UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ALISSON CEZAR CLAUDIO

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE ALGORITMO BASEADO NO MÉTODO DOS ELEMENTOS FINÍTOS PARA ANÁLISE ELÁSTICA LINEAR DE BARRAS SOB TORÇÃO

CAMPO MOURÃO

2018

ALISSON CEZAR CLAUDIO

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE ALGORITMO BASEADO NO MÉTODO DOS ELEMENTOS FINÍTOS PARA ANÁLISE ELÁSTICA LINEAR DE BARRAS SOB TORÇÃO

Trabalho de Conclusão de Curso de Graduação apresentado à Disciplina de Trabalho de Conclusão de Curso 2, do Curso Superior em Engenharia Civil do Departamento Acadêmico de Construção Civil – DACOC – da Universidade Tecnológica Federal do Paraná – UTFPR, para obtenção do título de bacharel em engenharia civil.

Orientador: Prof. Dr. Leandro Waidemam

CAMPO MOURÃO

2018

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Campo Mourão
Diretoria de Graduação e Educação Profissional
Departamento Acadêmico de Construção Civil
Coordenação de Engenharia Civil

TERMO DE APROVAÇÃO

Trabalho de Conclusão de Curso

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE ALGORITMO BASEADO NO MÉTODO DOS ELEMENTOS FINITOS PARA ANÁLISE ELÁSTICA LINEAR DE BARRAS SOB TORÇÃO

por

Alisson Cezar Claudio

Este Trabalho de Conclusão de Curso foi apresentado às 15h30min do dia 28 de Junho de 2018 como requisito parcial para a obtenção do título de ENGENHEIRO CIVIL, pela Universidade Tecnológica Federal do Paraná. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Dr. Jeferson Rafael Bueno
(UTFPR)

Prof. Dr. Marcelo Rodrigo Carreira
(UTFPR)

Prof. Dr. Leandro Waidemam
(UTFPR)
Orientador

Responsável pelo TCC: **Prof. Me. Valdomiro Lubachevski Kurta**

Coordenador do Curso de Engenharia Civil:
Prof. Dr. Ronaldo Rigobello

A Folha de Aprovação assinada encontra-se na Coordenação do Curso.

A minha família, dedico este trabalho.

AGRADECIMENTOS

Gostaria de agradecer aos meus pais, James e Angela, por todo o apoio durante a minha vida, sempre incentivando meus estudos, e pelo carinho sempre presente. Minha formação e toda a experiência de vida que tive não seria possível sem a ajuda de vocês.

Agradeço a minha namorada Flávia pelo carinho e pela atenção que foram muito importantes para a minha formação. Obrigado pelo amor que você me dá a todo momento, e pelos momentos de alegria que tivemos durante esses anos que, mesmo longe, foram incríveis e são muito importantes para mim.

Agradeço aos professores da UTFPR, muito importantes na minha formação. Em especial, ao professor Dr. Leandro, muito importante para a execução deste trabalho, e um dos melhores professores que conheci.

Agradeço também a CAPES pela bolsa de estudos que me possibilitou viver e estudar em Budapeste, numa das melhores universidades da Europa, experiência que me fez uma pessoa melhor, e ainda mais motivado a continuar estudando e buscando conhecimento.

Um agradecimento imenso aos amigos que fiz em Campo Mourão, pelo convívio durante todos esses anos de estudo, pelos momentos de alegria, pela amizade que guardarei para sempre.

Por fim, a todos que de alguma forma contribuíram para a minha formação, muito obrigado.

RESUMO

CLAUDIO, Alisson Cezar. **Desenvolvimento e implementação de algoritmo baseado no método dos elementos finitos para análise elástica linear de barras sob torção**. 2018. 55 f. Trabalho de Conclusão de Curso (Graduação) - Curso de Engenharia Civil, Universidade Tecnológica Federal do Paraná. Campo Mourão, 2018.

Este trabalho teve como objetivo a implementação de um código computacional baseado no Método dos Elementos Finitos capaz de analisar o comportamento elástico linear de barras sob torção. Para tal, a formulação foi desenvolvida utilizando-se o elemento finito linear composto por dois nós, cada um com um grau de liberdade. Em termos de carregamento, dois tipos foram implementados: torques nodais e torque linearmente distribuído ao longo do elemento. A partir do Princípio dos Trabalhos Virtuais foi obtida a equação de equilíbrio do elemento, envolvendo o vetor de forças nodais equivalentes, o vetor de giros nodais e a matriz de rigidez do elemento, possibilitando a implementação de um código capaz de obter giros e reações de apoio de problemas relacionados. Por fim, exemplos foram apresentados a fim de validar o método e a correta implementação do código, comparando os resultados obtidos com os fornecidos por *software* específico da área ou com resultados obtidos nas referências consultadas.

Palavras-chave: Método dos Elementos Finitos, Análise elástica linear, Barras sob torção.

ABSTRACT

CLAUDIO, Alisson Cezar. **Development and implementation of an algorithm based on the finite element method for elastic-linear Analysis of bars under torsion.** 2018. 55 f. Trabalho de Conclusão de Curso (Graduação) – Curso de Engenharia Civil, Universidade Tecnológica Federal do Paraná. Campo Mourão, 2018.

This work had the objective of developing an algorithm based on the Finite Element Method capable of analyzing the elastic-linear behavior of bars under pure torsion. For such purpose, the formulation was developed using a linear-finite element composed of two nodes, with one freedom degree each. In terms of load, two types were implemented: Nodal torques and linearly distributed torques along the element. From the virtual work principle, the equation of equilibrium was obtained, involving the equivalent nodal force vector, the nodal twist vector, and the element stiffness matrix, enabling the implementation of a code capable of obtaining the twists and reaction values of problems related to this work's subject. In the end, some examples were presented and solved by the algorithm, with results compared to the presented by the references consulted or by software of the study field, in order to validate the implementation.

Keywords: Finite Element Method, Elastic-linear analysis, Bars under torsion.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

F	Força pontual vertical
T	Torque
$\phi(x)$	Giro em função do comprimento x da barra
ρ	Distância radial
$\Delta\phi$	Variação de giro
θ'	Ângulo interno de elemento de superfície após a deformação
γ	Ângulo de deformação por cisalhamento
Δx	Comprimento do elemento de superfície
c	Raio
τ	Tensão de cisalhamento
G	Módulo de elasticidade transversal
J	Momento polar de inércia
K	Constante que representa o momento polar de inércia
U_e	Trabalho externo
U_i	Trabalho interno
m_x	Momento nodal
ϕ_x	Giro nodal
U_e^*	Trabalho virtual externo
U_i^*	Trabalho virtual interno
$\delta\gamma^*$	Deformação virtual por cisalhamento
φ	Função interpoladora
A	Área da barra
V	Volume da barra
L	Comprimento da barra
$\{\overline{F}\}_e$	Vetor de momentos nodais do elemento
$\{\overline{K}\}_e$	Matriz de rigidez do elemento
$\{\overline{u}\}_e$	Vetor de giros nodais do elemento

SUMÁRIO

1 INTRODUÇÃO

Desde os primórdios da humanidade, a engenharia tem um papel importante na sociedade. Seja em pequenas construções, como simples moradias, até grandes obras que desafiam a capacidade dos projetistas, como, por exemplo, a usina hidrelétrica de Itaipu, sempre existiram análises, pesquisas, projetos e cálculos a serem realizados para garantir a integridade e funcionalidade das estruturas em questão.

Com o avanço da tecnologia, a experiência adquirida pelos profissionais da construção civil e criatividade dos projetistas, puderam-se desenvolver estruturas cada vez mais complexas, propiciando maiores desafios aos engenheiros em seus trabalhos de análise estrutural. Süssekind (1987, p. 1) define a análise estrutural como sendo a área da mecânica que estuda os esforços e os deslocamentos que geram, quando as estruturas ficam sob efeito de cargas, variações térmicas ou movimento de apoios.

Os novos desafios exigiram que a metodologia empregada nos processos de cálculo estrutural também evoluísse, o que foi possível, particularmente, pela revolução computacional dos últimos tempos, representada pelo aumento na velocidade de processamento de dados e na capacidade de armazenamento destes, aliado ao desenvolvimento de métodos numéricos cada vez mais precisos.

Dentre os métodos numéricos, o surgimento do Método dos Elementos Finitos (MEF) foi de grande importância para a solução de problemas relacionados à física e engenharia, particularmente a estrutural, abordada neste trabalho. O princípio básico do método é se utilizar da discretização do elemento estrutural analisado em elementos finitos menores, interligados por nós, criando “pequenos problemas” a serem resolvidos, os quais, unidos, possibilitam a obtenção da solução do problema original (LOGAN, 2007).

Tendo em vista os diversos problemas existentes na engenharia civil o presente trabalho tem como objetivo apresentar um código computacional baseado no MEF capaz de realizar análises estruturais de barras submetidas à torção, em regime elástico linear.

2 OBJETIVOS

2.1 OBJETIVO GERAL

Implementar um código computacional baseado no Método dos Elementos Finitos capaz de analisar o comportamento elástico linear de barras submetidas ao efeito da torção.

2.2 OBJETIVOS ESPECÍFICOS

- Compreender a formulação do método em questão, particularmente aplicado ao comportamento elástico linear de barras sob torção.
- Desenvolver um código computacional baseado na linguagem *PYTHON* que utilize a formulação descrita para a solução de problemas de barras sob torção e forneça, como resultados finais, giros e reações de apoio.
- Avaliar o código desenvolvido por meio de exemplos descritos em referências bibliográficas da área, comparando os resultados obtidos com os dados pelos autores, bem como com os fornecidos por *software* comercial.

3 JUSTIFICATIVA

No presente trabalho, escolheu-se analisar a torção em barras por ser um problema aplicável em diversas áreas da engenharia, como em eixos transmissores de torque, barras de direção, vigas submetidas a momentos torsores, sejam estes provenientes de forças externas ou em decorrência de deformações dos sistemas estruturais em que estão contidas, entre outros.

Na engenharia civil, pode-se verificar exemplos de torção em vigas, principalmente presentes em pórticos espaciais e grelhas, estruturas essas encontradas em edificações de concreto armado, madeira e aço. Em tais análises, forças e momentos aplicados em um ponto da estrutura resultam, com a transmissão dos esforços, na torção de vigas ligadas aquelas que resistem aos momentos fletores e cortantes.

Considerando a viga em regime elástico linear, a proporcionalidade entre as tensões e deformações é relacionada através da Lei de Hooke (HIBBELER, 2010), garantindo assim que os deslocamentos possam ser relacionados aos esforços de forma linear, ou seja, com um módulo de elasticidade constante.

A análise numérica é muito utilizada na solução de problemas em que a solução analítica é inviável de ser encontrada. Logo, os métodos numéricos podem ser usados para encontrar soluções aproximadas de tais problemas (BUFFONI, 2002). E, um dos métodos mais utilizados na engenharia é o Método dos Elementos Finitos (MEF).

Como já foi dito anteriormente, o MEF é um método de solução aproximada que visa a resolução de problemas com difícil solução analítica, dividindo-os em problemas menores. Soriano (2009, p. 8) destaca o uso do método descrito por diversos *softwares* comerciais devido à “facilidade de generalização, programação e uso”.

