IMPLEMENTAÇÃO DE ANÁLISE MATRICIAL DE TRELIÇAS PLANAS USANDO LINGUAGEM OPEN SOURCE E ORIENTAÇÃO A OBJETO

DOI: 10.37702/2175-957X.COBENGE.2023.4629

Leon Vale Lobo - leonvlobo@gmail.com
UFPA

José Cristhian Mendes Saldanha - josecristhian7@gmail.com
UFPA

Lohameky Gomes Alves - lohameky.alves@outlook.com
UFPA

Gabriela Freire Pereira da Silva - gabriela.freiresilva@gmail.com
UFPA

Jayne Silva Benmuyal Damasceno - jayne.damasceno@itec.ufpa.br
UFPA

Resumo: *A análise matricial de estruturas é a formalização de um método de análise estrutural conhecido como método dos deslocamentos (ou método da rigidez) adaptado para implementação computacional. Essa adaptação é feita visando a facilidade de implementação da teoria em códigos de linguagem de programação de computadores de forma que todos os cálculos que foram aprendidos manualmente em disciplinas teóricas podem agora ser aplicados em scripts que estimulam criatividade e pensamento lógico nos estudantes de engenharia. O presente trabalho traz uma proposta de implementação desta análise (também denominada método da rigidez direta) em uma linguagem gratuita e open source, Julia, que surgiu a poucos anos e já possui dados concretos em eficiência de processamento além de sintaxe e semântica muito parecida a outras linguagens comumente utilizadas para computação técnica e científica, tudo isso em um paradigma de programação conhecido como Programação Orientada a Objeto.*

Palavras-chave: *Método da Rigidez Direta, Programação, Treliças, Julia Language e Orientação a Objeto*

IMPLEMENTAÇÃO DE ANÁLISE MATRICIAL DE TRELIÇAS PLANAS USANDO LINGUAGEM OPEN SOURCE E ORIENTAÇÃO A OBJETO PARA O ENSINO DE ESTRUTURAS

1 INTRODUÇÃO

A análise estrutural é pré-requisito de qualquer projeto de engenharia pois é nesta etapa que é analisada a distribuição de solicitações no sistema e a partir disso, segundo LOZOVEY (2022), é possível definir sistema construtivos, materiais escolhidos, mão de obra a ser empregada, logística e etc.

Por ser uma etapa tão fundamental está presenta nas mais diversas áreas da engenharia, como mecânica, civil, sanitária, ambiental, ferroviária naval entre outras, tendo seu assunto geralmente ministrado em disciplinas nos núcleos comuns das ciências básicas da engenharia, como a resistência dos materiais.

Dentro dos métodos de análise de estruturas formadas por barras pode se definir o método dos deslocamentos (ou método da rigidez) como o mais propício para implementação computacional pela sua alta capacidade de automação de cálculos baseados em tabelas. Sua resolução é baseada na lei de Hooke que pode ser formulada em parâmetros matriciais conforme mostra a equação $\{F\} = [K]\{D\}$ (*1*).

