

```

1  import unittest
2  import geometry
3  import helpers
4  import numpy as np
5  import math
6  import boom
7  import edges
8  import DiscreteSection
9  import matplotlib.pyplot as plt
10
11 def main():
12     stringer_area = 42 *10**(-6)
13     neutral_axis = (0, 1, 0)
14
15     # CREATE LIST OF COORDINATES FOR BOOMS
16     # give the coordinates of the booms with respect to the hinge point (origin of coordinates)
17     # for the first eight, they are on a straight line
18     coordinates = []
19     for n in range(16):
20         coordinates.append([(43.45 - 5.43125/2 * (n + 1)) * 10) *10**(-3), ((1.40625/2 * (n + 1)) * 10)*10**(-3)])
21     # booms 8, 9 and 10 are along a semi-circle
22     coordinates.append([-112.5*10**(-3) * math.sin(math.pi / 8), 112.5*10**(-3) * math.cos(math.pi / 8)])
23     coordinates.append([-112.5*10**(-3) * math.sin(math.pi / 4), 112.5*10**(-3) * math.cos(math.pi / 4)])
24     coordinates.append([-112.5*10**(-3) * math.sin(3 * math.pi / 8), 112.5*10**(-3) * math.cos(3 * math.pi / 8)])
25     coordinates.append([-112.5*10**(-3), 0.0])
26
27     # the last 8 are symmetric to the first eight wrt the z-axis
28     for i in range(18, -1, -1):
29         coords = coordinates[i]
30         coordinates.append([coords[0], -coords[1]])
31     # the ones on the spar are always at z=0 and distributed equally along the height of the spar
32     coordinates.append([0.0, (22.5 + 45)*10**(-3)])
33     coordinates.append([0.0, 22.5*10**(-3)])
34     coordinates.append([0.0, -22.5*10**(-3)])
35     coordinates.append([0.0, (-22.5 - 45)*10**(-3)])
36
37
38     # CREATE BOOM INSTANCES AND INSERT THEM IN LIST OF BOOMS
39     booms = []
40     boom0 = boom.Boom(0, coordinates[0], 0.0, neutral_axis)
41     booms.append(boom0)
42     boom1 = boom.Boom(1, coordinates[1], stringer_area, neutral_axis)
43     booms.append(boom1)
44     boom2 = boom.Boom(2, coordinates[2], 0.0, neutral_axis)
45     booms.append(boom2)
46     boom3 = boom.Boom(3, coordinates[3], stringer_area, neutral_axis)
47     booms.append(boom3)
48     boom4 = boom.Boom(4, coordinates[4], 0.0, neutral_axis)
49     booms.append(boom4)
50     boom5 = boom.Boom(5, coordinates[5], stringer_area, neutral_axis)
51     booms.append(boom5)
52     boom6 = boom.Boom(6, coordinates[6], 0.0, neutral_axis)
53     booms.append(boom6)
54     boom7 = boom.Boom(7, coordinates[7], stringer_area, neutral_axis)
55     booms.append(boom7)
56     boom8 = boom.Boom(8, coordinates[8], 0.0, neutral_axis)
57     booms.append(boom8)
58     boom9 = boom.Boom(9, coordinates[9], stringer_area, neutral_axis)
59     booms.append(boom9)
60     boom10 = boom.Boom(10, coordinates[10], 0.0, neutral_axis)
61     booms.append(boom10)
62     boom11 = boom.Boom(11, coordinates[11], stringer_area, neutral_axis)
63     booms.append(boom11)
64     boom12 = boom.Boom(12, coordinates[12], 0.0, neutral_axis)
65     booms.append(boom12)
66     boom13 = boom.Boom(13, coordinates[13], stringer_area, neutral_axis)
67     booms.append(boom13)
68     boom14 = boom.Boom(14, coordinates[14], 0.0, neutral_axis)
69     booms.append(boom14)
70     boom15 = boom.Boom(15, coordinates[15], 0.0, neutral_axis)
71     booms.append(boom15)
72
73     # semi circle booms
74     boom16 = boom.Boom(16, coordinates[16], 0.0, neutral_axis)
75     booms.append(boom16)
76     boom17 = boom.Boom(17, coordinates[17], stringer_area, neutral_axis)
77     booms.append(boom17)
78     boom18 = boom.Boom(18, coordinates[18], 0.0, neutral_axis)
79     booms.append(boom18)
80     boom19 = boom.Boom(19, coordinates[19], stringer_area, neutral_axis)
81     booms.append(boom19)
82     boom20 = boom.Boom(20, coordinates[20], 0.0, neutral_axis)
83     booms.append(boom20)
84     boom21 = boom.Boom(21, coordinates[21], stringer_area, neutral_axis)
85     booms.append(boom21)
86     boom22 = boom.Boom(22, coordinates[22], 0.0, neutral_axis)
87     booms.append(boom22)
88
89     # lower straight line booms

