Projeto 01 - ES020

1. Tensões e Deflexões

1.1 Momento Fletor - Função de Singularidade

Algo fundamental para o estudo de Resistência de Materiais é a função de singularidade, que nada mais é do que representar, de forma compacta, uma função por partes, como mostrado abaixo:

$$\langle x-y\rangle^n = \begin{cases} 0 & x < y \\ (x-y)^n & x \geq y \end{cases}$$

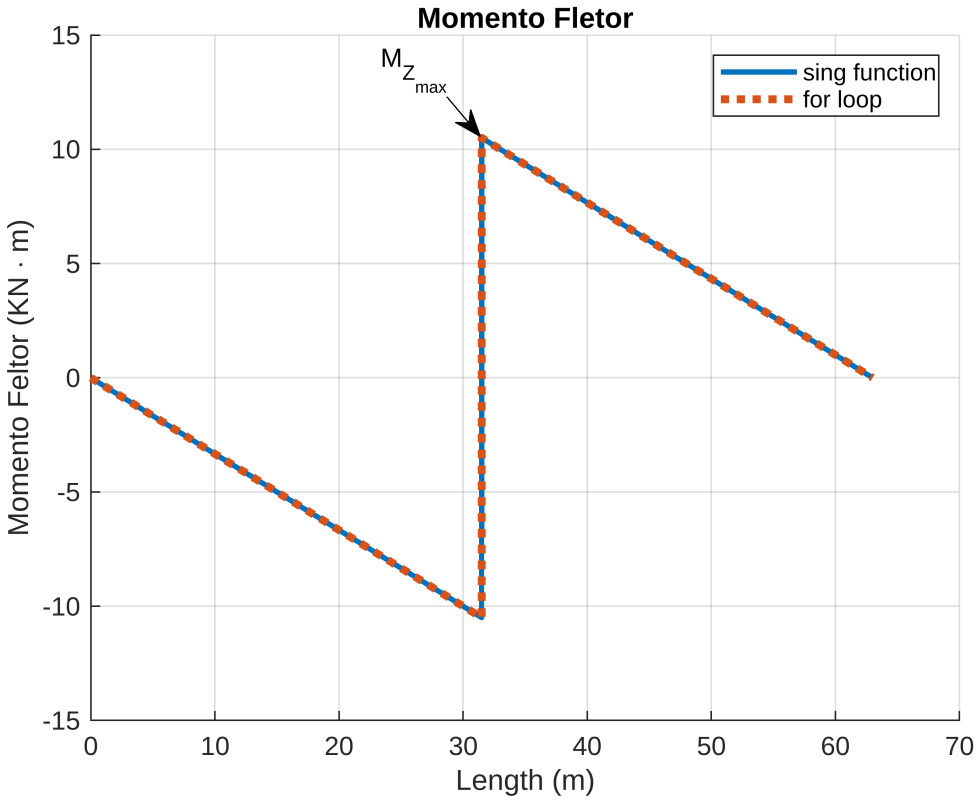
Abaixo demonstra-se duas formas diferentes de se programar tal comportamento, utilizando o momento feltor, para $P = M_0/L$:

```
base_numbers = number_gen('215663');              % Generate, from RA, the
base numbers
[L, Izz, M0, H, B] = modeling_data_gen(base_numbers) % Generate, from the
Base numbers, the necessary parameters for modeling
```

```
L = 63
Izz = 0.3325
M0 = 21000
H = 1.8600
B = 0.6200
```

```
P = M0 ./ L;
X = 0:0.01:L;

% First method, using sing_function
Mz_1 = (P.*L) - M0 - (P.*X) + M0 .* sing_function(X - L./2, 0);

% Second method, using for, if statement
Mz_2 = zeros(size(X)); %Initialize with all zeros

for i = 1:numel(X)
    if (X(i) >= L/2)
        Mz_2(i) = (P.*L) - M0 - (P.*X(i)) + M0;
    else
        Mz_2(i) = (P.*L) - M0 - (P.*X(i));
    end
end

hold on, grid on;
plot(X, Mz_1 .* 1e-3, LineWidth=2);      % Plot first method, with solid
lines
plot(X, Mz_2 .* 1e-3, ":", LineWidth=3); % Plot second method, with dotted
lines so we can see both overlapping

title('Momento Fletor');
```

```
legend('sing function', 'for loop');
xlabel('Length (m)'), ylabel('Momento Feltor (KN \cdot m)');
annotation(gcf,'textarrow',[0.44765625 0.47734375],[0.851688693098385
0.804698972099853],'String',{'M_{Z_{max}}'});
hold off;
```



