1.º Projeto Computacional

Relatório de Mecânica e Modelação Computacional

Docente: João Folgado

Grupo 14

Frederico Gonçalves - ist1106088
Gabriela Gomes - ist1106056
Sofia Costa - ist1106821

Outubro de 2024

1 Métodos

O programa de *Python* feito neste projeto lê um ficheiro `.txt`, processa a informação do mesmo e retorna um novo ficheiro com os resultados e um gráfico representativo dessas mesmas conclusões. Vamos então explicar o código desenvolvido.

Leitura do ficheiro

O ficheiro de texto (`data.txt`) que criámos conteve informações sobre o material, as coordenadas dos nós (em metros), as conectividades de elementos, condições fronteira e forças existentes e os seus respetivos ângulos em relação ao eixo do x. Para além disso adicionámos duas constantes: o Módulo de Young (Pa) e a *Cross Sectional Area* (m^2). Cada linha do ficheiro .txt é lida e armazenada quando a classe `sistema` é inicializada. As coordenadas (`self.coord`), forças (`self.forca`), a *Cross Sectional Area* (`self.a`) e o módulo de Young (`self.e`) são armazenados como *float*, as conectividades (`self.connect`) e condições fronteira (`self.front`) como *int*. Ainda armazenámos também o produto da *Cross Sectional Area* com o Módulo de Young (`self.ea`). O número de nós (`self.n_pont`) e elementos (`self.n_elem`) é o comprimento da lista das coordenadas e de conectividades, respetivamente, enquanto que o número de deslocamentos (`self.n_desloc`) é o dobro do número de nós.

f_dist

Para obter o comprimento de um elemento calculámos a distância entre os dois nós do mesmo, cujas coordenadas conseguimos obter a partir da conectividade do elemento, utilizando a seguinte expressão matemática: $\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$

f_ang

Para descobrir o ângulo do elemento em relação ao eixo do x calculámos, em primeiro lugar, a distância entre as coordenadas x (`cat_x`) e y (`cat_y`) dos nós do elemento. Em seguida, resolvemos $arctan = \frac{cat\_y}{cat\_x}$. Relativamente às componentes x e y das forças, utilizámos o cosseno e o seno, respetivamente e fizemos a mudança de graus para radianos usando a função `m.radians(a)` do módulo `math`.

f_k_elem

Para a determinação da matriz rigidez para cada elemento, K_e começámos por criar uma matriz 4x4 com zeros e foram então utilizadas as funções `f_ang(e)`, `f_dist(e)`, referidas anteriormente, e foi calculado o coeficiente, `coef = self.ea/h_e`. Assim, preenchemos a matriz de acordo com a fórmula apresentada na seguinte figura 1.

$$\left[K^e\right]=\frac{EA}{h^e}\left[\begin{array}{cc:cc} c^2 & c\cdot s & -c^2 & -c\cdot s \\ c\cdot s & s^2 & -c\cdot s & -s^2 \\ \hdashline -c^2 & -c\cdot s & c^2 & c\cdot s \\ -c\cdot s & -s^2 & c\cdot s & s^2 \end{array}\right] \qquad \text{onde,} \quad \begin{array}{l} c=\cos(\theta^e) \\ s=\sin(\theta^e) \end{array}$$

Figura 1: Matriz Rigidez para cada elemento de Treliças 2D

f_k_tot

Com a determinação da matriz rigidez para cada elemento foi possível determinarmos a matriz rigidez completa, K_{tot}. Inicializámos novamente a zeros e percorremos os elementos, guardando os nós correspondentes. A um dado nó i correspondem os graus de liberdade $2i-1$ e $2i$ (2i e 2i+1 no código, pois os índices começam em 0 em vez de 1). Assim, o vetor *et* armazena os índices dos graus de liberdade (ou deslocamentos) dos nós do elemento, p1 e p2, que indicam os índices da matriz global. Os valores de K_e são somados nessas posições globais de K_{tot}.