```

```

90 boom23 = boom.Boom(23, coordinates[23], 0.0, neutral_axis)
91 booms.append(boom23)
92 boom24 = boom.Boom(24, coordinates[24], 0.0, neutral_axis)
93 booms.append(boom24)
94 boom25 = boom.Boom(25, coordinates[25], stringer_area, neutral_axis)
95 booms.append(boom25)
96 boom26 = boom.Boom(26, coordinates[26], 0.0, neutral_axis)
97 booms.append(boom26)
98 boom27 = boom.Boom(27, coordinates[27], stringer_area, neutral_axis)
99 booms.append(boom27)
100 boom28 = boom.Boom(28, coordinates[28], 0.0, neutral_axis)
101 booms.append(boom28)
102 boom29 = boom.Boom(29, coordinates[29], stringer_area, neutral_axis)
103 booms.append(boom29)
104 boom30 = boom.Boom(30, coordinates[30], 0.0, neutral_axis)
105 booms.append(boom30)
106 boom31 = boom.Boom(31, coordinates[31], stringer_area, neutral_axis)
107 booms.append(boom31)
108 boom32 = boom.Boom(32, coordinates[32], 0.0, neutral_axis)
109 booms.append(boom32)
110 boom33 = boom.Boom(33, coordinates[33], stringer_area, neutral_axis)
111 booms.append(boom33)
112 boom34 = boom.Boom(34, coordinates[34], 0.0, neutral_axis)
113 booms.append(boom34)
114 boom35 = boom.Boom(35, coordinates[35], stringer_area, neutral_axis)
115 booms.append(boom35)
116 boom36 = boom.Boom(36, coordinates[36], 0.0, neutral_axis)
117 booms.append(boom36)
118 boom37 = boom.Boom(37, coordinates[37], stringer_area, neutral_axis)
119 booms.append(boom37)
120 boom38 = boom.Boom(38, coordinates[38], 0.0, neutral_axis)
121 booms.append(boom38)
122
123 # booms on the spar
124 boom39 = boom.Boom(39, coordinates[39], 0.0, neutral_axis)
125 booms.append(boom39)
126 boom40 = boom.Boom(40, coordinates[40], 0.0, neutral_axis)
127 booms.append(boom40)
128 boom41 = boom.Boom(41, coordinates[41], 0.0, neutral_axis)
129 booms.append(boom41)
130 boom42 = boom.Boom(42, coordinates[42], 0.0, neutral_axis)
131 booms.append(boom42)
132
133
134 # CREATE EDGES INSTANCES AND PUT THEM IN EDGE LIST
135 edge_list = []
136 edge10 = edges.Edge([1, 0], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
137 edge_list.append(edge10)
138 edge21 = edges.Edge([2, 1], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
139 edge_list.append(edge21)
140 edge32 = edges.Edge([3, 2], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
141 edge_list.append(edge32)
142 edge43 = edges.Edge([4, 3], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
143 edge_list.append(edge43)
144 edge54 = edges.Edge([5, 4], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
145 edge_list.append(edge54)
146 edge65 = edges.Edge([6, 5], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
147 edge_list.append(edge65)
148 edge76 = edges.Edge([7, 6], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
149 edge_list.append(edge76)
150 edge87 = edges.Edge([8, 7], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
151 edge_list.append(edge87)
152 edge98 = edges.Edge([9, 8], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
153 edge_list.append(edge98)
154 edge109 = edges.Edge([10, 9], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
155 edge_list.append(edge109)
156 edge1110 = edges.Edge([11, 10], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
157 edge_list.append(edge1110)
158 edge1211 = edges.Edge([12, 11], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
159 edge_list.append(edge1211)
160 edge1312 = edges.Edge([13, 12], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
161 edge_list.append(edge1312)
162 edge1413 = edges.Edge([14, 13], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
163 edge_list.append(edge1413)
164 edge1514 = edges.Edge([15, 14], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
165 edge_list.append(edge1514)
166
167 # booms on semicircle
168 edge1615 = edges.Edge([16, 15], 1.1*10**(-3), 44.179*10**(-3))
169 edge_list.append(edge1615)
170 edge1716 = edges.Edge([17, 16], 1.1*10**(-3), 44.179*10**(-3))
171 edge_list.append(edge1716)
172 edge1817 = edges.Edge([18, 17], 1.1*10**(-3), 44.179*10**(-3))