$$\{F\} = [K]\{D\} \quad (1)$$

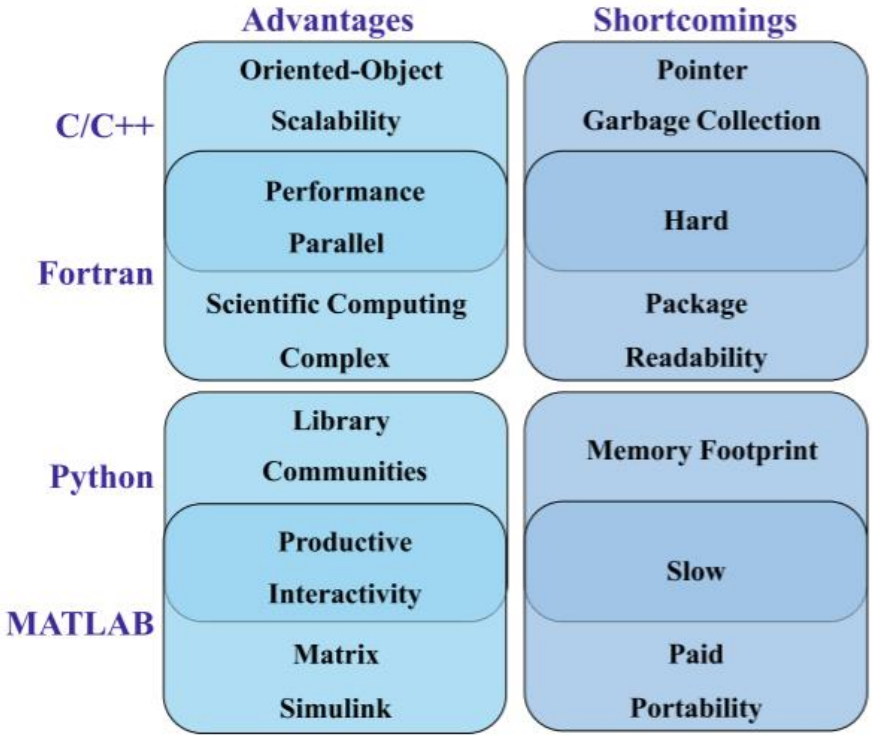
Onde $\{F\}$ é o vetor de carregamentos nodais aplicados na estrutura, que representa as ações incidentes na mesma, $[K]$ é a matriz de rigidez da estrutura, que é definida pelos parâmetros elásticos e geométricos de suas barras, e $\{D\}$ é o vetor de deslocamentos nodais da estrutura, que é o campo cinemático que caracteriza por completo a resposta do sistema estrutural. Tendo essas 3 informações é possível o cálculo de todas as outras variáveis de interesse como reações de apoio, esforços internos, deformações e etc.

Dada a complexidade cada vez maior dos fenômenos simulados na engenharia é usual a utilização de soluções numéricas para proposição de respostas dos problemas estudados (SANTOS et al., 2022), dentre os métodos numéricos mais comuns na análise estrutural tem-se o método dos elementos finitos, cuja formulação aplicada ao método dos deslocamentos de estruturas reticuladas é chamada de Método da Rigidez Direta (MRD).

Sendo um método numérico a capacidade de iteração de um problema até sua convergência é o cerne da técnica, sendo a utilização de computadores a peça chave que torna tudo isso possível; dentro dessa perspectiva a escolha da linguagem é parte fundamental da definição de como será feita a implementação do problema.

Uma das linguagens mais utilizadas em problemas de engenharia é o MATLAB, que possui vantagens como uma comunidade bem grande e ativa que produz pacotes para diversas soluções já implementadas e uma linguagem de alto nível que torna a escrita de princípios matemáticos simples (XIAO et al., 2021). Embora todas essas vantagens também culminem em desvantagens como lentidão de processamento e necessidade de pagamento pela licença, conforme mostra a Figura 1.

Figura 1 - Vantagens e desvantagens de linguagens popularmente utilizadas em computação científica (Fonte: XIAO et al., 2021*).*



Sendo a linguagem Julia uma linguagem relativamente nova com o lançamento de sua versão 1.0 em 2018, tendo os mesmos pontos vantajosos do MATLAB mas com contrapontos de ganho de velocidade de processamento (associada a seu compilador JIT) e licença gratuidade open source, que permite a interação de uma comunidade acadêmica de colaboração para desenvolvimento de pacotes matemáticos fundamentais para a computação técnico-científica (COLEMAN et al., 2021).

Tendo em vista o exposto, Julia se afirma como uma forte alternativa a linguagens tradicionalmente usadas por acadêmicos e engenheiros como o MATLAB e demonstra ter grande potencial para escrita de algoritmos já consagrados da análise estrutural, campo de estudo fundamental para diversas engenharias. O presente trabalho apresenta uma proposta de implementação de um programa para análise de treliças planas utilizando tal linguagem.

