parametric geometry with numerical analysis, allowing an accurate and integrated representation of geometric shapes and the variables of interest. Using NURBS basis functions in the formulation enhances the connection between geometry and numerical analysis, resulting in more effective and reliable solutions.

3.1 Relevant spaces

The concept of parametric space refers to the space defined by the knot spans; in the two-dimensional case, we have two-knot spans, one for each direction $\hat{\Omega}_e = [\hat{\xi}_i, \hat{\xi}_{i+1}] \otimes [\hat{\eta}_j, \hat{\eta}_{j+1}]$. The physical space is the one that contains the geometry itself; for two dimensions, a NURBS surface forms the geometry. The parent space is not related to the NURBS nature, being used only to allow numerical integration via Gauss quadrature, in the two-dimensional case, we have $\tilde{\Omega} = [-1, 1]^2$.

3.2 IGA formulation

As explained in Section 2, the B-Splines and NURBS basis functions are defined in terms of parametric coordinates. However, applying these functions in the IGA context requires mapping between spaces that allow manipulation in the parent domain. The use of NURBS for numerical analysis incorporates the concept of parametric space, which is not present in traditional finite element analysis. Thus, two mappings are necessary to transform the parent to the physical domain: the first between the parent and the parametric spaces $\tilde{\phi}(\tilde{\xi}) = \tilde{\Omega} \rightarrow \hat{\Omega}_e$ given by a linear relation according to eq. (9), whose associated Jacobian is shown in eq. (10); the second between parametric and physical spaces $\hat{\phi}(\xi) = \hat{\Omega}_{\ddot{e}} \rightarrow \Omega_e$, given simply by expressing a NURBS surface in terms of parametric coordinates according to eq. (8), with the corresponding Jacobian exposed in eq. (11). Fig. 1 illustrates the spaces and mappings between them.

$$\tilde{\phi}(\tilde{\xi},\tilde{\eta}) = \begin{cases} \xi = \dfrac{1}{2}[(\hat{\xi}_{i+1} - \hat{\xi}_i)\tilde{\xi} + (\hat{\xi}_{i+1} + \hat{\xi}_i)] \\ \eta = \dfrac{1}{2}[(\hat{\eta}_{i+1} - \hat{\eta}_i)\tilde{\eta} + (\hat{\eta}_{i+1} + \hat{\eta}_i)] \end{cases} \tag{9}$$

$$\boldsymbol{J}_{\tilde{\xi}} = \begin{bmatrix} \dfrac{\partial \xi}{\partial \tilde{\xi}} & \dfrac{\partial \xi}{\partial \tilde{\eta}} \\ \dfrac{\partial \eta}{\partial \tilde{\xi}} & \dfrac{\partial \eta}{\partial \tilde{\eta}} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{2}\ \hat{\xi}_{i+1} - \hat{\xi}_i & 0 \\ 0 & \dfrac{1}{2}\ \hat{\eta}_{i+1} - \hat{\eta}_i \end{bmatrix} \tag{10}$$

$$\boldsymbol{J}_{\xi} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial x}{\partial \eta} \\ \dfrac{\partial y}{\partial \xi} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} P_{x\ 1,1} & \cdots & P_{x\ n,m} \\ P_{y\ 1,1} & \cdots & P_{y\ n,m} \end{bmatrix} \begin{bmatrix} \dfrac{\partial R_{1,1}^{p,q}}{\partial \xi} & \dfrac{\partial R_{1,1}^{p,q}}{\partial \eta} \\ \vdots & \vdots \\ \dfrac{\partial R_{n,m}^{p,q}}{\partial \xi} & \dfrac{\partial R_{n,m}^{p,q}}{\partial \eta} \end{bmatrix} \tag{11}$$