173 edge_list.append(edge1817)
174 edge1918 = edges.Edge([19, 18], 1.1*10**(-3), 44.179*10**(-3))
175 edge_list.append(edge1918)
176 edge2019 = edges.Edge([20, 19], 1.1*10**(-3), 44.179*10**(-3))
177 edge_list.append(edge2019)
178 edge2120 = edges.Edge([21, 20], 1.1*10**(-3), 44.179*10**(-3))
179 edge_list.append(edge2120)
180 edge2221 = edges.Edge([22, 21], 1.1*10**(-3), 44.179*10**(-3))
181 edge_list.append(edge2221)

```



```

182 edge2322 = edges.Edge([23, 22], 1.1*10**(-3), 44.179*10**(-3))
183 edge_list.append(edge2322)
184
185 # booms on lower spar
186 edge2423 = edges.Edge([24, 23], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
187 edge_list.append(edge2423)
188 edge2524 = edges.Edge([25, 24], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
189 edge_list.append(edge2524)
190 edge2625 = edges.Edge([26, 25], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
191 edge_list.append(edge2625)
192 edge2726 = edges.Edge([27, 26], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
193 edge_list.append(edge2726)
194 edge2827 = edges.Edge([28, 27], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
195 edge_list.append(edge2827)
196 edge2928 = edges.Edge([29, 28], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
197 edge_list.append(edge2928)
198 edge3029 = edges.Edge([30, 29], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
199 edge_list.append(edge3029)
200 edge3130 = edges.Edge([31, 30], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
201 edge_list.append(edge3130)
202 edge3231 = edges.Edge([32, 31], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
203 edge_list.append(edge3231)
204 edge3332 = edges.Edge([33, 32], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
205 edge_list.append(edge3332)
206 edge3433 = edges.Edge([34, 33], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
207 edge_list.append(edge3433)
208 edge3534 = edges.Edge([35, 34], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
209 edge_list.append(edge3534)
210 edge3635 = edges.Edge([36, 35], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
211 edge_list.append(edge3635)
212 edge3736 = edges.Edge([37, 36], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
213 edge_list.append(edge3736)
214 edge3837 = edges.Edge([38, 37], 1.1*10**(-3), 56.103 * 0.5*10**(-3))
215 edge_list.append(edge3837)
216 edge038 = edges.Edge([0, 38], 1.1*10**(-3), 56.103*10**(-3))
217 edge_list.append(edge038)
218
219 # booms on the vertical spar
220 edge3915 = edges.Edge([39, 15], 2.9*10**(-3), 45*10**(-3))
221 edge_list.append(edge3915)
222 edge4039 = edges.Edge([40, 39], 2.9*10**(-3), 45*10**(-3))
223 edge_list.append(edge4039)
224 edge4140 = edges.Edge([41, 40], 2.9*10**(-3), 45*10**(-3))
225 edge_list.append(edge4140)
226 edge4241 = edges.Edge([42, 41], 2.9*10**(-3), 45*10**(-3))
227 edge_list.append(edge4241)
228 edge2342 = edges.Edge([23, 42], 2.9*10**(-3), 45*10**(-3))
229 edge_list.append(edge2342)
230
231 # CREATE INSTANCE OF AILERON GEOMETRY WITH ALL THE BOOMS AND EDGES
232 aileron_geometry = geometry.Geometry(43, booms, edge_list, [19880.391*10**(-6), 36225*10**(-6)], 28 * 10**9)
233 aileron_geometry.construct_geometry()
234 aileron_geometry.calc_centroid()
235 aileron_geometry.cells = [[edge038, edge3837, edge3736, edge3635, edge3534, edge3433, edge3332, edge3231, edge3130,
236 edge3029, edge2928, edge2827, edge2726, edge2625, edge2524, edge2423, edge2342, edge4241,
237 edge4140, edge4039, edge3915, edge1514, edge1413, edge1312, edge1211, edge1110, edge109,
238 edge98, edge87, edge76, edge65, edge54, edge43, edge32, edge21, edge10],
239 [edge2019, edge1918, edge1817, edge1716, edge1615, edge3915, edge4039, edge4140, edge4241,
240 edge2342, edge2322, edge2221, edge2120]]
241
242 # calculate areas and distances to centroid of all booms
243 for element in booms:
244     element.calculate_area(aileron_geometry)
245 for boom_element in booms:
246     boom_element.calc_y_dist(aileron_geometry)