2 Metodologia

O ensino de engenharia, sendo esta, uma área do conhecimento tecnológico, é presumidamente aplicado e desenvolvido sempre com foco em utilização para solução de algum problema de cunho teórico-científico, dentro deste foco de ensino e aprendizagem existem poderosas ferramentas que podem amparar os estudantes e futuros engenheiros no seu processo de formação, botando o aluno em papel ativo na construção de seu conhecimento a partir da prática (FILHO et al., 2018), dentre estas se encontra a programação.

Cadeiras de programação estão geralmente postas nos ciclos básicos de disciplinas iniciais dos cursos de engenharia sendo introduzidos logo no início da graduação pois é no interim dessa cadeira que são apresentadas as principais formas de resolução de

problemas atuais da engenharia, sendo na engenharia moderna o auxílio computacional indispensável para as práticas que o mercado exige no nível contemporâneo.

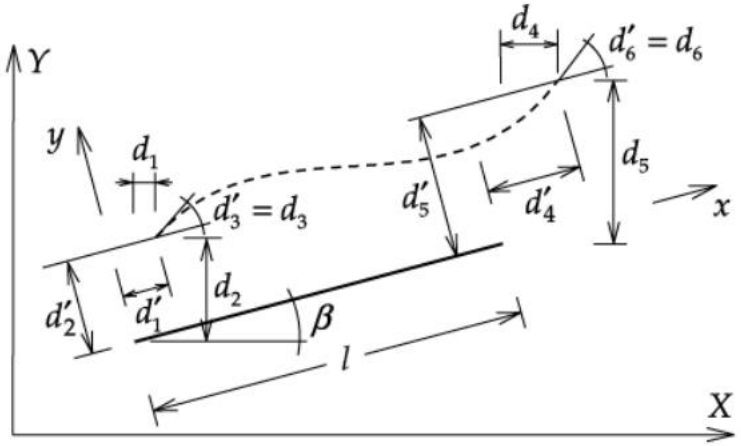
Na área da engenharia estrutural não sendo diferente, o MRD possui apelo intrínseco a sua formulação em implementação computacional. Segundo SILVA e DRUMOND (2012), a ideia principal do algoritmo de MRD é comunicar os carregamentos externos descritos em coordenadas globais a matrizes que descrevam a rigidez estrutural de um certo elemento e consigam a converter, utilizando uma matriz de rotação à coordenadas locais, tais carregamentos em esforços e deslocamentos.

São apresentadas as matrizes citadas nos próximos tópicos.

2.1 Formulação do Método da Rigidez Direta: Matriz de Rotação $[R]$

A Matriz de Rotação ($[R]$) é uma matriz que transforma vetores (também chamados de campos) de forças ou deslocamentos descritos em sistemas de coordenadas ditos globais para locais e vice-versa. No sistema de coordenadas locais é desenvolvida a formulação de rigidez do elemento, e após calculada essa rigidez associada aos graus de liberdade nodais, é feita a decomposição em componentes nos eixos globais que são usados para somar a rigidez de cada elemento, construindo assim a rigidez estrutural.

Figura 2 - Barra com graus de liberdades descritos em coordenadas locais e globais para utilização na matriz $[R]$ *(Fonte:* MARTHA, 2019*).*



$$[R] = \begin{bmatrix} +cos\beta & +sen\beta & 0 & 0 \\ -sen\beta & +cos\beta & 0 & 0 \\ 0 & 0 & +cos\beta & +sen\beta \\ 0 & 0 & -sen\beta & +cos\beta \end{bmatrix}$$

Onde $cos\beta$ e $sen\beta$ são componentes da matriz de transformação de sistemas que usam as informações de comprimento da barra (l) e posições do nó inicial i ($Xi\ e\ Yi$) e final ($Xj\ e\ Yj$) para calcularem os valores conforme mostram as equações *(2), (3) e (4).*