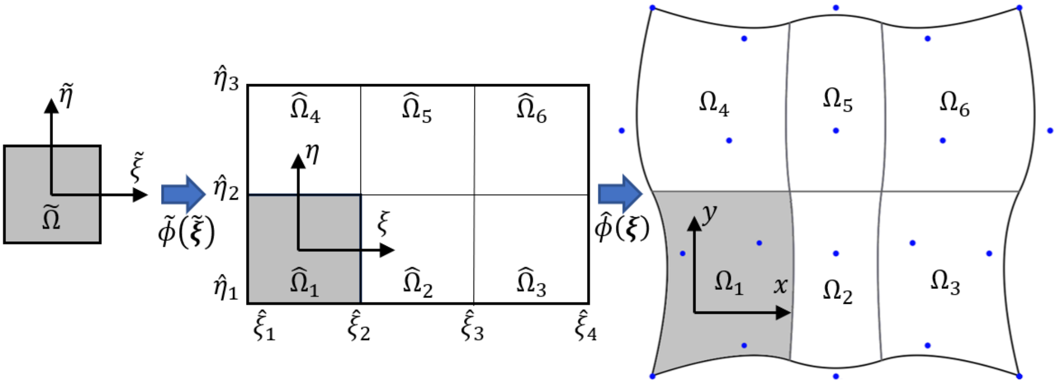


Figure 1. Mappings in IGA

Considering the presented mappings, the element stiffness matrix integrated into the parent space is given by eq. (12), where t is the thickness, B is the strain-displacement matrix shown in eq. (13), and D is the constitutive matrix.

$$K_e = t\int_{-1}^{1}\int_{-1}^{1} B^T(\tilde{\xi},\tilde{\eta}) D B(\tilde{\xi},\tilde{\eta}) \left|\boldsymbol{J}_{\tilde{\xi}}\right| \left|\boldsymbol{J}_{\xi}\right| d\tilde{\xi} d\tilde{\eta} \tag{12}$$

$$B_e = \begin{bmatrix} R_{1,x} & 0 & R_{2,x} & 0 & \cdots \\ 0 & R_{1,y} & 0 & R_{2,y} & \cdots \\ R_{1,y} & R_{1,x} & R_{2,y} & R_{2,x} & \cdots \end{bmatrix} \tag{13}$$

3.3 Multi-patches NURBS

Standard isogeometric discretization usually requires a global parametrization of the domain. However, modeling with multiple patches with separate discretization and parametrization can often be convenient. In this work, we assume geometric compatibility between adjacent patches, meaning that the control points along the edge of a patch have a one-to-one correspondence with control points at the edge of an adjacent patch, forming the so-called conforming patches. Figure 2 shows how this compatibility is achieved. When assembling the stiffness matrix and force vector, the connectivity of these control points is identified and they are interpreted by the program as a single entity. The result is the joining of all patches into a single system.

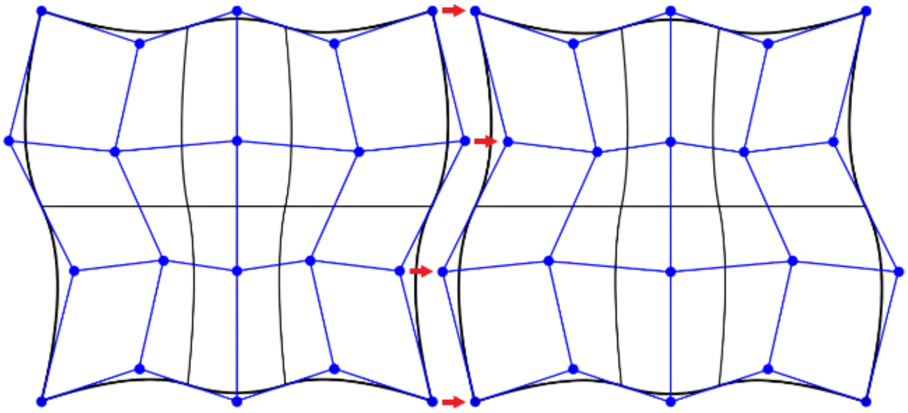


Figure 2: Conforming patches

4 Software design

In this section, we will explore the tools proposed in this work. The FEMEP, used for solid modeling, is a program implemented in Python with the object-oriented paradigm based on the HETOOL half-edge library [8] developed by the research group. The program interface is created with the QT and OpenGL systems, providing a compelling visualization of the models. Likewise, the software adopts the MVC (Model-View-Controller) pattern, with each component playing a distinctive role in the process. It is worth mentioning that FEMEP is an open-source software, promoting collaboration and flexibility for the continuous improvement of this tool. Figure 3 presents a schematic with the main modules of FEMEP.

