ES020 - Trabalho 02

Parte 0 - Extração de Dados

A primeira coisa que precisa ser feita é determinar as constantes e as variáveis usadas para a análise de Resistência dos Materiais que será feita. Como queremos analisar os esforços internos e a deformação sofrida por uma viga, iremos precisar de:

- L- Comprimento da viga (m)
- B - Largura da viga (m)
- a - Altura da viga (m)
- w_0- Intensidade do carregamento (N/m)
- E - Módulo de Elasticidade do Material (Pa)

Para tal, iremos ler os dados pre-determinados no arquivo `.m`:

```
load input.mat % Load saved data
RA = '215663'  % Student's RA
```

```
RA =
'215663'
```

```
[L, B, a, w0, E] = dados_de_entrada(RA, T)
```

```
L = 8.8000
B = 0.2160
a = 0.4550
w0 = 98800
E = 2.0040e+11
```

Parte 1 - Análise Geométrica

A partir dos dados coletados, somos capazes de prosseguir e calcular os parâmetros geométricos necessários para as análises.

1.1 Área

Como o perfil da viga em análise é composta de uma fórmula matemática, somos capazes de calcular a área fazendo sua integral, entre os limites de integração corretos, como mostrado abaixo:

```
y = @(z) a*cos(pi*z*1/B); % Function representing the side profile

upper_limit = B/2;        % Upper integration limit
lower_limit = -B/2;       % Lower integration limit

A = integral(y, lower_limit, upper_limit)
```

```
A = 0.0626
```

1.2 Momento de Área

Outra informação que é de extrema importância é o momento de área Q_z, que somos capazes de calcular de duas formas distintas:

- Integral Simples
- Integral Dupla

```
%% Inegral Simples %%
q_Z_simple = @(z) (cos(pi * z * 1/B)).^2;

upper_limit_s = B/2;        % Upper integration limit
lower_limit_s = -B/2;       % Lower integration limit

q_z1 = (a^2 /2)* integral(q_Z_simple, lower_limit_s, upper_limit_s);

%% Ingegral Dupla %%
q_z_double = @(z, y) y;                         % This is the required order
to have a function as upper limit

upper_limit_second = @(z) a.*cos(pi.*z.*1/B); % There can only be a function
as a limit for the second integration
lower_limit_second = 0;                         % Lower limit of the
integration is zero

upper_limit_first = B/2;                        % Upper limit of the first
integral
lower_limit_first = -B/2;                       % Lower limit of the first
integral

q_z2 = integral2(q_z_double, lower_limit_first, upper_limit_first,
lower_limit_second, upper_limit_second);
present_data({q_z1, q_z2}, ["Qz - Integral Simples", "Qz - integral Dupla"])
```

ans = 1×2 table

	Qz - Integral Simples	Qz - integral Dupla
1	0.0112	0.0112

Onde podemos observar que ambas as formas resultam no valor correto. Há, entretanto, uma maior facilidade de se implementar a integral simples, por não precisar se preocupar com qual a ordem das variáveis de integração (tendo em vista que podemos usar uma function handle como um limite de integração somente na segunda integração).

1.3 Centróide

Já para o centróide, temos:

```
y_c = q_z1 / A;
y_c_shortcut = pi * a * 1/8;
present_data({y_c y_c_shortcut}, ["Yc - Calc" "Yc - Form"])
```

ans = 1×2 table

	Yc - Calc	Yc - Form
1	0.1787	0.1787

Onde podemos observar que ambas as formas também geram o resultado correto.