f_correcao

Esta função aplica as condições fronteira e insere as forças na matriz de rigidez global K_{tot}, adicionando-as a uma coluna adicional e transformando-a numa matriz alargada K_{tot_ala}. Além disso, faz a correção para deslocamentos nulos, ou seja, redefine a linha correspondente na matriz para garantir que o deslocamento seja zero. Os valores da linha são substituídos por zero à exceção da diagonal principal que é substituída por 1. Desta maneira a matriz representa o sistema de equações do problema, com base na relação entre a matriz compressibilidade, vetor de deslocamentos e vetor de forças, represntado na equação (1).

$$[K]u = F \tag{1}$$

f_resolver

Para resolvermos o sistema de equações, ou seja resolvermos a matriz K_{tot_ala} utilizamos o Método de eliminação de Gauss, começando por tornar a matriz triangular superior e criamos o *vetor_u* inicializado a zeros. Posteriormente resolvemos o sistema e calculamos os deslocamentos, que armazenamos no *vetor_u*.

f_reacoes

Já sabendo os deslocamentos podemos então calcular as Reações nos apoios. Para isso calculamos o vetor de forças (*vetor_f*) multiplicando a matriz rigidez total pelo vetor de deslocamentos, segundo a equação (1).

De seguida inicializamos o *vetor_reacoes* a zeros, onde vamos armazenar as reações. Para o cálculo apenas subtraimos as componentes x (índices pares) e y (índices ímpares) das forças aplicadas, que são conhecidas, ao vetor de forças de acordo com a equação (2).

$$F = F_{aplicadas} + R <=> R = F - F_{aplicadas} \tag{2}$$

Sabendo assim as reações nos apoios em x e em y de todos os nós.

f_tensoes

As tensões são calculadas através do método linear, com base na extensão de cada elemento e nas propriedades do material (Módulo de Young, E), de acordo com as equações (3) e (4).

$$\varepsilon_{AB} = \frac{u_B - u_A}{L} \tag{3}$$

$$\sigma = E * \varepsilon \tag{4}$$

Com isto temos então de ir buscar os deslocamentos que correspondem ao referencial da barra, já que o *vetor_u* representa os deslocamentos no referencial global. Para isso usamos a transformação de coordenadas referidas nas equações (5).

$$u_A = cos * vetor\_u_1 + sen * vetor\_u_2 \quad , \quad u_B = cos * vetor\_u_3 + sen * vetor\_u_4 \tag{5}$$

Isto quando o nó A está associado ao referencial u_1 e u_2 e o nó B ao referencial u_3 e u_4. Assim podemos calcular as tensões, de acordo com a equação (6), e guardá-las no *vetor_tensoes*.

$$\sigma_{AB} = E * \frac{u_B - u_A}{L} \tag{6}$$

f_atualizar_coords

Por fim, atualizamos as coordenadas dos nós para construirmos o gráfico com as deformações. Para isso fizemos uma cópia do vetor das coordenadas iniciais e adicionámos os deslocamentos que cada nó sofreu, multiplicádos por um fator. Este ou é subemtido como argumento da função principal (`projeto_1`) ou é calculado com base no módulo máximo de deslocamento e no cumprimento máximo dos elementos como representado na equação (7) de maneira a replicar o cálculo do *Scaling factor* por parte do *Abaqus* [1]. Assim as deformações são visíveis.

$$fator = \frac{L_{max}}{10 * u_{max}} \tag{7}$$

f_mostrar e projeto_1

No final, as coordenadas originais, os deslocamentos e as tensões são utilizados para fazer um gráfico com as configurações original e deformada da estrutura, e tensões através de um esquema de cores. Para além disso, criamos um ficheiro `Resultados.txt`, com todas as informações úteis para a análise da estrutura: coordenadas iniciais, conectividades, condições fronteira, forças aplicadas, Módulo de Young, *Cross Sectional Area*, vetor dos deslocamentos, vetor das reações e vetor das tensões.

2 Discussão e Resultados

Na Figura 2 conseguimos observar a estrutura dada no enunciado. Esta apresenta barras de aço, pelo que utilizámos o módulo de Young correspondente: 200 GPa [2].