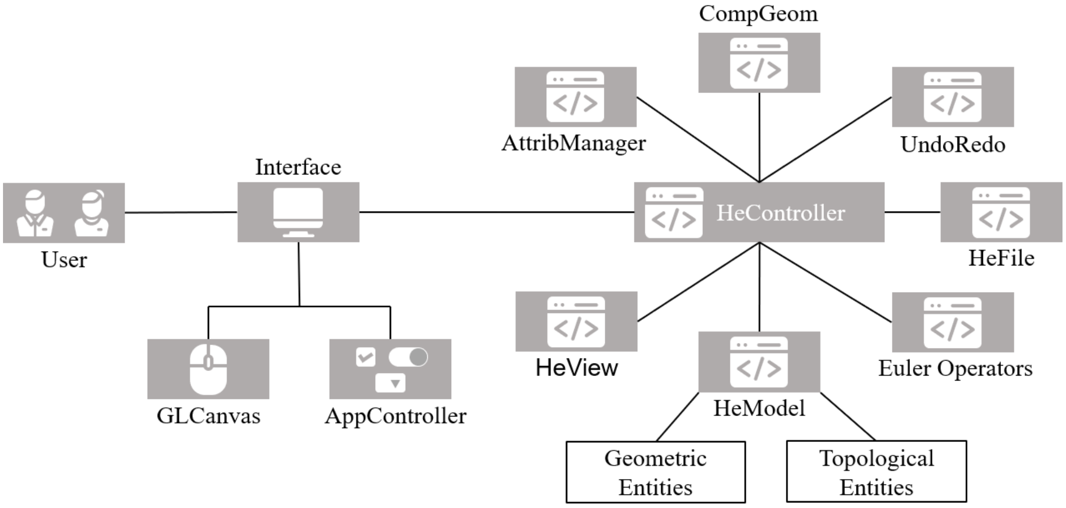


Figure 3. FEMEP general scheme

Initially, the user provides modeling information through the interface, which is composed of the *GLCanvas* and *AppController* classes. The former was developed from the *openGL* library and manages mouse events and renders the modeled solid on the screen. The *AppController* class aims to control the QT environment's interface elements (Widgets). The *HeController* class has a central role, as it accesses most modules. Its attributes cover the collection of information the user provides and the subsequent modifications in the *HeModel* through Euler operators. The Euler operators module is a set of classes that perform changes in the data structure at the level of topological entities, called in response to each modeling iteration executed by the program. In particular, these operators create closed surfaces and modify existing surfaces by adding or deleting faces, edges, and vertices.

The *HeModel* class has the function of storing geometric and topological entities. Entities of topological nature follow the hierarchy of the Half-edge data structure. On the other hand, geometric entities include classes that describe different aspects of geometry. Concretely, there is the *Patch* class, which denotes plane regions; the *Segment* class, which represents the curves; and the *Point* class, which encompasses discrete points. There are several pre-defined curves, which the user can insert: Lines, Polylines, Cubic Splines, Circles, Circle Arcs, Ellipses, and Ellipse Arcs. All these curves have an inherent property of their shapes, expressed in the context of NURBS. The representation of this property takes the form of an object from the *geomdl* library, which retains crucial information, such as the order of the curve, the control points, the knot vector, and the associated weights.

Patches are essentially delimited regions automatically recognized by the software, which can adopt a NURBS configuration. To enable such distinction, the user must select the "isogeometric" option in the mesh management interface. This choice presupposes compliance with requirements specific to NURBS surfaces, such as the equivalence of order of opposite curves, knot vector, and conformity between the ends. Once this step has been performed, a given patch incorporates the NURBS property, also represented as an object from the *geomdl* library. In this context, isocurves are plotted to delineate the contours of isogeometric elements associated with a given patch. To support this NURBS surface generation process, class methods are available to refine the curves by increasing the degree and inserting knots. In addition, it is also possible to perform refinement by directly inserting knots on the surface. Such procedures are performed through buttons in the interface, which activate functions in the *HeController* through the *AppController*.