$$l = \sqrt{(Xj - Xi)^2 + (Yj - Yi)^2} \qquad (2)$$

$$cos\beta = \frac{Xj - Xi}{l} \qquad (3)$$

$$sen\beta = \frac{Yj - Yi}{l} \qquad (4)$$

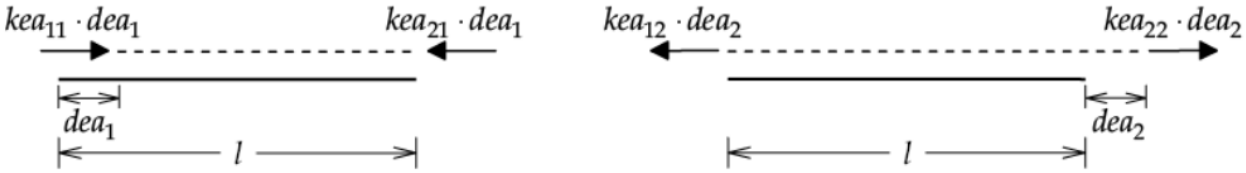
Logo, para construção da matriz $[R]$ basta que se tenham as coordenadas em x e y do nó que forma o elemento.

2.2 Formulação do Método da Rigidez Direta: Matriz de Rigidez $[k]$

A Matriz de Rigidez de um Elemento de Treliça ($[k]$) é função do comprimento de sua barra e da rigidez axial associadas as suas propriedades elásticas (módulo de elasticidade do material, E) e geométricas (área da seção transversal, A). Em sua interpretação física pode-se afirmar que ela relaciona qual seria a força ou momento reativo que surge no grau de liberdade da coluna pela aplicação de um deslocamento associado à sua linha.

Figura 3 - Coeficientes de rigidez de um elemento de treliça (Fonte: (MARTHA, 2019)*).*

$$\{fea\} = [kea] \cdot \{dea\} \quad \rightarrow \quad \begin{Bmatrix} fea_1 \\ fea_2 \end{Bmatrix} = \begin{bmatrix} kea_{11} & kea_{12} \\ kea_{21} & kea_{22} \end{bmatrix} \cdot \begin{Bmatrix} dea_1 \\ dea_2 \end{Bmatrix}.$$



No caso de uma barra de treliça plana para que sejam representados os mesmos graus de liberdade que $[R]$, organiza-se a matriz de rigidez da seguinte forma:

$$k = \begin{bmatrix} +\frac{E.A}{l} & 0 & -\frac{E.A}{l} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{E.A}{l} & 0 & +\frac{E.A}{l} & 0 \\ 0 & 0 & -0 & 0 \end{bmatrix}$$

Logo, para a construção da matriz $[k]$ basta que se tenham as informações de rigidez axiais associadas ao material e a seção de cada elemento, além do seu já calculado comprimento. Com isso é possível calcular a rigidez da estrutura ($[K]$) que remonta a equação *(1)*, cuja inversa é a solução que entrega os deslocamentos dos nós não restringidos e todas as outras respostas da análise estrutural descritas até aqui.

$$[K] = [R]^T . [k] . [R]$$

2.3 Programação Orientada a Objeto

Os paradigmas de programação são formas de estruturação dos códigos escritos segundo uma lógica de utilização da sintaxe e semântica da linguagem, no caso da Programação Orientada a Objeto (POO) existem alguns pilares que norteiam o script, como: encapsulamento, herança, classe e polimorfia (DEMEYER; DUCASSE; NIERSTRASZ, 2003).

No exemplo da análise estrutural são criadas classes que compõe o modelo estrutural, como 'Nó' e 'Elemento', e as propriedades que compõe essas classes de objetos só podem ser alteradas por funções (ou métodos) que acessam e modificam os atributos das mesmas, a isso é chamado de encapsulamento. As funções são capazes de criar outros objetos a partir do original acesso que herde os mesmos atributos que seus antecessor, a isto é denominada herança, porém este objeto filho pode desempenhar processos diferentes quanto exposta as mesmas funções, a isto denomina-se polimorfia.