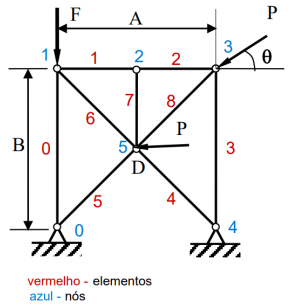

Figura 2: Configuração inicial

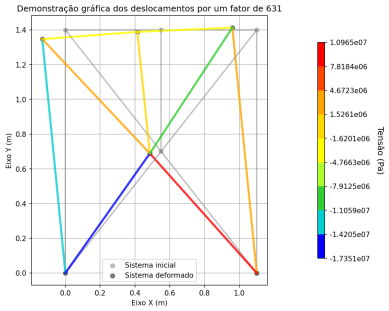

Figura 3: Configuração deformada obtida através do código

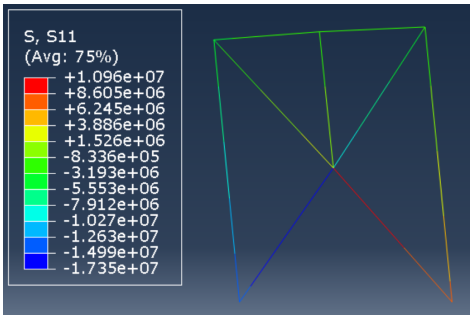

Figura 4: Configuração deformada obtida através do *Abaqus*

Como podemos reparar pela Figura 3 e 4, as configurações deformadas obtidas são idênticas no *Python* e no *Abaqus*, tal como era de esperar. A análise feita a partir do *Abaqus* serviu de confirmação para a figura e valores obtidos através do código em *Python*. Como os deslocamentos são infinitesimais, o fator utilizado foi de 631 em *Python*, que aumenta 631 vezes os deslocamentos. Cálculamos este fator automaticamente da mesma forma que o *Abaqus*, ficando assim o valor arredondado do *Abaqus* (630,664). Assim, na figura 3 conseguimos comparar a configuração inicial (em cinzento) e a configuração deformada, que apresenta as cores respetivas às tensões.

Deslocamentos Nodais

A tabela 1 apresenta os valores dos deslocamentos nodais em x e em y (metros) obtidos tanto na implementação do código como no *Abaqus*.

Tabela 1: Valores dos deslocamentos em x e em y, em metros, no código e no *Abaqus*

	Código		***Abaqus***	
Nó	Deslocamento em x (m)	Deslocamento em y (m)	Deslocamento em x (m)	Deslocamento em y (m)
0	0.00000000e+00	0.00000000e+00	-3.43034e-33	-8.21574e-33
1	-2.10511411e-04	-8.42153750e-05	-2.10512e-04	-8.42154e-05
2	-2.16249766e-04	-1.80757155e-05	-2.16250e-04	-1.80757e-05
3	-2.21988122e-04	2.09299287e-05	-2.21988e-04	2.09300e-05
4	0.00000000e+00	0.00000000e+00	-2.16774e-33	3.71574e-33
5	-1.02000515e-04	-1.80757155e-05	-1.02001e-04	-1.80757e-05

Tabela 2: Desvios relativos dos deslocamentos obtidos pelo código em comparação com os calculados pelo *Abaqus*

	Desvios relativos	
Elemento	Deslocamento em x	Deslocamento em y)
0	—	—
1	2.79794e-06	2.96858e-07
2	1.08208e-06	8.57505e-07
3	5.49579e-07	3.40661e-06
4	—	—
5	4.75488e-06	8.57504e-07

Através do código, constatámos que nos nós 0 e 4, que funcionam como apoios, não ocorrem deslocamentos nas direções x e y, uma vez que esses valores são iguais a zero. Em contraste, os resultados obtidos pelo *Abaqus* mostram que os deslocamentos não são exatamente zero, mas estão significativamente próximos, apresentando uma ordem de grandeza menor ou igual a 10^{-32}. Ao comparar os restantes nós, verificamos que os valores de deslocamento são semelhantes no código e no *Abaqus*. Os deslocamentos em x são todos negativos, pois as forças nessa direção atuam no sentido negativo do eixo x. Já em y há forças em ambos sentidos (forças aplicadas e reações nos apoios), logo o sentido do deslocamento depende do nó considerado.