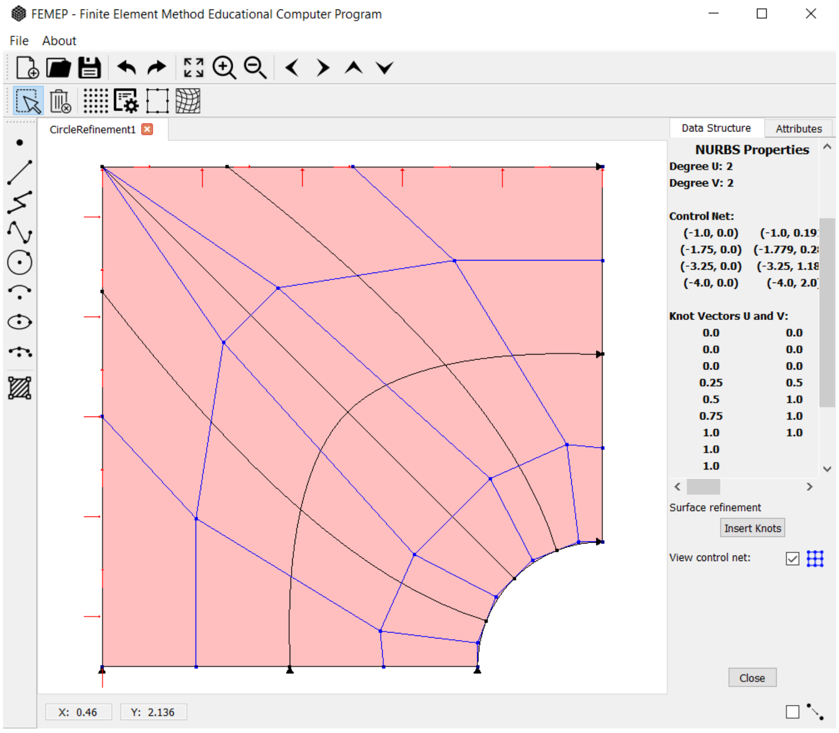
Other classes stand out, such as *UndoRedo*, which stores a certain amount of commands provided by the user, where it is possible to undo and redo specific actions. The *HeFile* class is responsible for saving and reading information related to the model, which is held in a JSON file equivalent to a dictionary structure in Python, consisting of structures formed by key:value. One particular aspect of FEMEP is its ability to manage attributes, allowing the allocation of information such as materials, support conditions, and loads directly into the model. The *AttribManager* class presents a set of methods that carry out the insertion or removal of attributes, as well as the modification of the corresponding values.

Subsequently, the procedure continues with the export of the FEMEP information for the analysis in the FEMOOLab environment in MATLAB. Such export is carried out through a text file that includes all information for the analysis, such as control points, their respective weights, and descriptions of the surfaces in question. Each surface is characterized by comprising the connectivity of the control points and specifying the order in the two directions and the corresponding knot vectors. The analysis input file also includes essential information about the conditions of support, loading, and the material's properties.

5 Numerical examples

5.1 Infinity plate with circular hole

The use of the developed tool will be demonstrated in a simple example of linear static analysis. The problem under consideration is an infinite plate subjected to constant in-plane horizontal tension and containing a centered circular hole. The selection of this particular problem stems from its inherent simplicity, which affords the possibility for comparative analysis with an analytical solution. The model encompasses a quarter of the plate due to its symmetry, with dimensions denoted as the side length of $L \text{ x } L$ with $L = 4$ m, while the circular hole has a unit radius. The material properties are defined by Young's modulus $E = 10^5$ MPa and Poisson’s ratio $\upsilon = 0.3$. Within this context, the analysis assumes a plane stress condition. The approach employs quadratic NURBS meshes to address this scenario. Figure 4 demonstrates the example in question modeled through FEMEP.



NURBS Properties

Degree U: 2

Degree V: 2

Control Net:

(-1.0, 0.0)	(-1.0, 0.19
(-1.75, 0.0)	(-1.779, 0.2
(-3.25, 0.0)	(-3.25, 1.18
(-4.0, 0.0)	(-4.0, 2.0

Knot Vectors U and V:

0.0	0.0
0.0	0.0
0.0	0.0
0.25	0.5
0.5	1.0
0.75	1.0
1.0	1.0
1.0	
1.0	

View control net: ☑

X: 0.46 Y: 2.136

Figure 4. Plate with circular hole modeled in FEMEP

The FEMOOLab software was used to perform the analysis, the obtained distribution of the σ_{xx} stress component is shown in Fig. 5 for a model of 1224 degrees of freedom. The analytical solution of the problem gives a maximum stress concentration of $\sigma_{xx} = 30$ MPa at the top edge of the hole. A comparison between the number of degrees of freedom and the σ_{xx} component at the aforementioned boundary obtained with isogeometric and isoparametric formulations is shown in Fig. 6. The converged value of the stress is 30 MPa for all cases. Notably, it was observed that the isogeometric formulation exhibited a faster rate of convergence. Furthermore, the study juxtaposed the findings with those acquired through the isogeometric solution using the IGAFEM software [14]. A noteworthy distinction pertained to the post-processing methodology; whereas FEMOOLab extrapolates the Gauss point stresses to the element edges, IGAFEM calculates stresses directly at the edges. This analysis reveals convergence nuances, enhancing understanding of isogeometric modeling's practical implications and advantages.