Figura 4 - Exemplos de classes escritas como 'struct' no Julia para descrever a estrutura a partir de objetos 'Nó' e 'Elemento' (Fonte: Autores, 2023).

```
#01.02. Dados dos nós
struct Carga
    Fx :: Float64
    Fy :: Float64
end
struct Restrição
    Rx :: Bool
    Ry :: Bool
end
struct Deslocamento
    Dx :: Float64
    Dy :: Float64
end
struct Nó
    Identificador :: Int64
    Coordx :: Float64
    Coordy :: Float64
    F :: Carga
    R :: Restrição
    D :: Deslocamento
end
```

```
#01.03. Dados dos elementos
struct Material
    Nome :: String
    E :: Float64
end
struct Seção
    Nome :: String
    A :: Float64
    Mat :: Material
end
struct Elemento
    Identificador :: Int64
    NoI :: Nó
    NoJ :: Nó
    Sec :: Seção
end
```

Um exemplo de aplicação de herança e polimorfia seria quando vários objetos da classe 'Elemento' são criados com diferentes formulações de rigidez e rotação implementadas, fazendo com que o cálculo da rigidez da estrutura utilize de diferentes processos nos mesmos métodos. No caso desse trabalho não foram implementados outros tipos de elementos como de vigas e pórticos.

Essas características do paradigma de POO adicionam eficiência ao código ao passo que diminuem a quantidade de memória usada para alocar variáveis utilizadas apenas em subprocessos locais e permitem a reutilização do código por terceiros que queiram colaborar ou reescrever adaptações da rotina desenvolvida, sendo ambos objetivos objetos de interesse da linguagem Julia.

Sendo Julia uma linguagem de despacho múltiplo (BEZANSON et al., 2012) características como classe, herança e polimorfia são empregadas com muita otimização ao permitirem que o mesmo código trate de diferentes classes de elementos estruturais usando a mesma estrutura e funções, como seria o caso de códigos que fizessem análise de vigas ou pórticos.

No caso do encapsulamento da POO, por se tratar de uma linguagem compilada que trabalha em dois níveis de escopos em seu laço: os locais e globais, Julia aproveita da vantagem do não desperdício de alocação de memória em variáveis que não serão armazenadas para ganhar velocidade de processamento, mostrando notável vantagem sobre linguagens interpretadas muito utilizadas como o MATLAB.

3 Exemplo Numérico

Para exemplificação dos conceitos teóricos desenvolvidos foi aplicada a uma treliça a discretização em nós e elementos conforme mostra a Figura 6, e a esta treliça são aplicadas as funções descritas no código da Figura 5. Com o processamento da estrutura são computados os valores de deslocamentos nodais e os esforços internos da estrutura, permitindo a plotagem de sua forma deformada de cinza e sua forma indeformada de azul conforme