Assim, este trabalho utilizará do Método dos Elementos Finitos para a solução de problemas envolvendo barras sob torção, realizando a aplicação computacional de tal método na linguagem *PYTHON,* a qual é muito versátil e possui diversas funcionalidades.

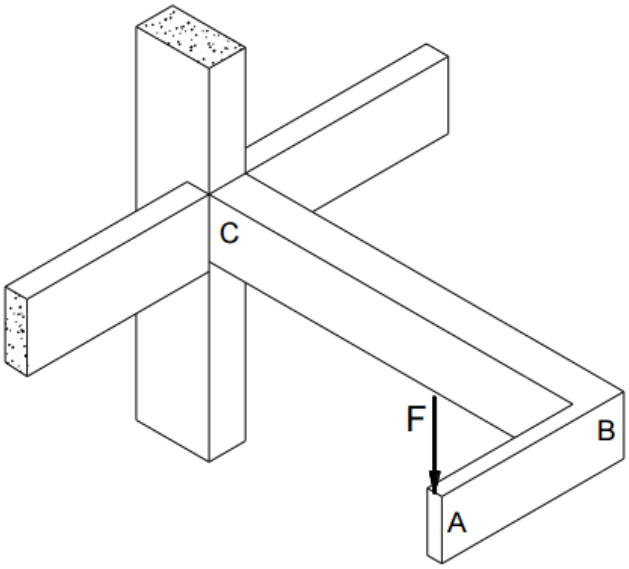
Por fim, o trabalho final é visto como uma contribuição ao desenvolvimento da área de métodos numéricos e programação, ficando disponível com o código fonte para estudo e testes futuros.

4 REFERENCIAL TEÓRICO

4.1 TORÇÃO DE VIGAS

Na construção civil, alguns exemplos de torção podem ser verificados, em geral, combinados com força cortante e momentos. Por menores que sejam os esforços, precisam ser considerados para garantir que os estados limites último e de serviço sejam verificados corretamente. Como exemplo disso, pode-se analisar a Figura 1, de Bastos (2015), onde a aplicação de uma força na viga AB gera momentos torsores na viga BC.

Figura 1 - Estrutura de concreto carregado com uma força pontual "F".

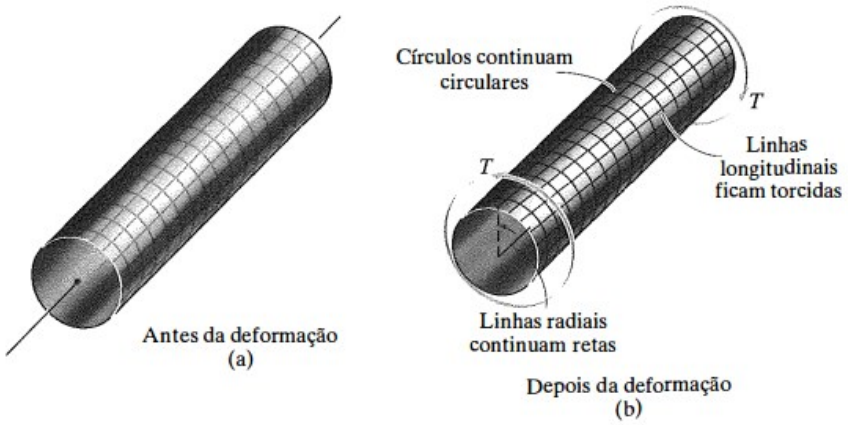


Fonte: Bastos (2015, p. 1).

Coulomb estudou a torção em 1784 enquanto analisava o ângulo de torção gerado por cargas elétricas em uma esfera suspensa por um fio, estudo esse que foi utilizado para descrever o comportamento de elementos submetidos à torção (FIGUEIREDO, 2014).

Quando um torque é aplicado a uma barra circular, esta tende a ser distorcida longitudinalmente, como visto na Figura 2 (HIBBELER, 2010). A barra é representada com secções formando uma grade para demonstrar o efeito da torção ao longo do seu comprimento.

Figura 2 - Barra submetida a um torque "T".

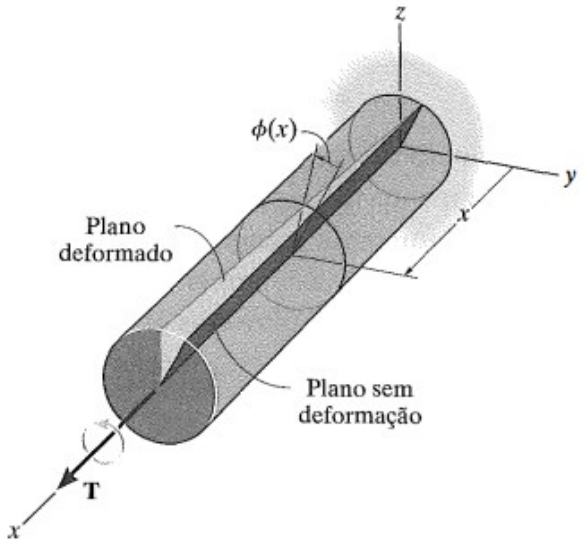


Fonte: Hibbeler (2010, p. 125).

É possível verificar que a seção transversal permanece circular, assim como os círculos ao longo da barra, além das linhas radiais que continuam retas, enquanto que as linhas longitudinais são distorcidas, girando em torno da barra. Logo, segundo Hibbeler (2010, p. 125), para pequenos giros, o comprimento e o raio do elemento não mudam.

Ainda, se uma das extremidades da barra estiver engastada, e o torque for aplicado à outra extremidade, tem-se um giro $\phi(x)$ variando ao longo da barra, sendo este igual à zero no engaste e máximo na extremidade livre, como mostrado na Figura 3.

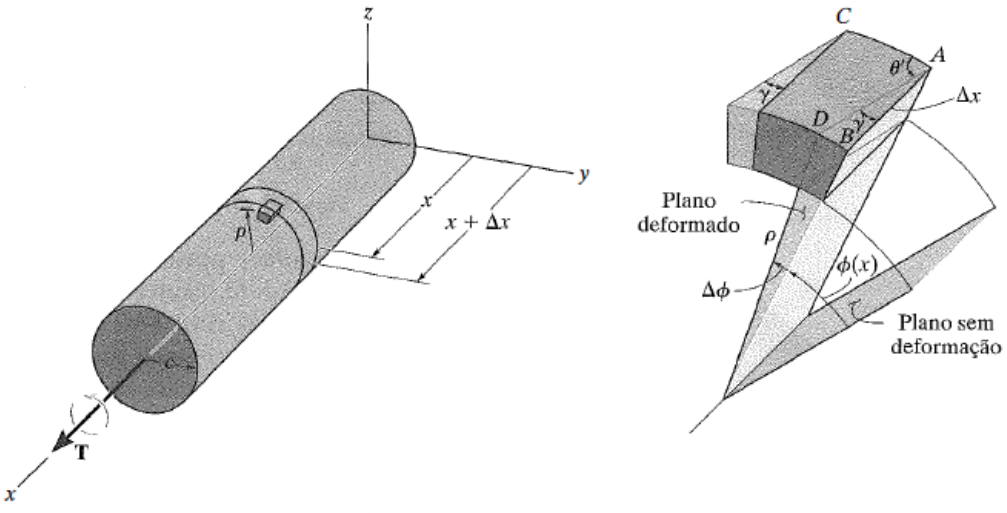
Figura 3 - Variação do ângulo de torção ao longo da barra.



Fonte: Hibbeler (2010, p. 126).

Isolando um pequeno elemento a uma distância radial ρ do eixo da barra mostrada na Figura 3, é possível observar a rotação nas faces posterior e anterior do mesmo, de $\phi(x)$, e $\phi(x)+\Delta\phi$, respectivamente (HIBBELER, 2010, p.125). A seguir, a Figura 4 mostra esse efeito.

Figura 4 - Barra e elemento analisado.



Fonte: Hibbeler (2010, p. 126).

A tensão do cisalhamento surge da diferença entre as rotações, $\Delta\phi$. O cálculo dessa tensão vem da análise da distorção que o elemento sofre, sendo o ângulo entre BA e AC de 90º antes da rotação e o ângulo θ' entre DA e AC depois da rotação do elemento. Ainda, segundo Hibbeler (2010), pela definição de deformação por cisalhamento, tem-se:

$$\gamma = \frac{\pi}{2} - \lim \theta' \tag{1}$$

O ângulo γ, que está definido entre as retas DA e AB, pode ser relacionado com comprimento Δx e a diferença no ângulo de rotação entre os planos sem deformação e o deformado $\Delta\phi$ (HIBBELER, 2010, p. 126). Assim, sendo $\Delta x \to dx$ e $\Delta\phi \to d\phi$, tem-se:

$$BD = \rho \cdot d\phi = dx \cdot \gamma \tag{2}$$

Isolando γ:

$$\gamma = \rho \frac{d\phi}{dx} \tag{3}$$

Como a razão entre $d\phi$ e dx não varia na seção (HIBBELER, 2010), a Equação (3) indica que a deformação varia linearmente em função de ρ, sendo zero no centro da barra e máxima na superfície da mesma. Sendo $d\phi/dx = \gamma/\rho = \gamma_{máx}/c$, a equação da deformação por cisalhamento é:

$$\gamma = \left(\frac{\rho}{c}\right)\gamma_{máx} \tag{4}$$

Com o material em regime elástico-linear, a lei de Hooke é válida, ou seja, a deformação por cisalhamento é linearmente proporcional à tensão de cisalhamento, ao longo do raio da barra. Assim, substituindo a lei de Hooke, $\tau = G\gamma$, com a Equação (4):

$$\tau = \left(\frac{\rho}{c}\right)\tau_{máx} \tag{5}$$

Hibbeler (2010) aplica a condição de equilíbrio na qual o torque resultante aplicado na barra é igual ao torque resultante das tensões na seção transversal da mesma. A Equação abaixo descreve o equacionamento feito pelo autor.

$$T = \int_A \rho\left(\tau dA\right) = \int_A \rho\left(\frac{\rho}{c}\right)\tau_{máx} dA \tag{6}$$