Além de observarmos que ambos os metodos resultam no mesmo gráfico, observamos também que o valor máximo do momento fletor é dado por $M_{z_{max}} = 10.5KN \cdot m$, no ponto $x = L/2 = 31.5m$.

1.2 Deflexão

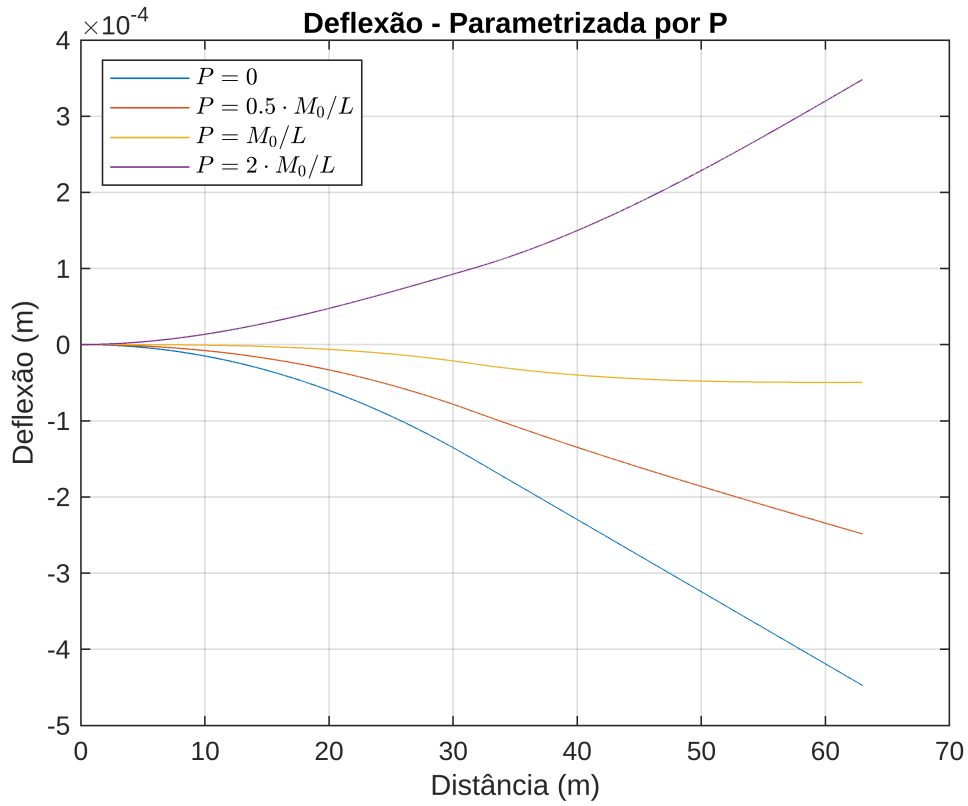
A deflexão da viga é dada por:

$$\nu_y(x) = \frac{1}{EI_{zz}}\left[(PL - M_0)\frac{x^2}{2} - \frac{P}{6}x^3 + \frac{M_0}{2}\left\langle x - \frac{L}{2}\right\rangle^2\right]$$

Que podemos ver ser dependente do load P sendo aplicado. A fim de estudar a deflexão como uma função de x, mas "parametrizada" por P, plotaremos seu valor para diferentes valores do load $P = \left\{0, \frac{M_0}{2L}, \frac{M_0}{L}, \frac{2M_0}{L}\right\}$:

```
E = 210e9;  % Elasticity coefficient
RHO = 7850; % Density

P = [0; (M0/(2*L)); M0/L; (2*M0/L)]; % All values of the external load to be
analyzed
```

```
v_y = (1/(E * Izz)) .* ((P .* L - M0) .* (X.^2)./2 - (P .* X.^3)./6 +
(M0./2).*sing_function(X - L/2, 2));

plot(X, v_y);
grid on;
title('Deflexão - Parametrizada por P');
xlabel('Distância (m)'), ylabel('Deflexão (m)');
legend('$P = 0$', '$P = 0.5 \cdot M_0/L$', '$P = M_0/L$', '$P = 2\cdot M_0/
L$', 'Interpreter', 'latex', Location='northwest');
```