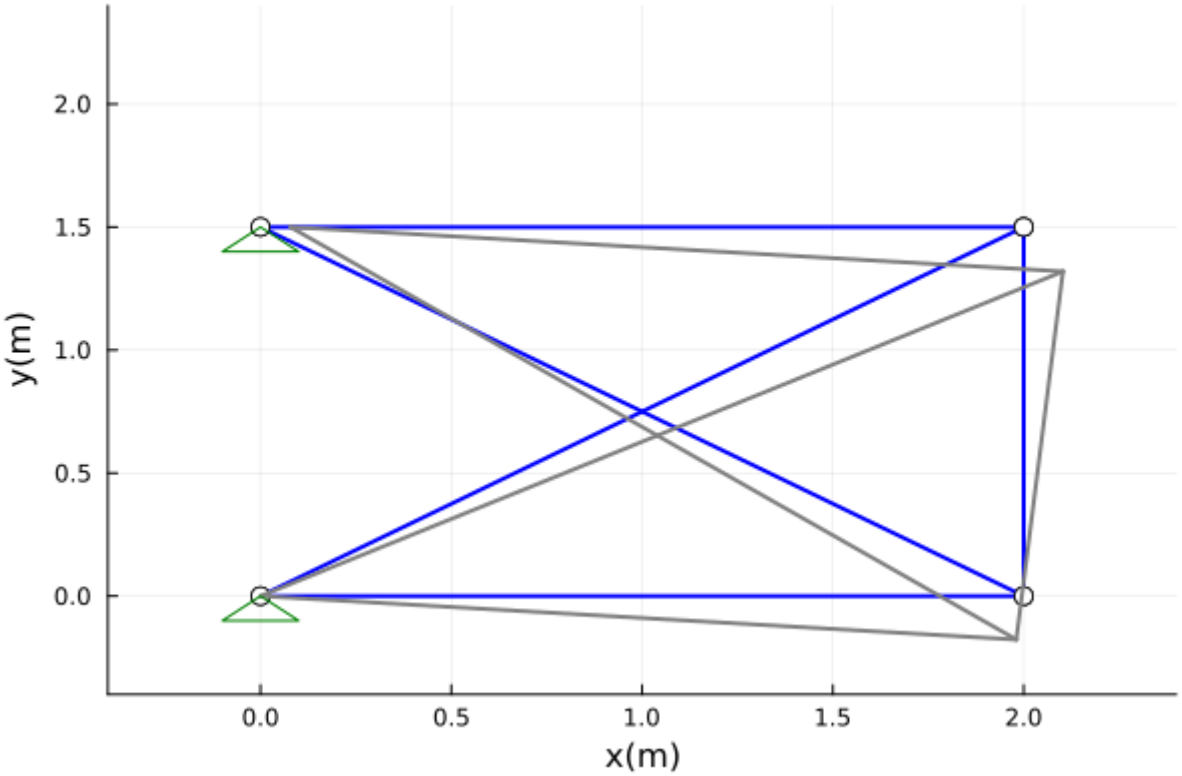
Figura 5 - Funções de construção das Matrizes de Rotação e Rigidez (Fonte: Autores, 2023).

```
#captura das informações dos nós para criação da matriz de rotação
Xi = El.NoI.Coordx
Xj = El.NoJ.Coordx
Yi = El.NoI.Coordy
Yj = El.NoJ.Coordy
Δx = Xj-Xi
Δy = Yj-Yi
L[El.Identificador] = √(Δx^2+Δy^2)
c = Δx/L[El.Identificador]
s = Δy/L[El.Identificador]
R[:,:,El.Identificador] = [
    +c +s 0 0;
    -c +s 0 0;
    0 0 +c +s;
    0 0 -c +s
]
#captura das informações dos elementos para criação das matrizes de rigidez local e global do elemento
E = El.Sec.Mat.E
A = El.Sec.A
kl[:,:,El.Identificador] = (E*A/L[El.Identificador])*[
    +1 0 -1 0;
    0 0 0 0;
    -1 0 +1 0;
    0 0 0 0
]
```

Figura 6 - Descrição da estrutura analisada como 'struct's conforme as classes da POO (Fonte: Autores, 2023).

```
#02. Entrada de Dados
#02.01. Tabela de dados dos nós
#casos de carga
F0 = Carga(0.0,0.0)
F1 = Carga(0.0,-30.0)
F2 = Carga(10.0,-20)
#atribuição de restribuições
R0 = Restrição(0,0)
R = Restrição(1,1)
#prescrição de deslocamentos
D0 = Deslocamento(0.0,0.0)
D = Deslocamento(1.0e-03,0.0)
#nós
No1 = Nó(1,0.0,0.0,F0,R,D0)
No2 = Nó(2,2.0,0.0,F2,R0,D0)
No3 = Nó(3,0.0,1.5,F0,R,D)
No4 = Nó(4,2.0,1.5,F1,R0,D0)
Nos = [No1;No2;No3;No4]
#02.02. Tabela de dados dos elementos
#material utilizado
Aço = Material("Aço",2e08)
#seção transversal
A1 = Seção("Aa=Ab=Ac",10e-4,Aço)
A2 = Seção("Ad=Ae",15e-04,Aço)
#elementos
Elemento1 = Elemento(1,No1,No2,A1)
Elemento2 = Elemento(2,No3,No4,A1)
Elemento3 = Elemento(3,No2,No4,A1)
Elemento4 = Elemento(4,No1,No4,A2)
Elemento5 = Elemento(5,No2,No3,A2)
Elementos = [Elemento1;Elemento2;Elemento3;Elemento4;Elemento5]
```

Figura 7 - Forma deformada e indeformada, de azul e cinza respectivamente (Fonte: Autores, 2023).