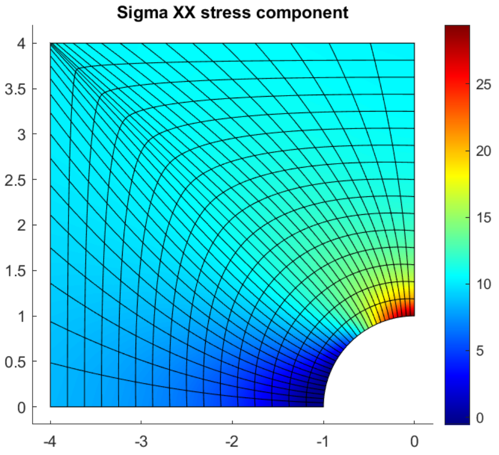


Figure 5. σ_{xx} stress obtained with 1224 dofs

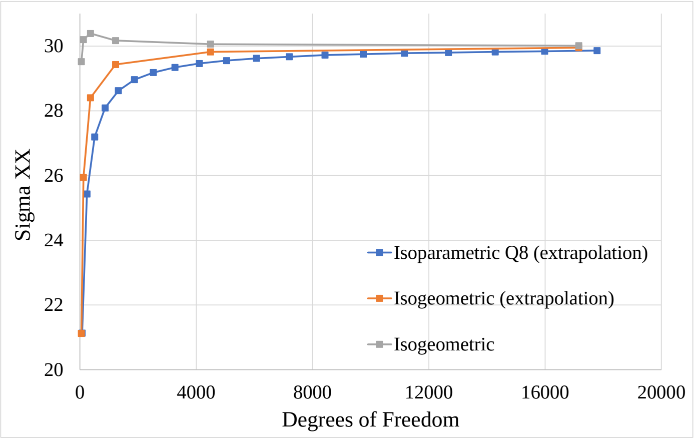


Figure 6. Max stress σ_{xx} and number of dofs

5.2 Steel bracket with multi-patches

The isogeometric modeling software offers a remarkable functionality by enabling the creation and manipulation of multiple NURBS patches, as showcased in Fig. 7 where we model a steel bracket from [10]. This problem presents some complexities due to the presence of internal holes, necessitating the use of 38 individual patches to accurately represent the geometry with NURBS surfaces. The results of the analysis are illustrated in Fig. 8, depicting the distribution of the sigma *yy* stress component.

As mentioned in Section 3.3, when dealing with conforming patches, adjacent faces share common control points, which are treated as a unified entity with one-to-one correspondence. This implies that in multi-patch problems, the stiffness matrix assembly is carried out taking into account the connectivity between adjacent patches. Consequently, a single global stiffness matrix is constructed, irrespective of the number of patches involved, and its dimensions are determined by the total number of degrees of freedom without any consideration for duplicated control points. For instance, in the example of Fig. 7, there are 612 unique control points, with no replication of control points across adjacent patches.