Como $\tau_{máx}/c$ é constante:

$$T = \frac{\tau_{máx}}{c}\int_A \rho^2 dA \tag{7}$$

A integral na Equação (7) depende apenas de ρ que é relacionado à geometria da barra e representa o momento polar de inércia, representado por J, possibilitando reescrever esta equação como:

$$\tau_{máx} = \frac{T \cdot c}{J} \tag{8}$$

Visto que c é o raio da barra, e que a tensão varia linearmente de zero a máxima ao longo do raio, as Equações (5) e (8) representam a tensão relacionada à distância ρ do eixo da barra:

$$\tau = \frac{T \cdot \rho}{J} \tag{9}$$

Segundo Hibbeler (2010), o momento polar de inércia de um eixo cilíndrico maciço é calculado através da seguinte equação:

$$J = \frac{\pi}{2} c^4 \tag{10}$$

E para um tubo com seção circular de raio interno c_o e raio externo c_i, o momento polar será:

$$J = \frac{\pi}{2}\left(c_o^4 - c_i^4\right) \tag{11}$$

Para o ângulo de torção, tem-se a Equação (3) que relaciona a deformação por cisalhamento e a rotação relativa entre as duas faces de um elemento localizado a uma distância dx do engaste da barra. Utilizando novamente a lei de Hooke e a equação do torque (9), tem-se:

$$d\phi = \frac{T(x)dx}{J(x)\cdot G} \tag{12}$$

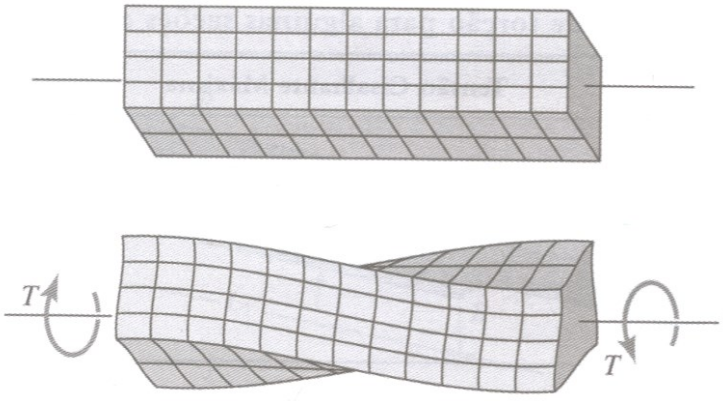
A integração pelo comprimento da barra fornece o ângulo de torção de uma extremidade em relação à outra. Com a equação do torque em relação ao comprimento da barra, pode-se encontrar o giro com a integração. Como em geral o torque é constante pela barra (HIBBELER, 2010), tem-se:

$$\phi = \frac{T\cdot L}{J\cdot G} \tag{13}$$

4.1.1 Barras de Seção Não Circular

Para barras não circulares, a torção provoca deformações que ocorrem fora do plano da seção transversal, mudando a sua forma original. Assim, a barra sofre do efeito de empenamento, como visto na Figura 5, onde uma barra de seção retangular é submetida à torção. Diferente da barra de seção circular, é possível perceber a seção inicial sendo distorcida.

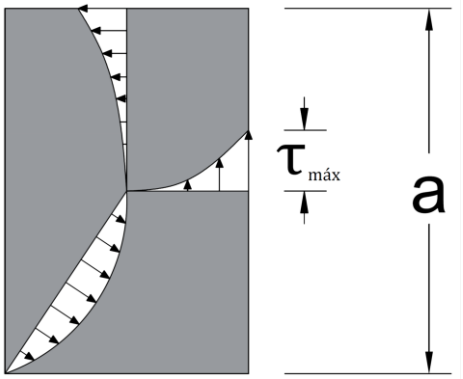
Figura 5 - Barra retangular submetida à torção.



Fonte: Ugural (2009, p. 187).

Além disso, as tensões cisalhantes não variam linearmente com relação à distância do eixo central da seção. Para uma seção retangular, por exemplo, as tensões variam de forma diferente para cada direção analisada, como mostra a Figura 6, de Ugural (2009).

Figura 6 - Distribuição de tensão cisalhante em uma seção retangular sob torção.



Fonte: Adaptado de Ugural (2009, p. 187).

Enquanto que na seção circular a tensão de cisalhamento máxima ocorre no ponto mais afastado do seu centro, tal ponto, na seção retangular, tem tensão nula. Como visto na Figura 6, as tensões variam de forma não linear e em diferentes padrões, de acordo com a direção analisada.

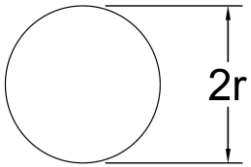
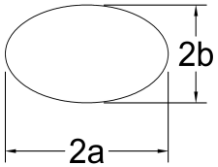
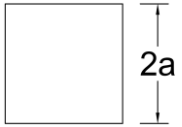
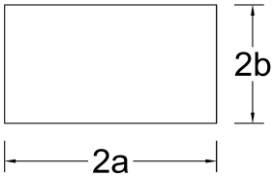
Logo, para seções não circulares, formulações mais complexas com análises experimentais são necessárias para descrever as tensões atuantes de forma precisa.

Young e Budynas (2002) apresentam uma tabela que relaciona diversas formas geométricas de seção transversal com seus respectivos pontos e valores de tensão de cisalhamento máximo, além de um fator K que aproxima um momento polar de inércia, podendo ser utilizado para o cálculo do ângulo de torção de barras com tais formas. Segundo os autores, a equação do giro é representada por:

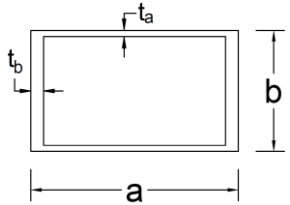
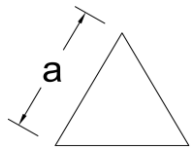
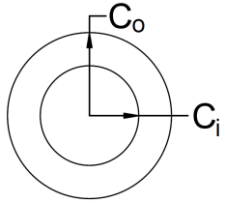
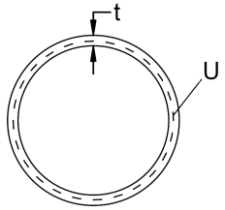
$$\phi = \frac{T \cdot L}{K \cdot G} \tag{14}$$

O fator K representa o momento polar de inércia para barras de seção circular e pode ser calculado de acordo com o quadro mostrado por Young e Budynas (2002, p. 401). O Quadro (1) relaciona o valor de K para as seções mais comuns.

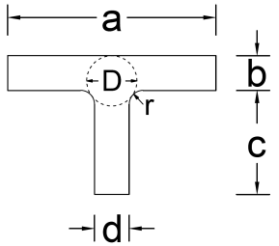
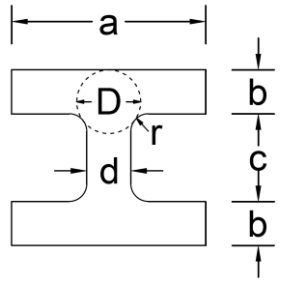
Quadro 1 - Relações entre seção transversal, fator *K* e tensão de cisalhamento. **(continua)**

Forma e dimensões da seção transversal	Fórmula de *K*	Cisalhamento
1. Seção circular 	$K = \frac{1}{2}\pi \cdot r^4$	$\tau_{máx} = \frac{2T}{\pi \cdot r^3}$ na borda da seção.
2. Seção elíptica 	$K = \frac{\pi \cdot a^3 \cdot b^3}{a^2 + b^2}$	$\tau_{máx} = \frac{2T}{\pi \cdot a \cdot b^2}$ no fim do eixo menor.
3. Seção quadrada 	$K = 2,25 \cdot a^4$	$\tau_{máx} = \frac{0,601 \cdot T}{a^3}$ no ponto médio de cada lado.
4. Seção retangular 	$K = a \cdot b^3 \cdot \left[\frac{16}{3} - 3,36 \cdot \frac{b}{a}\left(1 - \frac{b^4}{12a^4}\right)\right]$	$\tau_{máx} = \frac{3T}{8 \cdot a \cdot b^2}\left[1 + 0,6095\frac{b}{a} + 0,8865\left(\frac{b}{a}\right)^2 - 1,8023\left(\frac{b}{a}\right)^3 + 0,91\left(\frac{b}{a}\right)^4\right]$

(continuação)

Forma e dimensões da seção transversal	Fórmula de *K*	Cisalhamento
5. Seção retangular vazada de parede fina 	$K = \frac{2 \cdot t_a \cdot t_b \cdot (a - t_b)^2 \cdot (b - t_a)^2}{a \cdot t_b + b \cdot t_a - t_a^2 - t_b^2}$	$\tau_{médio} = \frac{T}{2 \cdot t_b \cdot (a - t_b) \cdot (b - t_a)}$, próximo à metade do comprimento dos lados menores. $\tau_{médio} = \frac{T}{2 \cdot t_a \cdot (a - t_b) \cdot (b - t_a)}$, próximo à metade do comprimento dos lados maiores.
6. Seção triângulo equilátero 	$K = \frac{a^4 \cdot \sqrt{3}}{80}$	$\tau_{máx} = \frac{20 \cdot t}{a^3}$ no ponto médio de cada lado
7. Seção circular vazada 	$K = \frac{1}{2} \cdot \pi \cdot (c_o^4 - c_i^4)$	$\tau_{máx} = \frac{2 \cdot T \cdot c_o}{\pi \cdot (c_o^4 - c_i^4)}$ na borda exterior
8. Tubo de parede fina 	$K = \frac{4 \cdot A^2 \cdot t}{U}$, sendo *U* o comprimento do círculo médio entre as bordas do tubo (círculo seccionado da figura).	$\tau_{médio} = \frac{T}{2 \cdot t \cdot A}$ (tensão é aproximadamente uniforme se *t* for pequeno)

(conclusão)

Forma e dimensões da seção transversal	Fórmula de *K*	Cisalhamento
9. Seção "T", sendo D : diâmetro do maior círculo inscrito, r : raio do adoçamento $t = b$ se $b < d$; $t = d$ se $d < b$; $t_1 = b$ se $b > d$; $t_1 = d$ se $d > b$ 	$K = 2 \cdot K_1 + K_2 + 2 \cdot \alpha \cdot D^4$, onde: $K_1 = a \cdot b^3 \left[\frac{1}{3} - 0,21 \frac{a}{b} \left(1 - \frac{b^4}{12 \cdot a^4} \right) \right]$ $K_2 = c \cdot d^3 \left[\frac{1}{3} - 0,105 \frac{d}{c} \left(1 - \frac{d^4}{192 \cdot c^4} \right) \right]$ $\alpha = \frac{t}{t_1} \left(0,15 + 0,1 \frac{r}{b} \right)$ $D = \frac{(b+r)^2 + r \cdot d + d^2/4}{(2 \cdot r + b)}$, para $d < 2(b+r)$	Em um ponto onde a curvatura é negativa (borda da seção côncava ou reentrante), a tensão máxima é dada de forma aproximada por: $\tau_{máx} = G \cdot \frac{\theta}{L} \cdot C$ ou $\tau_{máx} = \frac{T}{K} \cdot C$, onde: $C = \frac{D}{1 + \frac{\pi^2 \cdot D^4}{16 \cdot A^2}} \cdot \left\{ 1 + \left[0,118 \cdot \ln \left(1 - \frac{D}{2 \cdot r} \right) - 0,238 \frac{D}{2r} \right] \cdot \tanh \left(\frac{2\phi}{\pi} \right) \right\}$ $\phi =$ um ângulo positivo pelo qual a tangente da fronteira gira em torno da porção reentrante, medida em radianos (aqui r é negativo).
10. Seção "I" 	$K = 2 \cdot K_1 + K_2 + 2 \cdot \alpha \cdot D^4$, onde: $K_1 = a \cdot b^3 \left[\frac{1}{3} - 0,21 \frac{a}{b} \left(1 - \frac{b^4}{12 \cdot a^4} \right) \right]$ $K_2 = \frac{1}{3} c \cdot d^3$ $\alpha = \frac{t}{t_1} \left(0,15 + 0,1 \frac{r}{b} \right)$ D, t e t_1 iguais à seção 9.	