247     boom_element.calc_z_dist(aileron_geometry)
248 aileron_geometry.get_areas()
249
250 # calculate moments of inertia
251 aileron_geometry.moment_inertia_Izz()
252 aileron_geometry.moment_inertia_Iyy()
253
254 # PLOT AND PRINT GEOMETRICAL PROPERTIES
255 aileron_geometry.plot_edges()
256 for it, el in enumerate(booms):
257     print('area of boom ', it, ' : ', aileron_geometry.boom_areas[it], '[mm^2]')
258 print('centroid position : ', aileron_geometry.centroid)
259 print('z moment of inertia : ', aileron_geometry.Izz, '[mm^4]')
260 print('y moment of inertia : ', aileron_geometry.Iyy, '[mm^4]')
261
262 # GET THE LIST OF FORCES AND MOMENTS
263 file_name = "Loads.txt"
264 x_i_array = helpers.get_array_x_i(file_name)
265 Mx_array = helpers.get_array_Mx_i(file_name)
266 My_array = helpers.get_array_My_i(file_name)
267 Mz_array = helpers.get_array_Mz_i(file_name)
268 Sz_array = helpers.get_array_Sz_i(file_name)
269 Sy_array = helpers.get_array_Sy_i(file_name)
270
271 # create a matrix of normal stresses
272 stress_matrix = np.zeros((43, 101))
273 for j, location in enumerate(x_i_array):

```

```

274         for i, boom_member in enumerate(aileron_geometry.booms):
275             boom_member.calc_bending_stress(Mz_array[j], My_array[j], aileron_geometry)
276             stress_matrix[i][j] = boom_member.bending_stress
277
278     # find maximum stress on each boom along the span
279     max_stress_matrix = np.amax(stress_matrix, axis=1)
280     print(max_stress_matrix)
281
282     # set up matrix of shear stresses
283     stress_matrix_shear = np.zeros((len(aileron_geometry.edges), 101))
284     # set up lists of twist rates and thetas
285     twist_rate_list = []
286     thetas_list = []
287     section_numbers = 100
288     step = 2.771 / section_numbers
289     thetas_list.append(0.453786)\
290     # create file to store the list of thetas
291     file = open("thetas_list.txt", "w")
292
293     for i, x_i in enumerate(x_i_array):
294         # create new instance of section with new location
295         aileron_section = DiscreteSection.DiscreteSection(neutral_axis, aileron_geometry)
296         # calculate shear flows due to pure shear and torque
297         aileron_section.calc_total_shear_flow(Sz_array[i], Sy_array[i], Mx_array[i], edge2342)
298         # calculate shear stress due to total shear flows and insert in the shear stress matrix
299         aileron_section.calc_shear_stress()
300         for nl, edge_ex in enumerate(aileron_geometry.edges):
301             stress_matrix_shear[nl][i] = edge_ex.shear_stress
302         # append the twist rate (computed at the same time as torque shear flow) in the twist rate list
303         twist_rate_list.append(aileron_section.twist_rate)
304         # calculate theta with finite differences, append to the list and copy to the txt file
305         theta = twist_rate_list[i-1] * step + thetas_list[i-1]
306         thetas_list.append(theta)
307         file.write(str(float(theta)) + '\n')
308     file.close()
309
310     # find the maximum shear stress on each rib
311     print('the maximum shear stress in rib A : ', np.max(stress_matrix_shear[:, 97]))
312     print('the maximum shear stress in rib B : ', np.max(stress_matrix_shear[:, 51]))
313     print('the maximum shear stress in rib C : ', np.max(stress_matrix_shear[:, 41]))
314     print('the maximum shear stress in rib D : ', np.max(stress_matrix_shear[:, 18]))
315
316     # find the maximum normal stress on each rib
317     print('the maximum normal stress in rib A : ', np.max(stress_matrix[:, 97]))
318     print('the maximum normal stress in rib A : ', np.max(stress_matrix[:, 51]))
319     print('the maximum normal stress in rib A : ', np.max(stress_matrix[:, 41]))
320     print('the maximum normal stress in rib A : ', np.max(stress_matrix[:, 18]))
321
322     # plot twist angle along the span
323     plt.plot(x_i_array, thetas_list[:-1])
324     plt.show()
325
326     main()
327

```