1.3 Tensão

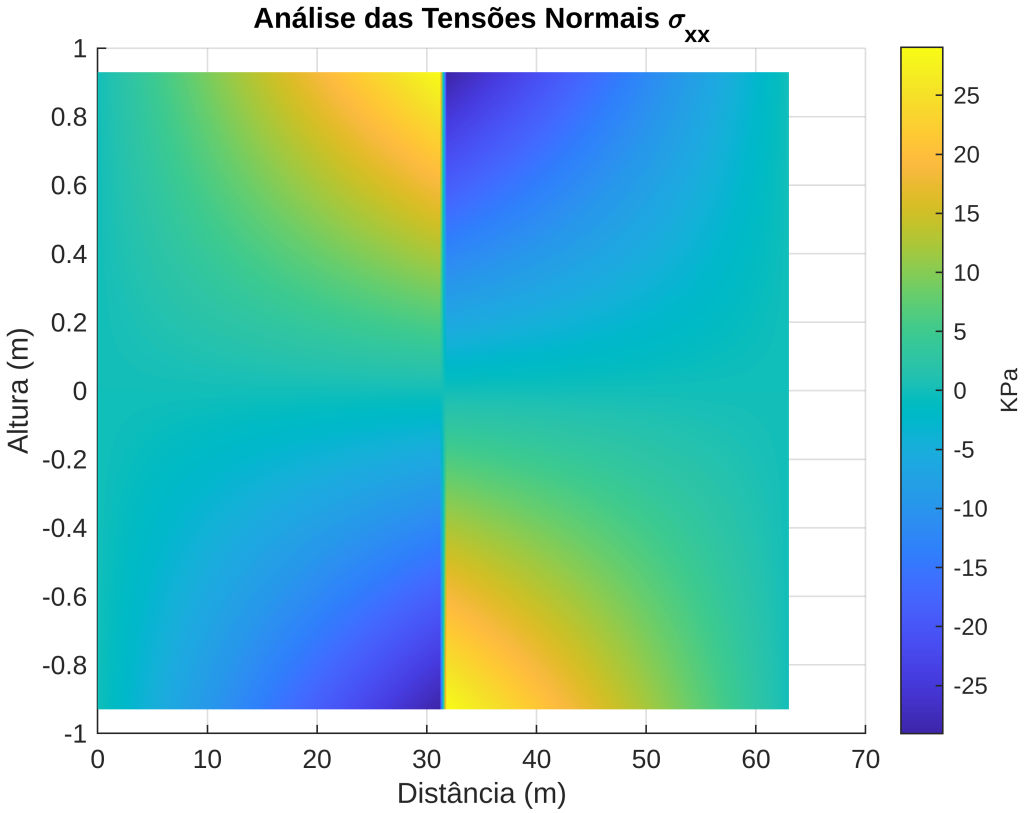
A tensão normal depende tanto do momento fletor no ponto x, mas também da altura no ponto da viga sob análise (tendo como ponto de origem da coordenada y o meio da viga), como mostrado abaixo:

$$\sigma_{xx}(x, y) = -\frac{y}{I_{zz}} M_z(x), \qquad -h/2 \leq y \leq h/2$$

A fim de analisarmos essa função do tipo $z = f(x, y)$, para o load $P = M_0/L$, iremos analisar a superfície resultante:

```
P = M0 / L;
Y = linspace(-H/2, H/2, 100);
X = linspace(0, L, 100);

[X, Y] = meshgrid(X,Y);
```

```
sigma = -(Y ./ Izz) .* ((P.*L) - M0 - (P.*X) + M0 .* sing_function(X - L./2,
0));

surf(X, Y, sigma * 1e-3);
view(2);

c_bar = colorbar;
shading interp;
title('Análise das Tensões Normais \sigma_{xx}');
xlabel('Distância (m)'), ylabel('Altura (m)');
c_bar.Label.String = 'KPa';
```