Fonte: Adaptado de Young e Budynas (2002, p. 401).

Young e Budynas (2002, p. 383) versam sobre os métodos utilizados para as formulações de K e τ. São eles:

- As fórmulas para as seções de 1 a 3, 6 e 7 são baseadas em análises matemáticas.
- Para a seção 4, as equações são dadas de forma simplificada e aproximada, com um erro menor que 4%.
- As equações das seções 5 e 8 são baseadas em análises matemáticas aproximadas.
- As equações para as seções 9 e 10 são baseadas na analogia das membranas e podem ser consideradas como aproximadas, com resultados que dificilmente chegam a 10% de erro.

4.2 MÉTODO DOS ELEMENTOS FINITOS

O Método dos Elementos Finitos (MEF), como já dito, é um dos métodos numéricos mais utilizados em *softwares* comerciais da área de engenharia. Seu desenvolvimento vem sendo aprofundado desde os anos 50, quando os engenheiros aeronáuticos Turner, Argyris e Associados iniciaram seus estudos, resultado da evolução da Análise Matricial de Estruturas, juntamente com a computação (SORIANO, 2009).

Segundo Fish e Belytschko (2009), Turner, Clough, Martin e Topp publicaram um dos primeiros artigos relacionados ao MEF, embora não citassem o termo “elementos finitos”. Os autores também lembram que nos anos 60, os elementos finitos despertaram o interesse de matemáticos que verificaram sua eficácia na solução de problemas lineares.

Assim, conforme mais pesquisas foram feitas sobre o assunto, mais utilizado o método tornou-se, possibilitando a solução de problemas mais complexos e de difícil solução através dos métodos convencionais (empíricos ou analíticos). A evolução da computação também teve grande influência nesse desenvolvimento.

Fish e Belytschko (2009) dizem que o MEF é um método de solução aproximada de equações diferenciais parciais. Ainda, na engenharia, o método é utilizado para a solução de problemas relacionados a tensões, mecânica dos fluidos, transferências de calor, entre outros, tendo todos grande aplicação computacional.

Soriano (2009) cita o MEF como um método de simulação numérica, sendo o mais utilizado em Mecânica do Contínuo. Já Giacchini (2012, p. 2) refere-se ao método como sendo "robusto e aplicável em domínios deveras elaborados".

A ideia central do Método dos Elementos Finitos é a discretização de uma estrutura em um número finito de elementos, os quais são interligados por nós, formando uma rede de elementos denominada "malha" (UGURAL, 2009).

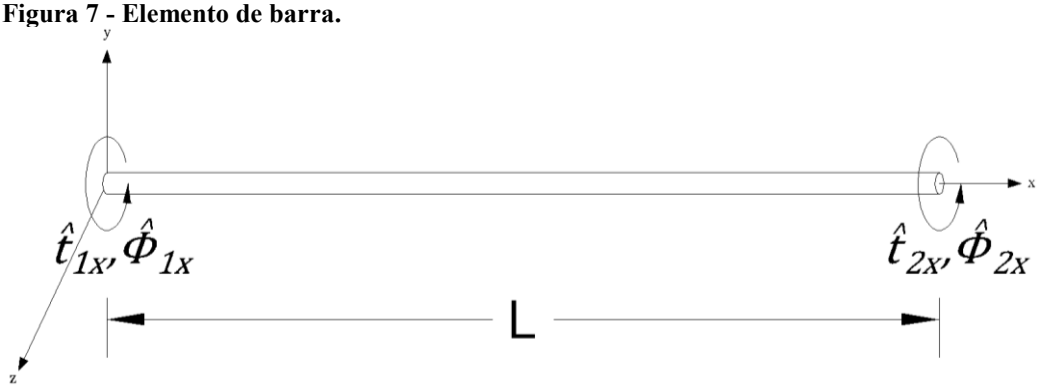
Com o aumento de nós, e consequentemente, de elementos finitos discretizados no elemento real, tem-se um refinamento da malha. Esse refinamento promove soluções mais precisas, embora exija um aumento no processamento de dados computacionais (FISH, BELYTSCHKO, 2009).

4.3 MÉTODO DOS ELEMENTOS FINITOS APLICADO À TORÇÃO DE BARRAS

Visando obter as equações de equilíbrio para o sistema discreto, neste trabalho optou-se pela aplicação do Princípio dos Trabalhos Virtuais (PTV) em um elemento de barra submetido à torção. Tal princípio baseia-se na conservação da energia de deformação, onde o trabalho externo U_e de um elemento deslocado por uma força externa é armazenado como energia de deformação, sendo este um trabalho interno U_e (HIBELLER, 2010, p. 531).

Assim, para a dedução do sistema linear de equações para um elemento finito, considera-se inicialmente um elemento de barra de seção transversal circular, com torques nodais aplicados nas extremidades do mesmo, como visto na Figura 7. Na figura, também são ilustrados os respectivos giros nodais.

Figura 7 - Elemento de barra.



Fonte: Adaptado de Logan (2007).

Logo, considerando a conservação da energia dos trabalhos virtuais interno e externo, tem-se:

$$U_e^* = U_i^* \tag{15}$$

Sendo os trabalhos interno e externo relacionados, respectivamente, às deformações e deslocamentos virtuais e considerando que a torção gera apenas giros em torno do eixo x, como visto na Figura 7, o trabalho externo virtual pode ser representado como sendo:

$$U_e^* = T \cdot \phi^* \tag{16}$$

Ainda, por se tratar de uma análise envolvendo apenas os efeitos de torção, ou seja, apenas tensões cisalhantes, o trabalho interno virtual pode ser escrito como:

$$U_i^* = \int_V \tau \cdot \delta\gamma^* dV \tag{17}$$

Considerando o material trabalhando em regime elástico-linear, é possível substituir a lei de Hooke para o cisalhamento na Equação (17) e, em seguida, a relação diferencial dada pela Equação (3), em termos de deformação real e virtual, $\delta\gamma$ e $\delta\gamma^*$, respectivamente. Assim, tem-se:

$$U_i^* = \int_V G \cdot \frac{d\phi}{dx} \cdot \rho \cdot \frac{d\delta\phi^*}{dx} \cdot \rho dV = G \int_0^L \frac{d\phi}{dx} \cdot \frac{d\delta\phi^*}{dx} \int_A \rho^2 dA dx \tag{18}$$

Na Equação (18), a integral de área representa o momento polar de inércia da seção transversal do elemento, sendo considerado, constante ao longo do seu comprimento. Assim:

$$U_i^* = G \cdot J \cdot \int_0^L \frac{d\phi}{dx} \cdot \frac{d\delta\phi^*}{dx} dx \tag{19}$$

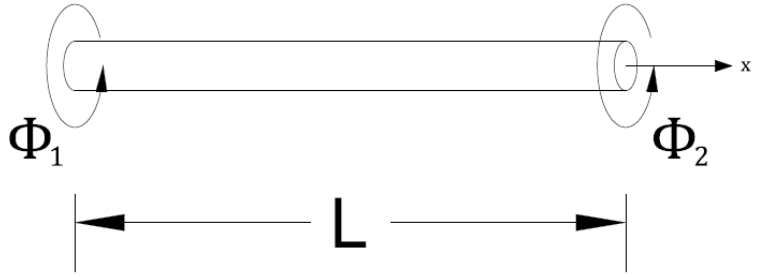
Para obter a função do giro devido à torção ao longo do elemento, considerando o equilíbrio ilustrado na Figura 7 e a linearidade do giro ao longo do elemento, pode-se escrever:

$$\phi = ax + b \tag{20}$$

Ainda, segundo a Figura 8, para encontrar os valores das incógnitas a e b, as seguintes condições de contorno podem ser estabelecidas:

$$\begin{cases} \text{para } x = 0 \rightarrow \phi = \phi_1 \rightarrow b = \phi_1 \\ \text{para } x = L \rightarrow \phi = \phi_2 \rightarrow a = \dfrac{\phi_2 - \phi_1}{L} \end{cases} \tag{21}$$

Figura 8 - Elemento de barra submetido a giros nodais.



Fonte: O autor.

Assim,

$$\phi_{(x)} = \left[\frac{\phi_2 - \phi_1}{L}\right] \cdot x + \phi_1 = \left[1 - \frac{x}{L}\right] \cdot \phi_1 + \left[\frac{x}{L}\right] \cdot \phi_2 \tag{22}$$

Portanto,

$$\varphi_1 = 1 - \frac{x}{L};\ \varphi_2 = \frac{x}{L} \tag{23}$$

A derivada da Equação (22) é dada por:

$$\frac{d\phi}{dx} = \left[-\frac{1}{L}\right]\phi_1 + \left[\frac{1}{L}\right]\phi_2 \tag{24}$$

Substituindo (24) em (19), reorganizando os termos e integrando a equação, obtém-se:

$$U_i^* = \frac{G \cdot J}{L} \cdot \left[\phi_1 \cdot \delta\phi_1^* - \phi_1 \cdot \delta\phi_2^* - \phi_2 \cdot \delta\phi_1^* + \phi_2 \cdot \delta\phi_2^*\right] \tag{25}$$

Que reescrita na forma matricial, é dada por:

$$U_i^* = \begin{bmatrix} \delta\phi_1^* & \delta\phi_2^* \end{bmatrix} \cdot \frac{G \cdot J}{L} \cdot \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{26}$$

O trabalho externo, dado pela Equação (16), considerando a atuação dos torques nodais ilustrados na Figura 7, pode ser escrito na forma matricial:

$$U_e^* = \begin{bmatrix} \delta\phi_1^* & \delta\phi_2^* \end{bmatrix} \cdot \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \tag{27}$$

Com os giros virtuais sendo representados por valores unitários e substituindo (26) e (27) em (15), obtém-se a Equação (28):

$$\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \frac{G \cdot J}{L} \cdot \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{28}$$

que na forma reduzida, pode ser escrita como segue:

$$\left\{\overline{F}\right\}_e = \left\{\overline{K}\right\}_e \cdot \left\{\overline{u}\right\}_e \tag{29}$$

Sendo,

$\left\{\overline{F}\right\}_e$: Vetor de torques nodais;

$\left\{\overline{K}\right\}_e$: Matriz de rigidez do elemento;

$\left\{\overline{u}\right\}_e$: Vetor de giros nodais.

Ainda, na Equação (28), como a mesma foi elaborada considerando uma barra circular, o momento polar de inércia J pode ser calculado como visto no item 4.1.1, sendo o momento polar de inércia de uma seção não circular, o fator K, utilizando o Quadro 1 como referência.