Reações

Na tabela 3 encontram-se apresentados os valores das reações (Newton) em x e em y, obtidos através da implementação do

código e do *Abaqus*.

Tabela 3: Valores das Reações em x e em y, em Newtons, no código e no *Abaqus* e os seus respetivos desvios relativos

	Código		***Abaqus***		**Desvios relativos**	
Nó	Reações em x (N)	Reações em y (N)	Reações em x (N)	Reações em y (N)	Reações em x	Reações em y
0	3.43034029e+03	8.21573336e+03	3.43034e+03	8.21574e+03	8.45397e-08	8.08205e-07
1	-2.45743377e-12	9.09494702e-13	0.	0.	—	—
2	7.28313281e-29	1.84863987e-13	0.	0.	—	—
3	-2.27373675e-12	1.59161573e-12	0.	0.	—	—
4	2.16773592e+03	-3.71573336e+03	2.16774e+03	-3.71574e+03	1.88215e-06	1.78699e-06
5	-1.81898940e-12	-1.27688874e-12	0.	0.	—	—

No código, as reações nos nós 1, 2, 3 e 5 deveriam ser nulas, dado que esses nós não têm qualquer apoio. No entanto, os valores obtidos são muito próximos de zero (ordem de grandeza menor ou igual a 10^{-12}), o que não compromete os resultados finais. Este pequeno desvio pode resultar de aproximações mínimas feitas pelo *Python* na implementação do código. Os resultados obtidos pelo código nos nós 1 e 4 estão de acordo com os obtidos no *Abaqus*.

Tensões

Na tabela 4 estão representados os valores das tensões aplicadas em cada elemento obtidos pela implementação do código e pelo *Abaqus*.

Tabela 4: Valores das Tensões obtidos pelo código e pelo *Abaqus*

	Código	***Abaqus***	Desvios
Elemento	Tensão (Pa)	Tensão (Pa)	
0	-1.203076e+07	-1.20308e+07	3.32480e-06
1	-2.086674e+06	-2.08668e+06	2.8754e-06
2	-2.086674e+06	-2.08668e+06	2.8754e-06
3	2.989989e+06	2.99e+06	3.67893e-06
4	1.096461e+07	1.09646e+07	9.12026e-07
5	-1.735098e+07	-1.7351e+07	1.15267e-06
6	3.377471e+06	3.37748e+06	2.66471e-06
7	0.	0.	—
8	-9.763844e+06	-9.76385e+06	6.14512e-07

Os valores para as tensões no código e no *Abaqus* são muito semelhantes. O *Abaqus* calcula as tensões, por definição, com base no método linear, tal como foi implementado no código. O elemento 7 não está sujeito a tensões, por ambos os métodos, como seria de esperar, uma vez que no nó 2, a resultante das forças tanto em x como em y são zero. Deste modo, como há uma única força em y, esta terá de ser também igual a zero, pelo que a tensão é nula. Os elementos 0, 1 ,2, 5 e 8 estão sujeitos a compressão (sinal negativo), o que faz sentido devido às forças F e P, e os elementos 3, 4 e 6 estão sujeitos a tração (sinal positivo), devido às forças de reação nos apoios e à força P.

Referências

[1] ABAQUS/CAE User's Manual,
`https://classes.engineering.wustl.edu/2009/spring/mase5513/abaqus/docs/v6.6/books/usi/default.htm?startat=pt05ch36s04hlb01.html`

[2] Unit of Young's Modulus of Elasticity,
`https://www.nuclear-power.com/nuclear-engineering/materials-science/material-properties/strength/hookes-law/unit-of-youngs-modulus-of-elasticity/`