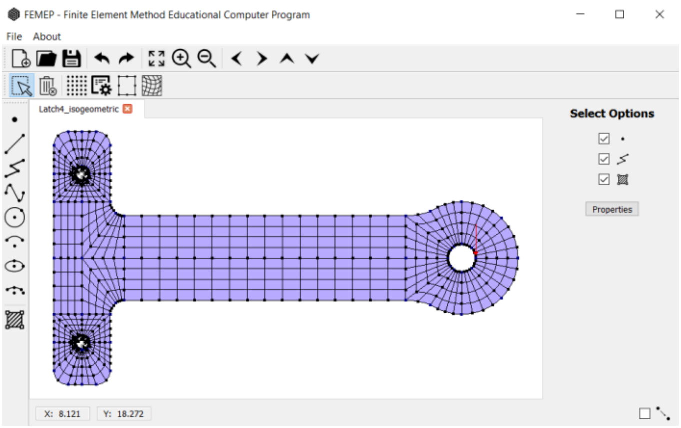


Figure 7: Steel bracket modeled in FEMEP with 38 patches

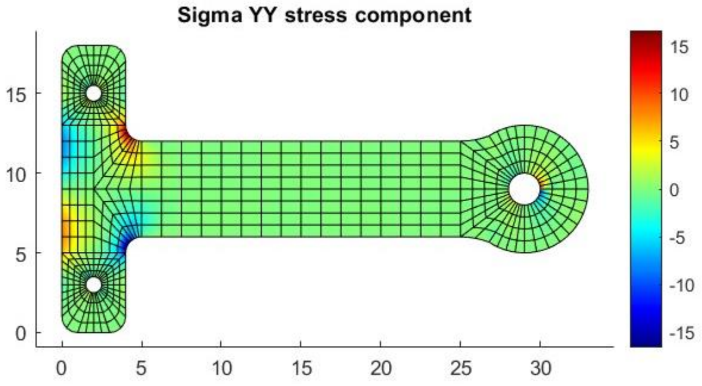


Figure 8: Distribution of sigma *yy* stress component

6 Conclusions

The comparative analysis between the isogeometric and isoparametric models presented in this study revealed valuable insights into the effectiveness and potential of IGA. The results showed the ability of the isogeometric model to accurately capture complex geometric features, offering a promising alternative to the traditional isoparametric formulation.

As we move towards more advanced modeling approaches, we must consider the continual improvement of the software features. In this context, a promising step for future work is incorporating the T-Splines technique into the software. While isogeometric mesh demonstrates its advantages, T-Splines provide greater flexibility, allowing the creation of unstructured meshes that can adapt to more complex scenarios.

Acknowledgements. The authors acknowledge the financial support from CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) – Finance Code 001, from FAPERJ (in Portuguese "Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro"), and from CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) – Fellowship Productivity in Research of the last author.

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] M. Mäntylä, An Introduction to Solid Modeling. Computer Science Press, 1988.
[2] R. Haber, M.S. Shephard, J.F. Abel, R.H. Gallagher, and D.P. Greenberg, "A general two-dimensional graphical finite element preprocessor utilizing discrete transfinite mappings". International Journal for Numerical Methods in Engineering, vol. 17, pp. 1015-1044, 1981.
[3] M.T.M. de Carvalho, Uma Estratégia para Desenvolvimento de Aplicações Configuráveis em Mecânica Computacional, Ph.D. thesis, Pontifical Catholic University of Rio de Janeiro. 1995.
[4] E.S.S. da Silveira, Um Sistema de Modelagem Bidimensional Configurável para Simulação Adaptativa em Mecânica Computacional, M.Sc. dissertation, Pontifical Catholic University of Rio de Janeiro, 1995.
[5] W.W.M. Lira, Um Sistema Integrado Configurável para Simulações em Mecânica Computacional, M.Sc. dissertation, Pontifical Catholic University of Rio de Janeiro, 1998.
[6] D.S. Bomfim, Uma estratégia de modelagem aberta e extensível para criação de modelos de subdivisões planares para mecânica computacional, M.Sc. dissertation, Pontifical Catholic University of Rio de Janeiro, 2022.
[7] D.S. Bomfim, R.L. Soares, L.F. Bez, P.C.F. Lopes, A.M.B. Pereira, and L.F. Martha, "Development of a Python Application Aiming at the Teaching-learning process of the Half-Edge Data Structure", Proceedings of the joint XLII IberoLatin-American Congress on Computational Methods in Engineering and III Pan-American Congress on Computational Mechanics, ABMEC-IACM, Rio de Janeiro, RJ, 2021.
[8] D.S. Bomfim, R.L. Soares, L.F. Bez, P.C.F. Lopes, A.M.B. Pereira, and L.F. Martha, "HETOOL: A Half-Edge Topological Object-Oriented Library for generic 2-D geometric modeling". SoftwareX, vol. 21, p. 101307, 2023.
[9] J.C.L. Peixoto, D.S. Bomfim, R.L. Soares, L.F. Bez, P.C.F. Lopes, A.M.B. Pereira, R.L. Rangel, and L.F. Martha, "Some modeling features for two-dimensional isoparametric and isogeometric finite element analysis", Proceedings of the joint XLIII Ibero-Latin-American Congress on Computational Methods in Engineering, ABMEC, Foz do Iguaçu, PR, 2022.
[10] K.J. Bathe, Finite Element Procedures in Engineering Analysis, Prentice-Hall, Englewood Cliffs, New Jersey, 1982
[11] J.A. Cottrell, T.J. Hughes, Y. Bazilevs, Isogeometric analysis: toward integration of CAD and FEA. John Wiley & Sons, 2009
[12] C. De Boor, A practical guide to splines. New York: springer-verlag, 1978.
[13] L. Piegl, W. Tiller, The NURBS book. Springer Science & Business Media, 1996.
[14] V. P. Nguyen, S. P. A Bordas, T. Rabczuk. Isogeometric analysis: an overview and computer implementation aspects. Mathematics and Computers in Simulation, vol. 117, pp. 89-116, 2015.