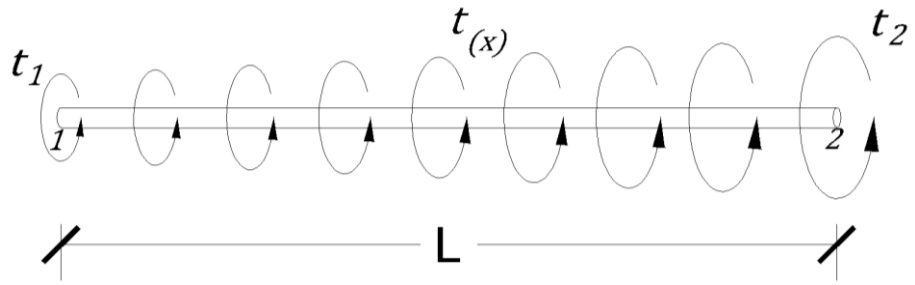
Assim, a matriz de rigidez do elemento não circular é dada por:

$$\left\{\overline{K}\right\}_e = \frac{G \cdot J}{L} \cdot \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{30}$$

Para utilizar o algoritmo para solucionar problemas envolvendo torque distribuído ao longo de um trecho de barra, faz-se necessário o equacionamento do mesmo, de forma a encontrar os torques nodais equivalentes.

Considerando um problema onde um torque distribuído de t_1 e t_2, do nó 1 ao nó 2, respectivamente, é aplicado a um elemento de barra, como visto na Figura 9, é necessário encontrar a equação que descreve tal distribuição.

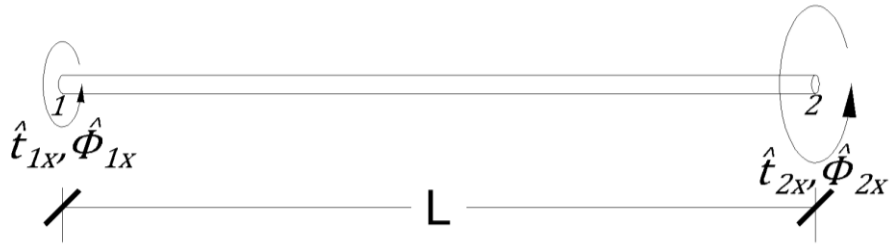
Figura 9 - Elemento de barra submetido a torque distribuído.



Fonte: O autor.

Ainda, pelo fato do Método dos Elementos Finitos relacionar cargas nodais a deslocamentos nodais, o carregamento distribuído pode ser representado por cargas aplicadas aos nós do elemento, como exemplifica a Figura 10.

Figura 10 – Cargas equivalentes para um elemento sob torção.



Fonte: O autor.

Por ser uma distribuição linear, a equação pode ser escrita como:

$$t = ax + b \tag{31}$$

E, segundo a Figura 9, as condições de contorno a seguir podem ser estabelecidas:

$$\begin{cases} \text{para } x = 0 \rightarrow t = t_1 \rightarrow b = t_1 \\ \text{para } x = L \rightarrow t = t_2 \rightarrow a = \dfrac{t_2 - t_1}{L} \end{cases} \tag{32}$$

Substituindo (32) em (31), obtém-se:

$$\phi_{(x)} = \left[\frac{t_2 - t_1}{L}\right] \cdot x + t_1 = \left[1 - \frac{x}{L}\right] \cdot t_1 + \left[\frac{x}{L}\right] \cdot t_2 \tag{33}$$

Para a representação do trabalho externo virtual visto na Equação (16), como o valor da torção é variável ao longo do comprimento do elemento, se faz necessário solucionar a integral representada abaixo. Assim:

$$U_e^* = \int_0^L t_{(x)} \cdot \phi_{(x)}^* dx$$

$$U_e^* = \int_0^L \left(\frac{t_2 - t_1}{L} \cdot x + t_1\right) \cdot \left[\varphi_1 \cdot \phi_1^* + \varphi_2 \cdot \phi_2^*\right] dx \tag{34}$$

Solucionando a Equação (34) e reescrevendo na forma matricial, obtém-se:

$$U_e^* = \begin{bmatrix} (2 \cdot t_1 + t_2) \cdot L/6 \\ (2 \cdot t_2 + t_1) \cdot L/6 \end{bmatrix} \tag{35}$$

Por fim, somando-se tal parcela em (28), junto a parcela de trabalho externo calculada para torques pontuais, temos a equação (36), usada para o cálculo dos giros.

$$\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} + \begin{bmatrix} (2 \cdot t_1 + t_2) \cdot L/6 \\ (2 \cdot t_2 + t_1) \cdot L/6 \end{bmatrix} = \frac{G \cdot J}{L} \cdot \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{36}$$

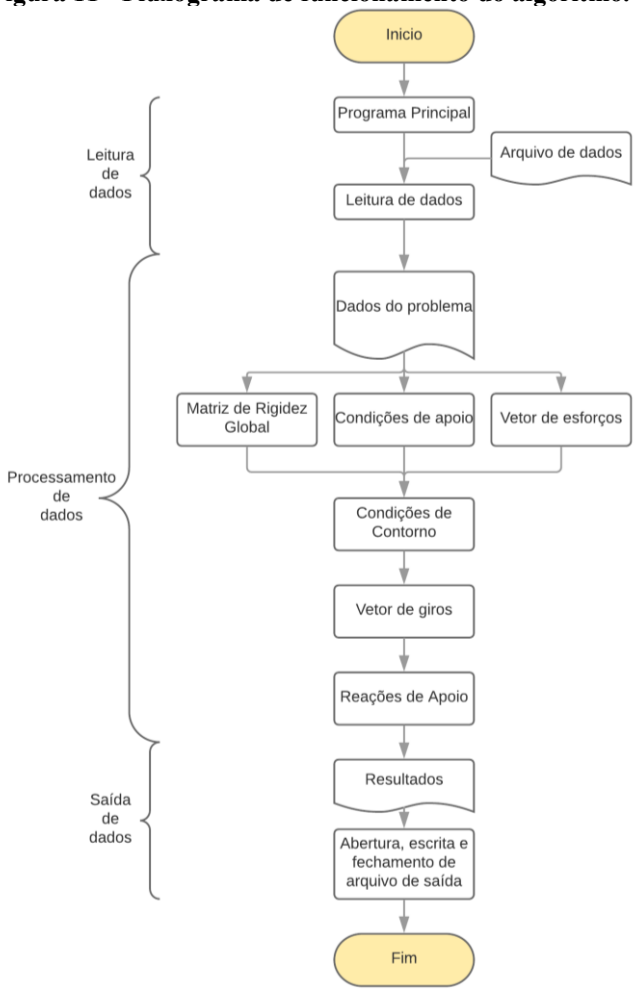
5 ASPECTOS COMPUTACIONAIS

O algoritmo no presente trabalho foi implementado utilizando a linguagem *PYTHON,* onde o mesmo utiliza da formulação do Método dos Elementos Finitos apresentada anteriormente para a solução de problemas relacionados a torção em barras em regime elástico linear, sendo os giros e reações de apoio apresentados como resultado da execução do código.

Vale ressaltar que a utilização da biblioteca *NumPy* ao longo do algoritmo, a qual realiza as multiplicações e inversões de matrizes, permitiu a redução do número de linhas do código e o tornou mais fácil de ser compreendido.

A fim de simplificar a implementação do algoritmo, o mesmo foi organizado em funções, sendo cada função responsável por uma ação dentro do código. A sequência de entrada de dados, processamento e saída de dados é representada na Figura 11.

Figura 11 - Fluxograma de funcionamento do algoritmo.



Fonte: O autor.

5.1 FUNÇÕES

A seguir, a sequência do algoritmo é mostrada. O código pode ser verificado no apêndice do trabalho.

5.1.1 Programa principal

A função *programa_principal* responsável por interagir com o usuário, recebendo a informação do arquivo a ser aberto e lido, utilizando outras funções para o processamento, abertura e fechamento dos arquivos de entrada e saída. Essa função recebe os resultados provenientes das outras funções, gravando-os em variáveis e, posteriormente, em arquivos de saída.

5.1.2 Leitura de dados

A função *leitura_dados* recebe o nome do arquivo a ser lido, realiza a abertura do mesmo e cria as variáveis que recebem as propriedades geométricas da barra, utilizando a função *inercia_J* para o cálculo do momento polar de inércia, os momentos atuantes, a localização e condição de apoio dos nós do problema, retornando duas matrizes com as informações dos nós e propriedades geométricas dos elementos (variável *dados_problema*).

A leitura dos arquivos segue o padrão mostrado no Apêndice B, onde os dados do Exemplo 1 são apresentados. Quanto a seção transversal, o Apêndice C exemplifica o formato da entrada de dados de cada uma das seções do Quadro 1.

5.1.3 Processamento de dados

No processamento de dados, a função *matriz_rigidez* realiza a construção da matriz de rigidez de cada elemento, gravando-as em uma variável do tipo *lista*. Após isso, a função *Mat_rig_g* cria a matriz de rigidez global, utilizando o número de nós e as matrizes locais como parâmetros.

Com isso, a função *calculo_giros* recebe os dados dos nós e a matriz de rigidez global, introduz as condições de contorno na mesma (função *condicoes_de_contorno*), cria, a partir do

torque distribuído nos elementos e dos torques concentrados nos nós, um vetor com os valores dos momentos nodais equivalentes e resolve a Equação 29, encontrando os giros nodais.

Por fim, a Equação 29 é utilizada novamente para encontrar as reações de apoio, usando a matriz de rigidez global sem as condições de contorno e os giros obtidos.

5.1.4 Saída de dados

A saída de dados é realizada pela função *saida_dados*, onde essa recebe as variáveis que contém os giros, as reações e a variável *opcao*, que contém o nome do arquivo de entrada. Assim, a função cria um arquivo de saída com a indicação de qual arquivo os dados se referem.

Com a criação do arquivo, os dados são inseridos no arquivo e o mesmo é fechado, finalizando o algoritmo.

6 RESULTADOS E DISCUSSÕES

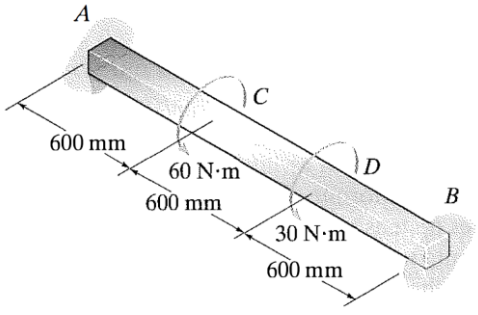
A fim de validar o código, alguns exemplos foram testados no mesmo, e seus resultados comparados com a literatura consultada ou com o programa *Autodesk Robot Structural Analysis Professional 2018 – Student Version*. Este é conhecido no mercado como uma ferramenta de análise estrutural, capaz de realizar, entre outras ações, análises elásticas de estruturas. Os resultados mostram a precisão tanto do método aplicado quanto das aproximações utilizadas nos cálculos do momento polar de inércia do Quadro 1 mostrado anteriormente.

Vale ressaltar que o Exemplo 3 foi desenvolvido de modo a ilustrar uma aplicação direta na área da engenharia, sendo os desenhos do mesmo baseados em projetos cedidos pela SEPLAD (Secretaria Municipal de Planejamento e Administração), da cidade de Curitiba, como resposta ao requerimento de acesso a informação de número 74-000986/2017, protocolado na Prefeitura de Curitiba. Os projetos são de autoria do Arquiteto Manoel Coelho.