1.4 Discução

Visualização de Dados

A visualização de dados acima podem ser divididos em 2 tipos:

- 2D - Utilizando a função `plot`, usando uma discretização de `0.01`
- 3D -Utilizando a função `meshgrid` para criar um sistema favorável para, posteriormente, se utilizar a função `surf`, que cria uma superfície tri-dimensional. Analizando o plot em um plano e escala de cor indicando os valores em z

As discretizações supracitadas foram escolhidas para que possamos analizar de forma mais detalhada os comportamentos dos gráficos.

Além disso, algo importante de se exaltar são as unidades referentes a cada `axis` dos gráficos. Eles foram escolhidos a fim de facilitar a análise dos gráficos, evitando utilizar notação científica e botando os resultados nas ordens de grandeza mais utilizados em engenharia (como tensão em KPa, representando $10^3 Pa$).

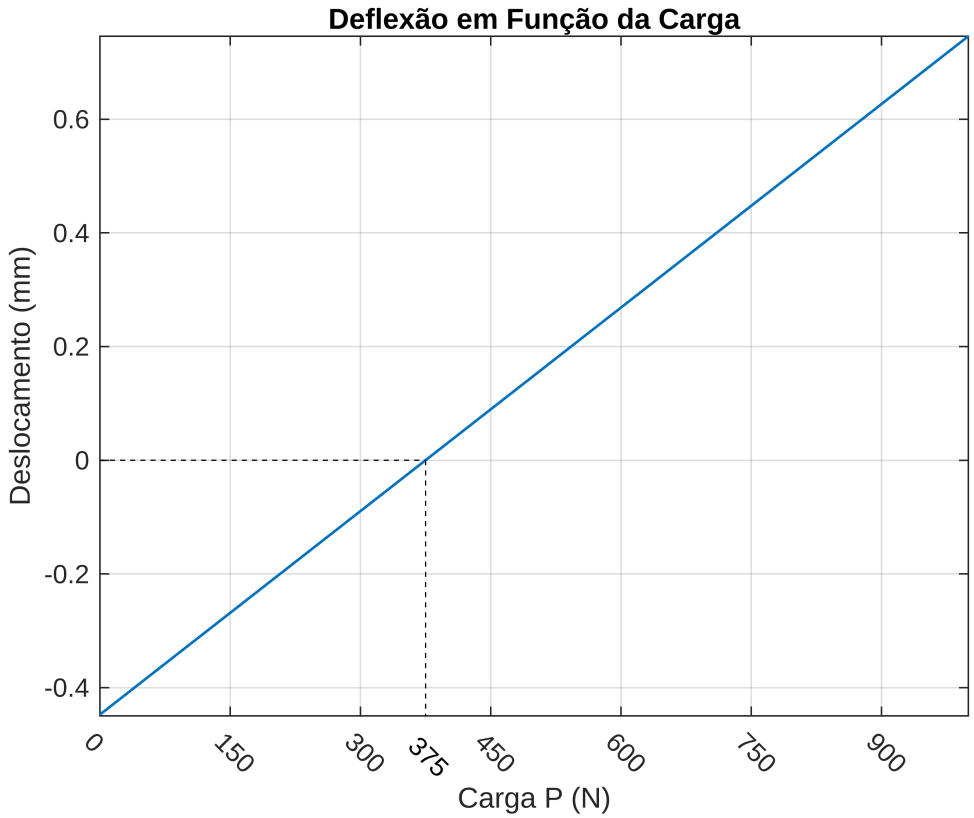
2. Deflexão em Função da Carga P

Como vimos, a deflexão e dada por:

$$\nu_y(x) = \frac{1}{EI_{zz}}\left[(PL - M_0)\frac{x^2}{2} - \frac{P}{6}x^3 + \frac{M_0}{2}\left\langle x - \frac{L}{2}\right\rangle^2\right]$$