Sendo vários os principais parâmetros matriciais gerados, cita-se as matrizes de rigidez da estrutura $[K]$, de rigidez do elemento $[k]$ e de rotação $[R]$, estas identificam a treliça calculada e caracterizam de que forma seus graus de liberdade possuem resistência aos carregamentos impostos e como esses distribuem e comunicam seus esforços membro a membro até seus apoios. No código escrito em Julia seus valores podem ser consultados a partir do objeto 'Matrizes' conforme mostrado no prompt do programa.

Figura 8 - Parâmetros matriciais calculados (Fonte: Autores, 2023).

```
julia> Matrizes.R
4×4×5 Array{Float64, 3}:
[:, :, 1] =
  1.0  0.0   0.0  0.0
 -1.0  0.0   0.0  0.0
  0.0  0.0   1.0  0.0
  0.0  0.0  -1.0  0.0

[:, :, 2] =
  1.0  0.0   0.0  0.0
 -1.0  0.0   0.0  0.0
  0.0  0.0   1.0  0.0
  0.0  0.0  -1.0  0.0

[:, :, 3] =
  0.0  1.0   0.0  0.0
 -0.0  1.0   0.0  0.0
  0.0  0.0   0.0  1.0
  0.0  0.0  -0.0  1.0

[:, :, 4] =
  0.8  0.6   0.0  0.0
 -0.8  0.6   0.0  0.0
  0.0  0.0   0.8  0.6
  0.0  0.0  -0.8  0.6

[:, :, 5] =
 -0.8  0.6   0.0  0.0
  0.8  0.6   0.0  0.0
  0.0  0.0  -0.8  0.6
  0.0  0.0   0.8  0.6
```

```
julia> Matrizes.kl
4×4×5 Array{Float64, 3}:
[:, :, 1] =
  100000.0  0.0  -100000.0  0.0
       0.0  0.0        0.0  0.0
 -100000.0  0.0   100000.0  0.0
       0.0  0.0        0.0  0.0

[:, :, 2] =
  100000.0  0.0  -100000.0  0.0
       0.0  0.0        0.0  0.0
 -100000.0  0.0   100000.0  0.0
       0.0  0.0        0.0  0.0

[:, :, 3] =
  1.33333e5  0.0  -1.33333e5  0.0
  0.0        0.0   0.0        0.0
 -1.33333e5  0.0   1.33333e5  0.0
  0.0        0.0   0.0        0.0

[:, :, 4] =
  120000.0  0.0  -120000.0  0.0
       0.0  0.0        0.0  0.0
 -120000.0  0.0   120000.0  0.0
       0.0  0.0        0.0  0.0

[:, :, 5] =
  120000.0  0.0  -120000.0  0.0
       0.0  0.0        0.0  0.0
 -120000.0  0.0   120000.0  0.0
       0.0  0.0        0.0  0.0
```

```
julia> Matrizes.K
8×8 Matrix{Float64}:
  176800.0  -57600.0            0.0      0.0       -100000.0      0.0   -76800.0   57600.0
  -57600.0       1.76533e5       0.0     -1.33333e5       0.0      0.0    57600.0  -43200.0
       0.0       0.0        176800.0   57600.0        -76800.0  -57600.0  -100000.0       0.0
       0.0      -1.33333e5   57600.0       1.76533e5   -57600.0  -43200.0        0.0       0.0
 -100000.0       0.0        -76800.0  -57600.0        176800.0   57600.0        0.0       0.0
       0.0       0.0        -57600.0  -43200.0         57600.0   43200.0        0.0       0.0
  -76800.0   57600.0       -100000.0      0.0             0.0       0.0   176800.0  -57600.0
   57600.0  -43200.0             0.0      0.0             0.0       0.0    -57600.0   43200.0
```

4 Conclusão

A análise estrutural moderna é feita a partir de programas computacionais, sendo estes objetos de estudos fundamentais para a formação de futuros engenheiros, que buscam em seus períodos de graduação que buscam desenvolver senso prático de aplicação de seus conhecimentos teóricos e habilidades que exercitem seu papel ativo na sugestão de soluções, como é o caso da programação.