6.1 EXEMPLO 1

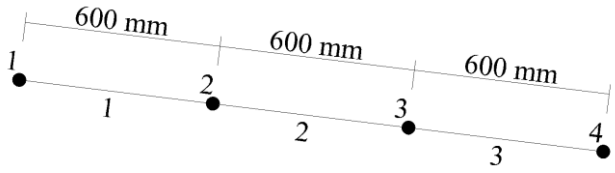
O exemplo 1 mostra uma viga de seção transversal quadrada de 50 mm de lado, bi engastada, de 1,8 m de comprimento. A mesma está sujeita a dois torques aplicados, como mostra a Figura 12. O módulo de Elasticidade Transversal é 27 GPa. Para o processamento do algoritmo, a barra foi discretizada como visto na Figura 13, com quatro nós e três elementos.

Figura 12 - Barra bi engastada de seção quadrada.



Fonte: Hibbeler (2010, p.163).

Figura 13 - Barra discretizada do exemplo 1.



Fonte: O autor.

Os resultados do algoritmo são exibidos na Tabela 1, abaixo, bem como os valores exibidos pelo autor do exemplo. Para o cálculo das reações de apoio, o algoritmo aplicou a Equação 28 com os giros obtidos e a matriz de rigidez global da estrutura, garantindo assim a eficácia do código para barras hiperestáticas.

Tabela 1 - Resultados do Exemplo 1.

	Algoritmo		**Hibbeler (2010)**	
Nó	**Giros (graus)**	**Reações de Apoio (N.m)**	**Giros (graus)**	**Reações de Apoio (N.m)**
1 (A)	0,0	-50	0,0	-50
2 (C)	0,072433182997	-	0,0723	-
3 (D)	0,057946546409	-	Não informado	-
4 (B)	0,0	-40	0,0	-40

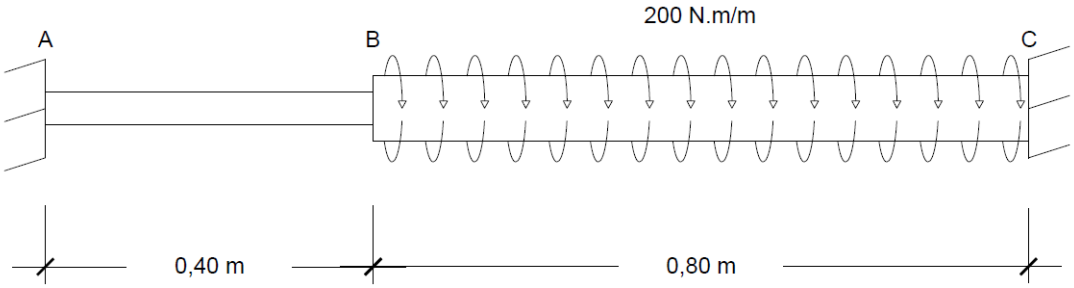
Fonte: O autor.

Tanto os resultados analíticos quanto o da literatura consultada confirmam a correta aplicação do método e programação do algoritmo, sendo que a precisão do resultado depende dos arredondamentos realizados pelo autor do exemplo, como para o cálculo do momento polar de inércia.

6.2 EXEMPLO 2

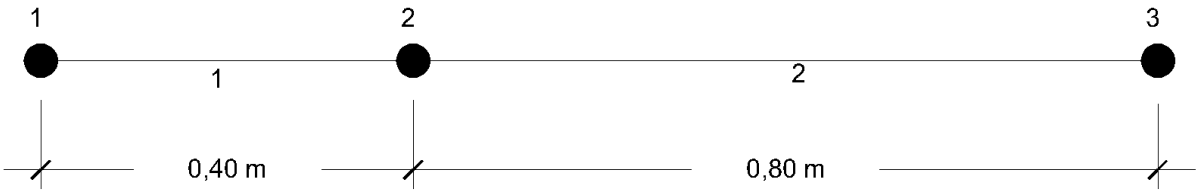
No seguinte exemplo, a Figura 14 ilustra uma barra circular bi engastada de Módulo de Elasticidade Transversal G = 75 GPa, sendo a mesma submetida a um torque constante distribuído de 200 N.m/m ao longo do trecho BC, de 30 mm de diâmetro, como ilustrado abaixo. O trecho AB, sem carregamento, possuí 20 mm de diâmetro. A discretização foi feita como ilustra a Figura 15, com três nós e dois elementos.

Figura 14 - Barra bi engastada submetida a carregamento distribuído.



Fonte: O autor.

Figura 15 - Barra discretizada do exemplo 2.



Fonte: O autor.

Para a validação dos resultados, o problema foi solucionado utilizando o *software Autodesk Robot Structural Analysis Professional 2018 – Student Version*. Os resultados são exibidos na Tabela 2.

Tabela 2 - Resultados do Exemplo 2.

	Algoritmo		***Software***	
Nó	**Giros (rad)**	**Reações de Apoio (N.mm)**	**Giros (rad)**	**Reações de Apoio (N.mm)**
1 (A)	0,0	-22654,87	0,0	-22654,87
2 (B)	0,007692019	-	0,007692019	-
3 (C)	0,0	-137345,13	0,0	-137345,13

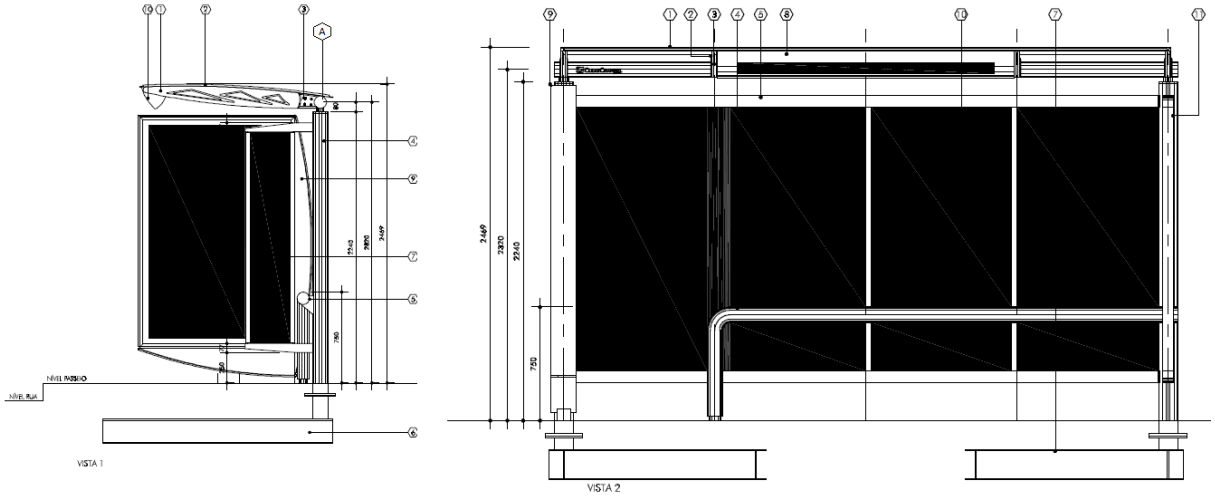
Fonte: O autor.

Os resultados indicam que o método foi implementado corretamente, com resultados precisos, já que os algarismos exibidos pelo algoritmo são iguais aos calculados pelo programa, sendo limitados ao arredondamento da última casa decimal exibida pelo mesmo.

6.3 EXEMPLO 3

A Figura 16 ilustra um abrigo de parada de ônibus da cidade de Curitiba. A partir do mesmo, o exemplo criado utilizando a estrutura e materiais descritos no projeto, com medidas aproximadas e valores de resistência de materiais retirados na bibliografia especializada, a fim de representar uma aplicação do algoritmo em um problema real.

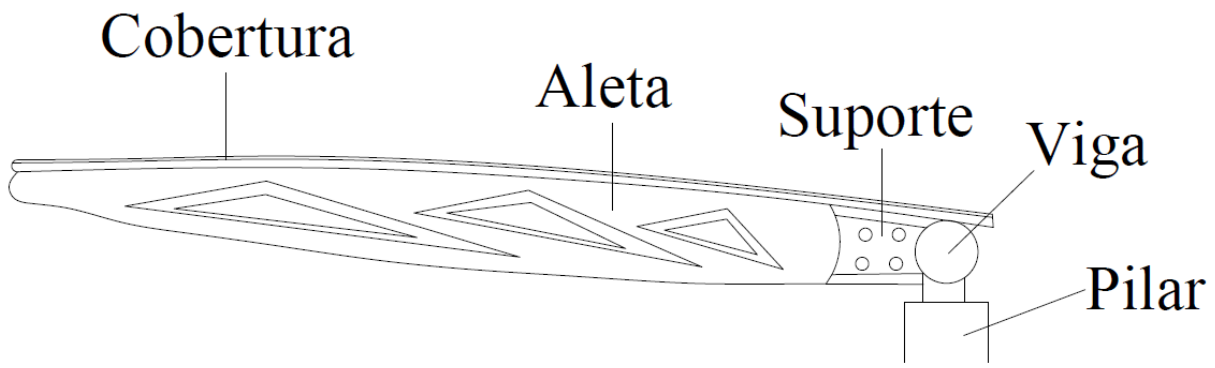
Figura 16 - Projeto de abrigo de parada de ônibus.



Fonte: SEPLAD (2017).

Assim, o exemplo foi elaborado como ilustra a Figura 17, sendo a cobertura em alumínio apoiada nas aletas, também em alumínio. As aletas são fixadas a um suporte em aço, o qual é engastado à viga, um tubo de parede fina de 10 cm de diâmetro externo e 6 mm de espessura de parede, que resiste aos esforços de torção, momento fletor e força cortante. Por fim, a viga é fixa aos pilares nas laterais da estrutura, transmitindo assim os esforços à fundação e ao solo. Ainda, foi considerado o Módulo de Elasticidade Transversal do Aço como G = 83 GPa.

Figura 17 - Projeto de abrigo de parada de ônibus (Lateral superior).



Fonte: Adaptado de SEPLAD (2017).

A Fotografia 1 ilustra a estrutura em questão, que difere do projeto obtido apenas na estrutura de propaganda, a qual não influencia nos esforços resistidos pela viga analisada. Observa-se que a estrutura de aletas e suporte visto na Figura 17 se repete ao longo do abrigo, num total de cinco aletas e suportes. A cobertura é apoiada ao longo das aletas.

Fotografia 1 - Abrigo de parada de ônibus da cidade de Curitiba.

Fonte: O autor.

Fotografia 2 - Abrigo de parada de ônibus da cidade de Curitiba (Vista lateral).

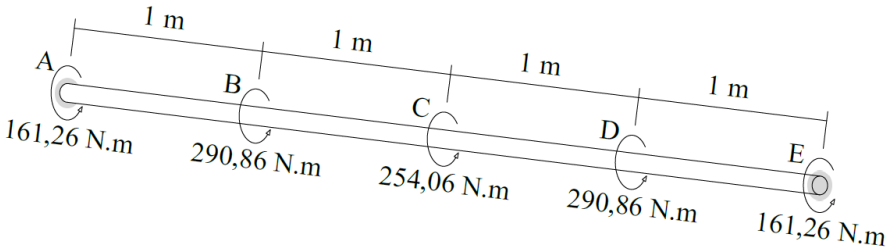
Fonte: O autor.