Que é uma função de x e é parametrizada por P (a carga). Dependendo do valor dessa carga externa temos que a deflexão na ponta $(x = L)$ pode ser tanto negativa quanto positiva. A fim de estudar o comportamento da deflexão na ponta, em função do parâmetro P, temos:

```
F_v_y = @(p) (1/(E * Izz)) .* ((p .* L - M0) .* (L.^2)./2 - (p .* L.^3)./6 +
(M0./2).*sing_function(L - L/2, 2));
deflex_zero = fzero(F_v_y, 4e2);

fplot(@(p) 1e3*F_v_y(p),[0 1000], LineWidth=1);
hold on;

line([deflex_zero, deflex_zero], [0 -0.45], 'LineStyle', '--', 'Color',
'black');
line([0, deflex_zero], [0 0], 'LineStyle', '--','Color', 'black');

title('Deflexão em Função da Carga');
xlabel('Carga P (N)'), ylabel('Deslocamento (mm)');
xticks(0:150:1e3);
xtickangle(-45);
annotation(gcf,'textbox',...
    [0.41 0.10952380952381 0.4757857142857144 0.0104761904761906],...
    'String','375',...
    'Rotation',-45,...
    'FitBoxToText','off',...
    'EdgeColor','none');

grid on;

hold off
```



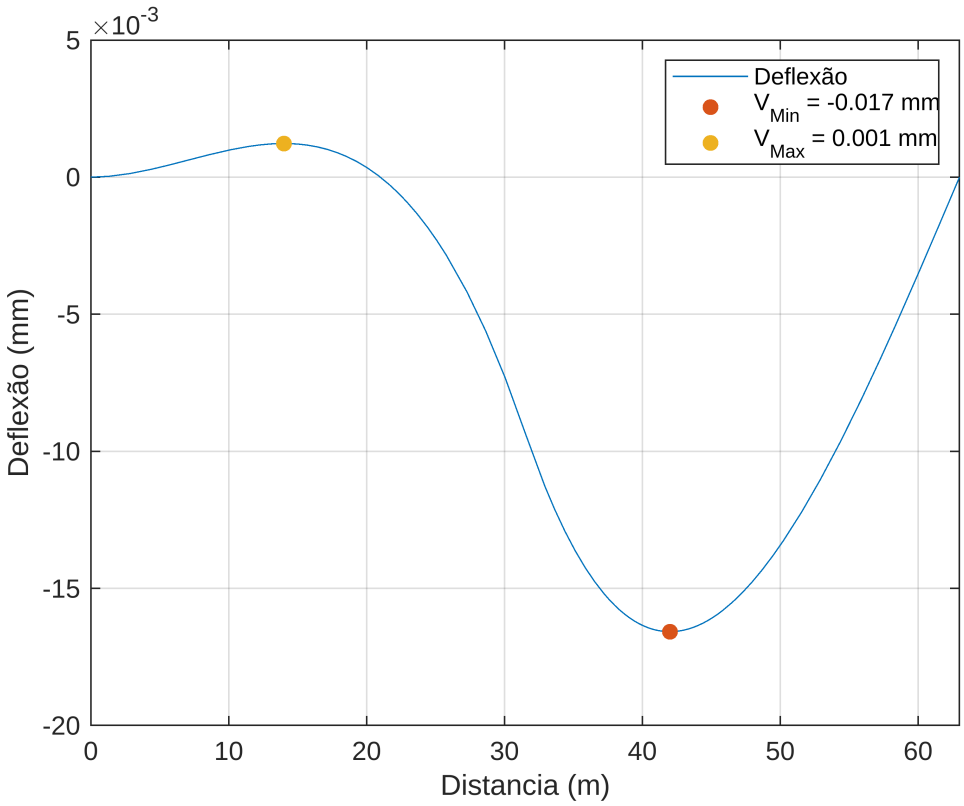
Onde temos que a deflexão na ponta da viga é zero para uma carga positiva de $P = 375N$. Fomos capazes de achar o valor do zero da função através do chute inicial de $P_0 \approx 400$, que foi escolhido depois de plotar a função com o comando `fplot` e observarmos que era aproximadamente o valor por onde a curva passava pelo zero.