1.3 Centróide

Analogamente ao momento de área, o segundo momento I_{zz} também possui duas formas de ser calculada (além da formula simplificadora), sendo elas:

```
%% Primeira Forma %%
f_Izz = @(z, y) y.^2;

upper_limit_1 = B/2;
lower_limit_1 = -B/2;

upper_limit_2 = @(z) a*cos(pi*z*1/B)-y_c;
lower_limit_2 = -y_c;

Izz_1 = integral2(f_Izz, lower_limit_1, upper_limit_1, lower_limit_2,
upper_limit_2);

%%Segunda Forma %%
f_Izz_linha = @(z) (cos(pi * z * 1/B)).^3;

upper_limit_linha = B/2;
lower_limit_linha = -B/2;

I_zz_linha = ((a^3)/3) * integral(f_Izz_linha, lower_limit_linha,
upper_limit_linha);
Izz_2 = I_zz_linha - A*y_c^2;

%% Formula %%
Izz_3 =     ((128 - 9*pi^2)/(288 * pi)) * (B * a^3);

present_data({Izz_1, Izz_2, Izz_3}, ["Izz - Integral Dupla", "Izz - Integral
Simples", "Izz - Formula"])
```

```
ans = 1×3 table
```

	Izz - Integral Dupla	Izz - Integral Simples	Izz - Formula
1	8.8092e-04	8.8092e-04	8.8092e-04

1.4 Discusão

Podemos observar que a integração numérica foi feita através da função `integral` e `integral2`, para integrais simples e duplas, respectivamente. Tal método foi utilizado pela sua simplicidade, pois podemos passar uma function handle ao invés de gerar uma matriz e aplicar a fórmula sobre ela, o que é muito mais

eficaz no que tange memória, tendo em vista que os vetores não nos interessa, mas sim somente o resultado final.

Como dito anteriormente, são utilizadas funções diferentes para integrações simples e integrações duplas. Sendo sua principal diferença o fato de que para a integral dupla, é necessário passar um function handle que receba duas variáveis (mesmo que sua função só dependa de uma) e a ordem das variáveis de integração (e por conseguinte dos limites de integração) possui uma restrição, sendo essa: Não é permitido que os limites de integração da primeira variável sejam dependentes (e portanto sejam representadas por function handle).

No que tange erros de cálculo, eles não foram expressivos, tendo os resultados obtidos de todas as formas distintas o mesmo valor.

Parte 2 - Deflexão da viga

2.1 Introdução

Para que possamos estudar a deflexão da viga, é necessário criarmos um modelo matemático que descreva tal fenômeno. Através da análise dos esforços internos, carregamentos externos e propriedades do material temos que esse modelo é uma equação diferencial de segunda ordem dada por:

$$EI_{zz}\left(\frac{d^2}{dx^2}v_y(x)\right) = M_z(x)$$

Um detalhe muito importante de ser visto, entretanto, é que estamos lidando com uma equação que necessita de valores de contorno (e não valores iniciais). Para os problemas de viga bi-apoiadas, temos que os únicos valores de contorno que temos são os valores da deflexão em ambas as extremidades (que são zero).

Levando isso em consideração, e também o fato de que a maioria dos integradores disponíveis são de primeiro grau, a primeira coisa que iremos transformar o problema de uma equação diferencial de 2 ordem em um sistema de equações de 1 ordem.

2.2 Sistema de EDOs

Analogamente ao que é feito na análise de vibrações mecânicas, iremos separar a EDO de 2 ordem em 2 EDOs de 1 ordem, através da determinação de 2 "Estados", sendo eles:

$$q_1 = v_y$$
$$q_2 = \dot{q}_1 = \frac{dv_y}{dx} = \theta$$

Dos quais podemos escrever o seguinte sistema de equações diferenciais:

$$\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}q_1\\ q_2\end{bmatrix} + \begin{bmatrix}0\\ 1/EI_{zz}\end{bmatrix}\begin{bmatrix}M_z\end{bmatrix}$$

Aux Functions

```
function [L, B, a, w0, E] = dados_de_entrada(RA, T)
```

```
% DADOS_DE_ENTRADA returns all the necessary information to solve the
% problem by passing your RA (as string) and saved data (as table)
%
%   [L, B, a, w0, E] = dados_de_entrada('215663', T) returns the data for
%   the student with RA 215663

target_data = T(strcmp(T.RA, RA), :);

L  = target_data.L;
B  = target_data.B;
a  = target_data.a;
w0 = target_data.w0;
E  = target_data.E;

end

function Table = present_data(data, variable_names)
Table = table(data{:}, 'VariableNames',variable_names);
end
```