Os esforços produzidos pela cobertura, aletas e suportes foram estimados e calculados para encontrar os valores de torção aproximados que agem sobre a viga, como visto na Figura 18. Tais cálculos foram estimados a partir de aproximações com relação ao peso próprio da

estrutura, utilizando os pesos específicos dos materiais para estimar o peso e os momentos torsores produzidos. Com esses valores calculados, o algoritmo foi utilizado para o cálculo dos ângulos de torção da estrutura. Os resultados obtidos podem ser visualizados na Tabela 3.

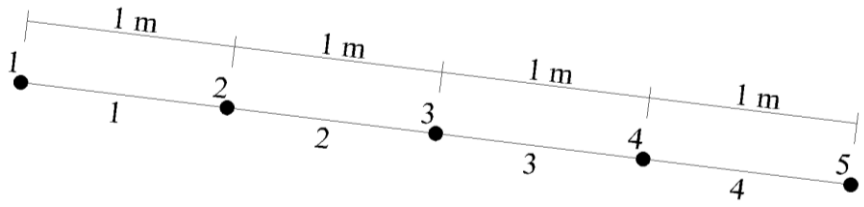
Cabe ressaltar que os giros nas extremidades são relativos ao pilar em que a viga é engastada, tanto em A quanto em E. Assim, espera-se que o giro calculado nos engastes sejam nulos. Para a discretização, a malha foi dividida em cinco nós, quatro elementos, como ilustrado na Figura 19. O problema também foi solucionado utilizando o *software Autodesk Robot Structural Analysis Professional 2018 – Student Version.*

Figura 18 - Viga bi engastada com cinco momentos aplicados.



Fonte: O autor.

Figura 19 – Barra discretizada do exemplo 3.



Fonte: O autor.

Tabela 3 - Resultados do Exemplo 3.

	Algoritmo		*Software*	
Nó	**Giros (rad)**	**Reações de Apoio (N.mm)**	**Giros (rad)**	**Reações de Apoio (N.mm)**
1 (A)	0,0	-579150	0,0	-579150
2 (B)	0,001281130	-	0,001281130	-
3 (C)	0,001670568	-	0,001670568	-
4 (D)	0,001281130	-	0,001281130	-
5 (E)	0,0	-579150	0,0	-579150

Fonte: O autor.

Assim como nos exemplos anteriores, os resultados apontam que o código foi implementado corretamente, com precisão que depende do arredondamento da última casa decimal do programa utilizado para a validação dos dados obtidos.

7 CONCLUSÃO

O trabalho apresentado teve como objetivo formular um código computacional baseado no Método dos Elementos Finitos capaz de analisar barras sob torção em regime elástico linear.

Para tal, o referencial teórico foi escrito de modo a compreender a metodologia em questão, seus princípios e formulações. Assim, aliado aos conceitos relacionados às tensões e deformações de barras submetidas a torção, as equações necessárias para a solução de problemas relacionados ao tema foram descritas.

Posteriormente, foi implementado um código na linguagem *PYTHON* capaz de ler dados de entrada, interpretá-los, realizar os cálculos necessários para sua utilização, aplicar o método descrito no referencial teórico e realizar a saída dos resultados ao usuário, sendo estes os giros nodais e as reações de apoio. O algoritmo foi dividido em funções, facilitando o entendimento e possibilitando a implementação de outras funcionalidades que possam ser úteis.

Por fim, foram apresentados exemplos do tema, os quais foram solucionados pelo algoritmo, tendo seus resultados comparados com os obtidos pela referência consultada ou pelo programa da área, validando com êxito o método aplicado e o código desenvolvido.

Assim, o código pode ser usado como instrumento de estudo de problemas relacionados a barras sob torção, seja no estudo do comportamento de barras com diferentes características ou como auxílio para alunos que estudam os métodos de cálculo.

REFERÊNCIAS BIBLIOGRÁFICAS

BUFFONI, Salete S. O. Apostila de introdução aos métodos numéricos – parte I. **Universidade Federal Fluminense**, 2002. Disponível em: <http://www.professores.uff.br/salete/imn/calnumI.pdf>. Acesso em: 12 maio 2017.

BASTOS, Paulo S. S. **Torção em vigas de concreto armado – Notas de aula.** Universidade Estadual Paulista. Bauru, 2014. Disponível em: < http://wwwp.feb.unesp.br/pbastos/concreto2/Torcao.pdf>. Acesso em: 13 maio 2017.

FIGUEIREDO, Tathiana C. S. P. **Estudo experimental do reforço à torção de vigas de concreto armado com compósitos de fibras de carbono**. 2014. 165 f. Dissertação (Mestrado em Engenharia Civil) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Civil, 2014. Disponível em: < http://www2.dbd.puc-rio.br/pergamum/tesesabertas/1112054_2014_cap_2.pdf>. Acesso em: 13 maio 2017.

FISH, Jacob; BELYTSCHKO, Ted. **Um primeiro curso em elementos finitos.** Rio de Janeiro, RJ: LTC, 2009. xi, 241 p.

GIACCHINI, Breno L. Uma breve introdução ao Método dos Elementos Finitos. **Departamento de Matemática: Instituto de Ciências Exatas, Universidade Federal de Minas Gerais**, 2012. Disponível em: < http://www.mat.ufmg.br/~rodney/notas_de_aula/elementos_finitos.pdf>. Acesso em: 13 maio 2017.

HIBBELER, Russell C. **Resistência dos materiais.** 7. ed. São Paulo, SP: Pearson, 2010. 637p.

LOGAN, Daryl L. **A First Course in the Finite Element Method**. 4. ed. Plateville: Thomson, 2007.

SEPLAD, Secretaria Municipal de Planejamento e Administração. **Resposta à solicitação nº 74-000986/2017 protocolada na Prefeitura de Curitiba.** [mensagem pessoal]. Mensagem recebida por: <alissoncezar.c@gmail.com> em 04 dez. 2017.

SORIANO, Humberto L. **Elementos finitos: formulação e aplicação na estática e dinâmica das estruturas**. Ciência Moderna, 2009.

SÜSSEKIND, José C. **Curso de análise estrutural**. 10. ed. São Paulo: Editora Globo S.A., 1989. 3 v.

UGURAL, Ansel C. **Mecânica dos materiais.** 1. ed. Rio de Janeiro: LTC, 2009.

YOUNG, Warren C.; BUDYNAS, Richard G. **Roark's formulas for stress and strain**. 7. ed. New York: McGraw-Hill, 2002.