3. Deflexão Máxima

Para o caso anteorior, onde a flecha da extremidade livre da viga é nulo, e o carregamento externo P é dado por $P = 375$, temos que, em diferentes pontos, ocorrem pontos de deflexão máxima e outro de mínima, sendo essas dadas por:

```
P = deflex_zero;
v_y = @(x) (1/(E * Izz)) .* ((P .* L - M0) .* (x.^2)./2 - (P .* x.^3)./6 +
(M0./2).*sing_function(x - L/2, 2));

X_min_vy = fminbnd(v_y, 0, L);
X_max_vy = fminbnd(@(x) -v_y(x), 0, L);

V_y_min = v_y(X_min_vy);
V_y_max = v_y(X_max_vy);

fplot(@(x) 1e3*v_y(x), [0 L]), hold on;
scatter(X_min_vy, 1e3 * v_y(X_min_vy), 'filled');
scatter(X_max_vy, 1e3 * v_y(X_max_vy), 'filled');
```

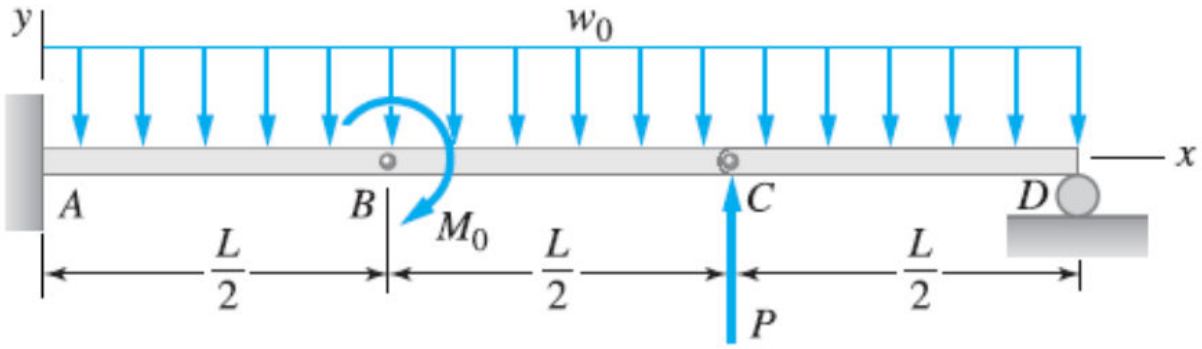
```
legend('Deflexão', sprintf('V_{Min} = %05.3f mm', V_y_min * 1e3),
sprintf('V_{Max} = %05.3f mm', V_y_max * 1e3));
ylim([-20e-3 5e-3]);
xlabel('Distancia (m)'), ylabel('Deflexão (mm)');
grid on;
hold off;
```



Para encontrarmos tais valores de máximo e mínimo utilizamos a função $fminbnd$ que, como o próprio nome indica, é utilizada para achar o ponto em X onde ocorre o mínimo da função, onde X é "Bounded", $i.e$ limitado, sendo para o nosso caso limitado pelo tamanho da viga (que vai de 0 até L).

4. Forças de Reação

Outra análise muito importante a ser feita durante os estudos e projetos de Resistência dos materiais é a ordem de grandeza e valores das forças de reação presente nos sistemas. A fim de estudarmos mais a fundo tal tópico, utilizaremos o sistema de Viga Engastada-Apoiada descrita abaixo:



Para esse caso, nós teremos um sistema de equações, originada a partir do Método das Equações Diferenciais de Equilíbrio, resultando em:

```
% Constantes
g  = 9.81;
w0 = RHO * B * H * g;
P = [0; (M0/(2*L)); M0/L; (2*M0/L)];

% Preparando a matriz do sistem de equa. [A]{x} = {b}, x = {Ra, Ma, Rb}'
A = zeros(3);
b_case_a = zeros(3, 4);    % b para diferentes valores de p
b_case_b = zeros(3, 2);    % b para o valor de P = M/L, com e sem peso
%%% Populando Matrizes %%%
% Primeira Eq.
A(1, 1) = L;
A(1, 3) = 1;

b_case_a(1, :) = ones(1, 4) * (-M0 + w0 * L^2/2);
b_case_b(1, :) = [(-M0 + w0 * L^2/2), (-M0)];

% Segunda Eq.
A(2, 1) = (3/2) * L - L;
A(2, 2) = 1;

b_case_a(2, :) = -M0 + (w0*(3*L/2)^2 /2) - P.*(3*L/2 - L);
b_case_b(2, :) = [(-M0 + (w0*(3*L/2)^2 /2) - (M0/L).*(3*L/2 - L)), (-M0 +
(M0/L).*(3*L/2 - L))];

% Terceira Eq.
A(3, 1) = 1;
A(3, 3) = 1;

b_case_a(3, :) = -P + w0*(3*L/2);
b_case_b(3, :) = [(-(M0/L) + w0*(3*L/2)), (-M0/L)];

%%% Calculation %%%
dA = decomposition(A);
```

```
results_a = dA\b_case_a;
results_b = dA\b_case_b;

results_case_a = table([results_a(:, 1)], [results_a(:, 2)], [results_a(:,
3)], [results_a(:, 4)],'VariableNames',{'P=0','P=(1/2)M/L', 'P=M/L', 'P=2M/
L'});
results_case_b = table([results_b(:, 1)], [results_b(:, 2)],
'VariableNames', {'w0 != 0', 'W0 = 0'});

results_case_a.Properties.RowNames = {'Ra', 'Ma', 'Rb'};
results_case_b.Properties.RowNames = {'Ra', 'Ma', 'Rb'};

results_case_a, results_case_b
```

results_case_a = 3×4 table

	P=0	P=(1/2)M/L	P=M/L	P=2M/L
1 Ra	2.7068e+06	2.7068e+06	2.7068e+06	2.7068e+06
2 Ma	3.1125e+08	3.1124e+08	3.1123e+08	3.1122e+08
3 Rb	5.6854e+06	5.6852e+06	5.6850e+06	5.6847e+06

results_case_b = 3×2 table

	w0 != 0	W0 = 0
1 Ra	2.7068e+06	-333.3333
2 Ma	3.1123e+08	0
3 Rb	5.6850e+06	0

Análise

Onde podemos ver que, na primeira tabela, temos as forças de reação para diferentes Loads (P) aplicado. Já a segunda tablea nos mostra, para P=M/L, as forças se considerarmos ou náo o peso distribuido da propria viga.

Auxiliares

```
function [f_x] = sing_function(f,n)
% SING_FUNCTION Returs the resulting values for the Singularity function
% <f>^n, which is zero for f<0 and f^2 otherwise.
%
% [f_x] = SIGN_FUNCTION(X - L./4, 2) Returns, over the X vector,
% the result of the singularity function <x - L/4>^2, which is zero for
% x< L/4 and (x - L/4)^2 for x>= L/4

    f_x = (f >= 0) .* (f .^ n); % If f(x) greater than 0, y = f(x)^n, else y
= 0
end
```

```
function [L, Izz, M0, h, b] = modeling_data_gen(d)
% MODELING_DATA_GEN Generates all the necessay parameters for the modeling
% of a beam
% [L, Izz, M0] = MODELING_DATA_GEN([2 1 5 6 6 3]) Returns L = length, Izz =
% Second Area Moment of Inertia, M0 = Moment being applied

    % Calculate L
    L = 10 .* d(5) + d(6);

    % Calculate Izz
    b = (10 .* d(3) + 2 .* d(4)) .* 1e-2;
    h = 3 .* b;
    Izz = (b .* h.^3)./12;

    % Calculate M0
    M0 = (10 .* d(1) + d(2)) .* 1e3;

end
```

```
function [n] = number_gen(RA)
% NUMBER_GEN Returs the numbers to be used for beam modeling
%   [n] = NUMBER_GEN('215663') Returns [2 1 5 6 6 3], 1x6 row vector

    s_RA  = regexp(RA, '\d', 'match'); % Get as a cell array all the number
charcters on the string
    s_RA  = string(join(s_RA, ' '));   % Join all the char separated with a
space to be compliant with the str2 num function

    n = str2num(s_RA);                 % Convert the number from char to
numeric
end
```