O Método da Rigidez Direta é uma metodologia da análise estrutural que possui muito apelo a implementação computacional, sendo, uma boa ferramenta para o ensino das disciplinas voltadas para tal ramo de estudo. A linguagem Julia por possuir sintaxe e semântica próxima ao do MATLAB se apresenta como uma boa alternativa para forma de implementação de tal problema e ainda conta com vantagens como velocidade de processamento e gratuidade no acesso.

Dentro do exposto a implementação do programa de análise de treliças planas em Julia mostra-se como um bom objeto de aprendizado, visto que se debruça-se sobre questões fundamentais da análise estrutural como rigidez, esforços e deslocamentos além de permitir que o futuro engenheiro entenda de maneira um pouco mais profunda como são realizados os cálculos de estruturais por trás dos programas usados no dia a dia do mercado.

REFERÊNCIAS

BEZANSON, J. et al. Julia: A Fast Dynamic Language for Technical Computing. 23 set. 2012.

LOZOVEY, Ana C. R. Ensino e aprendizagem de estruturas: simulações. 50º Congresso-Brasileiro de Educação em Engenharia e V Simpósio Internacional de Educação em Engenharia da ABENGE, 2022. **Anais.** Disponível em: http://www.abenge.org.br/sis_submetidos.php?acao=abrir&evento=COBENGE22&codigo=COBENGE22_00300_00003793.pdf. Acesso em 11 jun.2023.

COLEMAN, C. et al. Matlab, Python, Julia: What to Choose in Economics? **Computational Economics**, v. 58, n. 4, p. 1263–1288, 26 dez. 2021.

DEMEYER, S.; DUCASSE, S.; NIERSTRASZ, O. **Object-Oriented Reengineering Patterns**. 8. ed. [s.l.] Elsevier, 2003.

SANTOS, Ana C. dos.; SILVEIRA, Evilly R. H. da.; Silva, Sebastião S. da.; SALVIANO, Adenilda T. Utilização de uma biblioteca de elementos finitos para a análise de problemas no estado uniaxial, plano e tridimensional de tensão. 50º Congresso-Brasileiro de Educação em Engenharia e V Simpósio Internacional de Educação em Engenharia da ABENGE, 2022. **Anais.** Disponível em: http://www.abenge.org.br/sis_submetidos.php?acao=abrir&evento=COBENGE22&codigo=COBENGE22_00309_00003965.pdf. Acesso em 11 jun.2023.

FILHO, A. V. et al. Ensino de Engenharia Civil. **Revista Docência do Ensino Superior**, v. 8, n. 2, p. 30–43, 10 dez. 2018.

MARTHA, L. **Análise matricial de estruturas com orientação a objetos**. 1. ed. Rio de Janeiro: Elsevier, 2019.

SILVA, A. R.; DRUMOND, F. P. Desenvolvimento de um software para análise estrutural de sistemas reticulados espaciais usando o método dos deslocamentos. **Exacta**, v. 10, n. 3, p. 319–324, 21 dez. 2012.

XIAO, L. et al. Julia Language in Computational Mechanics: A New Competitor. **Archives of Computational Methods in Engineering**, 2021.

IMPLEMENTATION OF MATRIX ANALYSIS OF PLAN TRUSS USING OPEN SOURCE LANGUAGE AND OBJECT-ORIENTATION FOR STRUCTURE TEACHING

Abstract: *The matrix analysis of structures is the formalization of a structural analysis method known as the displacement method (or stiffness method) adapted for computational implementation. This adaptation is made aiming at the ease of implementation of the theory in computer programming language codes so that all the calculations that were learned manually in theoretical disciplines can now be applied in scripts that stimulate creativity and logical thinking in engineering students. The present work proposes the implementation of this analysis (also called the method of direct rigidity) in a free and open source language, Julia, which appeared a few years ago and already has concrete data on processing efficiency, as well as syntax and semantics very similar to other languages commonly used for technical and scientific computing, all in a programming paradigm known as Object Oriented Programming*

Keywords: *Direct Rigidity Method, Programming, Lattices, Julia Language and Object Orientation.*