APÊNDICE A – CÓDIGO COMPUTACIONAL IMPLEMENTADO

```
import numpy as np

def prm (m,titulo):                                    #Função que
imprime uma matriz 'm' de nome 'titulo'.
    print ("\n",titulo)
    for i in range (len(m)):
        for j in range (len(m[i])):
            print (m[i][j],end='|')
        print('')
    print('---------------------------------')

def inercia_J(v):                                  #Função que calcula
Momento Polar de Inércia segundo o Quadro 1.
    pi = 3.141592653589793
    if v[3] == 1:
        J = v[4]**4*pi/2
    elif v[3] == 2:
        J = pi*v[4]**3*v[5]**3/(v[4]**2+v[5]**2)
    elif v[3] == 3:
        J = 2.25*v[4]**4
    elif v[3] == 4:
        J = v[4]*v[5]**3*(16/3-3.36*v[5]/v[4]*(1-v[5]**4/(12*v[4]**4)))
    elif v[3] == 5:
        J = 2*v[6]*v[7]*(v[4]-v[7])**2*(v[5]-
v[6])**2/(v[4]*v[7]+v[5]*v[6]-v[6]**2-v[7]**2)
    elif v[3] == 6:
        J = v[4]**4*3**0.5/80
    elif v[3] == 7:
        J = pi*(v[4]**4-v[5]**4)/2
    elif v[3] == 8:
        A = pi*(v[4]**2-(v[4]-v[5])**2)
        U = 2*pi*(v[4]-v[5]/2)
        J = 4*A**2*v[5]/U
    elif v[3] == 9:
        K1 = v[4]*v[5]**3*(1/3-0.21*v[4]/v[5]*(1-v[5]**4/(12*v[4]**4)))
        K2 = v[6]*v[7]**3*(1/3-0.105*v[7]/v[6]*(1-
v[7]**4/(192*v[6]**4)))
        if v[5]<v[7]:
            t = v[5]
            t1 = v[7]
        else:
            t = v[7]
            t1 = v[5]
        alfa = t/t1*(0.15+0.1*v[8]/v[5])
        D = ((v[5]+v[8])**2+v[8]*v[7]+v[7]**2/4)/(2*v[8]+v[5])
        K = 2*K1+K2+2*alfa*D**4
    elif v[3] == 10:
        K1 = v[4]*v[5]**3*(1/3-0.21*v[4]/v[5]*(1-v[5]**4/(12*v[4]**4)))
        K2 = 1/3*v[6]*v[7]**3
        if v[5]<v[7]:
            t = v[5]
            t1 = v[7]
        else:
            t = v[7]
            t1 = v[5]
        alfa = t/t1*(0.15+0.1*v[8]/v[5])
        D = ((v[5]+v[8])**2+v[8]*v[7]+v[7]**2/4)/(2*v[8]+v[5])
```

```
        K = 2*K1+K2+2*alfa*D**4

    return J

def vet_string_to_float(vetor):                      #Função que
transforma os dados de um vetor para float.
    for i in range (len(vetor)):
        vetor[i] = float(vetor[i])
    return vetor

def criar_matriz_lista(a,b):                         #Função que cria
uma matriz com 'a' linhas e 'b' colunas, preenchida com zeros.
    matriz = []
    for i in range (a):
        vet = []
        for j in range (b):
            vet.append(0.0)
        matriz.append(vet)
    return matriz

def leitura_dados(dados):                             #Função que
realiza a leitura do arquivo de dados e retorna os dados dos nós e
elementos em duas matrizes.
    arquivo = open(dados+".txt","r")

    #Leitura das posições dos nós
    linha = arquivo.readline()
    if linha == "NÓS\n":
        linha = arquivo.readline()
        vet2 = []                                            #Criando
vetor auxiliar 2.
        while linha != '\n':                                 #Repetição
'while' para ler os dados do primeiro título (Quando a linha chegar ao
fim dos dados, a variavel linha será igual a uma linha vazia com quebra
de parágrafo).
            vet1 =  linha.split()                            #Vetor
auxiliar 1 recebe os dados em string.
            vet2.append(float(vet1[0]))                      #Conversão
de dado 'string' em 'float'.
            linha = arquivo.readline()                       #Leitura da
próxima linha de dados.
        Matriz_nós = criar_matriz_lista((len(vet2)),4)       #Criação da
Matriz de dados dos nós.
        for i in range (len(Matriz_nós)):
#Atualização dos dados da matriz.
            Matriz_nós[i][0] = i+1                           #Inserção
do número do nó.
            Matriz_nós[i][1] = float(vet2[i])                #inserção
da posição do nó no eixo X.

    #Leitura dos momentos
    linha = arquivo.readline()                           #Leitura da linha
com o título dos dados.
    if linha == "MOMENTOS\n":                            #Verificando se é o
título correto.
        linha = arquivo.readline()                       #Leitura da linha
de dados.
```

```
        vet2 = []                                   #Criando vetor
auxiliar 2.
        while linha != '\n':                        #Repetição 'while'
para ler os dados do primeiro título (Quando a linha chegar ao fim dos
dados, a variavel linha será igual a uma linha vazia com quebra de
parágrafo).
            vet1 =  linha.split()                   #Vetor auxiliar 1
recebe os dados em string.
            vet2.append(float(vet1[0]))             #Conversão de dado
'string' em 'float'.
            linha = arquivo.readline()              #Leitura da próxima
linha de dados.
        for i in range (len(Matriz_nós)):           #Atualização dos
dados da matriz.
            Matriz_nós[i][2] = vet2[i]              #Inserção do
momento no nó.

    #Leitura das vinculações
    linha = arquivo.readline()                      #Leitura da linha
com o título dos dados.
    if linha == "APOIOS\n":                         #Verificando se é o
título correto .
        linha = arquivo.readline()                  #Leitura da linha
de dados.
        vet2 = []                                   #Criando vetor
auxiliar 2.
        while linha != '\n':                        #Repetição 'while'
para ler os dados do primeiro título (Quando a linha chegar ao fim dos
dados, a variavel linha será igual a uma linha vazia com quebra de
parágrafo).
            vet1 =  linha.split()                   #Vetor auxiliar 1
recebe os dados em string.
            vet2.append(vet1[0])                    #Adicionando dado
ao vetor auxiliar 2.
            linha = arquivo.readline()              #Leitura da próxima
linha de dados.
        for i in range (len(Matriz_nós)):           #Atualização dos
dados da matriz.
            Matriz_nós[i][3] = vet2[i]              #Inserção do apoio
no nó.

    #Leitura das propriedades dos elementos.
    linha = arquivo.readline()
#Leitura da linha com o título dos dados.
    if linha == "PROP_ELEM\n":
#Verificando se é o título correto.
        linha = arquivo.readline()
#Leitura da linha de dados.
        Matriz_prop_elem = criar_matriz_lista((len(Matriz_nós))-1,4)
#Criação da matriz de propriedades dos elementos.
        for i in range (len(Matriz_nós)-1):
#Repetição para preencher a matriz de propriedades dos elementos.
            vet1 = linha.split()
#Vetor auxiliar 1 recebe os dados em string.
            vet2 = [i+1]
#Vetor auxiliar 2 recebe o número do elemento.
```

```
            vet2.append(Matriz_nós[i+1][1]-Matriz_nós[i][1])
#Cálculo do comprimento do elemento e adição ao vetor auxiliar 2.
            vet2.append(float(vet1[0]))
#Adição do dado 'Módulo de Elasticidade Transversal' do elemento ao
vetor auxiliar 2.
            vet2.append(inercia_J(vet_string_to_float(vet1)))
#Cálculo do momento polar de inércia do elemento e adição ao vetor
auxiliar 2.
            vet2.append(vet1[1])
            vet2.append(vet1[2])
            Matriz_prop_elem[i] = vet2
#Inserção dos dados lidos à matriz de propriedades dos elementos.
            linha = arquivo.readline()
#Leitura da próxima linha de dados.
    arquivo.close()
#Fechamento do arquivo.

    return Matriz_nós,Matriz_prop_elem
#Retorno das matrizes de nós e propriedades dos elementos pela função
de leitura de dados.

def matriz_rigidez(v):                                  #Função que
cria a matriz de rigidez de um elemento.
    K_local = criar_matriz_lista(2,2)
    for i in range (len(K_local)):
        for j in range (len(K_local)):
            if i == j:                                  #Valores na
diagonal principal positivos.
                K_local[i][j] = v[2]*v[3]/v[1]
            else:                                       #Outros valores
negativos.
                K_local[i][j] = -v[2]*v[3]/v[1]
    return K_local

def Mat_rig_g(n,k):                                 #Função que cria
matriz de rigidez global com 'n' nós, utilizando as matrizes locais
contidas em 'k'.
    K_glob = criar_matriz_lista(n,n)
    for i in  k:
        K_glob[i-1][i-1] += float(k[i][0][0])
        K_glob[i-1][i] += float(k[i][0][1])
        K_glob[i][i-1] += float(k[i][1][0])
        K_glob[i][i] += float(k[i][1][1])
    return K_glob

def Mat_momentos(nos,elementos):                        #Função que
cria uma matriz composta pelos valores de torque aplicados nas barras e
nós.
    mom = criar_matriz_lista(len(nos),1)                #Criação da
matriz.
    for i in nos:                                       #Repetição para
ler os torques aplicados em nós.
        mom[i[0]-1][0] = i[2]                           #Adicionando o
torque a respectiva posição de nó.
    for j in elementos:                                 #Repetição para
ler os torques distribuídos as barras.
```

```
        mom[j[0]-1][0] += (2*j[4] + j[5]) * j[1] / 6    #Adição do
torque proporcional ao trecho na respectiva posição do nó.
        mom[j[0]][0] += (j[4] + 2*j[5]) * j[1] / 6      #Adição do
torque proporcional ao trecho na respectiva posição do nó.
    return mom                                          #Retorna a
matriz.

def condicoes_de_contorno(nos,k,m):                     #Função que
recebe os dados dos apoios e a matriz global e aplica as condições de
contorno.
    l = len(k)
    for i in nos:
        if i[3] == "E":
            n = i[0] - 1
            m[n] = [0]
            for x in range (l):
                k[n][x] = 0
                k[x][n] = 0
            k[n][n] = 1
    return k, m

def calculo_giros (nos,K_gl,elementos):
#Função que realiza o cálculo dos giros.
    Momentos = Mat_momentos(nos,elementos)
#Criação da matriz com os valores de torção.
    mat_cond_cont, mom_cond_cont =
condicoes_de_contorno(nos,K_gl,Momentos)     #Aplicação das condições
de contorno.
    Giros = np.linalg.solve(mat_cond_cont,mom_cond_cont)
#Solução da equação, utilizando função da biblioteca NumPy.
    return Giros
#Retorna os valores dos giros.

def reacoes_de_apoio(nos,mat_global,giros,Mom_problema):    #Função que
retorna os nós fixos com seus respectivos valores de reação de apoio  .
    momentos = np.dot(mat_global,giros)                     #Cálculo
das reações de apoio, multiplicando os valores da matriz global com os
giros.
    reacoes = momentos - Mom_problema                       #Subtração
dos valores iniciais.
    ap_fixo = []                                            #Declaração
do vetor de resultados.
    for i in range (len(nos)):                              #Repetição
para ler os valores dos nós.
        if nos[i][3] == "E":
#Verificação do apoio do nó.
            ap_fixo.append([nos[i][0],reacoes[i][0]])       #Adição do
valor caso o nó seja do tipo "engaste".
    return ap_fixo                                          #Retorna os
valores de reação de apoio.

def copia_matriz (h):                                       #Função
para copiar os valores de uma matriz a outra variável.
    s = criar_matriz_lista(len(h),len(h[0]))
    for i in range(len(h)):
        for j in range(len(h[0])):
            s[i][j] = h[i][j]
```

```
    return s

def saida_dados(g,r,o):                                          #Função que
cria um arquivo e escreve os valores de saída do algoritmo.
    arq_saida = open(o+"_saida.txt",'w')
    string = "GIROS\n"
    for i in range (len(g)):
        string = string + "Nó " + str(i+1) + ": " + str(g[i][0]  ) + "
rad\n"
    arq_saida.write(string)
    string = "\nREAÇÕES DE APOIO\n"
    for i in range (len(r)):
        string = string + "Apoio " + str(r[i][0]) + ": " + str(r[i][1])
+ " N.mm\n"
    if string == "\nREAÇÕES DE APOIO\n":
        arq_saida.write("\nREAÇÕES DE APOIO\nNão há.")
    else:
        arq_saida.write(string)
    arq_saida.close()
    print ("\nSaída de dados Concluída")

def programa_principal():           #Função que executa as funções de
leitura de dados, cálculos e saída dos resultados.
    aux1 = 0
    while aux1 == 0:
        try:
            opcao = input("Nome do arquivo a ser importado('s' para
sair): ")
            if opcao == "s":
                return ("FIM")
            dados_problema = leitura_dados(opcao)
#Variável 'dados_problema' recebe as matrizes com os dados dos nós e
elementos.
            aux1 = 1
        except:
            print ("Arquivo inválido")
    Dic_K_local = {}
#Dicionário que armazena as matrizes de rigidez dos elementos.
    aux2 = 1
    for i in dados_problema[1]:                      #Repetição para
cálculo das matrizes de rigidez de cada elemento
        Dic_K_local[aux2] = matriz_rigidez(i)       #Cálculo da matriz
de rigidez de um elemento e adição do mesmo ao dicionário
        aux2 += 1
    K_Global = Mat_rig_g(len(dados_problema[0]),Dic_K_local)
    k_bkp = copia_matriz(K_Global)
    giros = calculo_giros(dados_problema[0], K_Global,
dados_problema[1])
    Momentos = Mat_momentos(dados_problema[0],dados_problema[1])
    reacoes = reacoes_de_apoio(dados_problema[0],np.array(k_bkp),
np.array(giros),Momentos)
    saida_dados(giros,reacoes,opcao)

#As linhas abaixo são responsáveis por re-executar o algoritmo,
possibilitando a realização dos cálculos de outros arquivos de entrada.
opc = "1"
while opc == "1":
```

```
programa_principal()
opc = input("1 - Continuar\nENTER para finalizar\n")
```

APÊNDICE B – ARQUIVO DE ENTRADA DO EXEMPLO 1

```
NÓS
0
600
1200
1800

MOMENTOS
0
60000
30000
0

APOIOS
E
L
L
E

PROP_ELEM
27000 0     0     3     25
27000 0     0     3     25
27000 0     0     3     25
```

APÊNDICE C – FORMATO DE ENTRADA DE DADOS DE PROPRIEDADES DOS ELEMENTOS (DE ACORDO COM A SEÇÃO TRANSVERSAL)

```
PROP_ELEM

G     t1    t2    1     r
G     t1    t2    2     a     b
G     t1    t2    3     a
G     t1    t2    4     a     b
G     t1    t2    5     a     b     ta    tb
G     t1    t2    6     a
G     t1    t2    7     co    ci
G     t1    t2    8     r     t
G     t1    t2    9     a     b     c     d     r
G     t1    t2    10    a     b     c     d     r
```

G = Módulo de Elasticidade Transversal;
t1 = Torque Distribuído no nó 1 do elemento;
t2 = Torque Distribuído no nó 2 do elemento;
1, 2, 3, 4, 5, 6, 7, 8, 9, 10 = Identificação da seção transversal, segundo o Quadro 1;
r, a, b, ta, tb, co, ci, t, c, d, r = Dimensões da seção transversal.