Design Problem 1

July 12, 2020

This notebook is my solution for Design Problem #1 of the Structural Mechanics course of MIT Open Course Work.

Paul Lagace. 16.20 Structural Mechanics. Fall 2002. Massachusetts Institute of Technology: MIT OpenCourseWare, https://ocw.mit.edu. License: Creative Commons BY-NC-SA.

[1]:
```
import pandas as pd
import numpy as np
import math
```

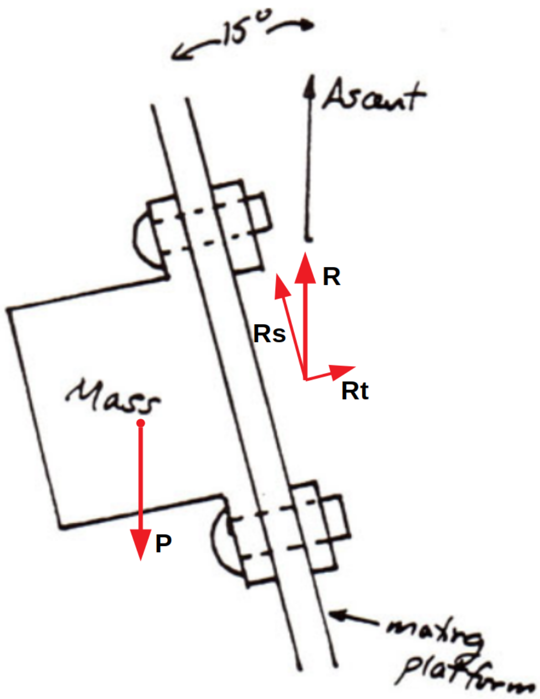
[2]:
```
# set up the initial parameters
M = 2100  # kg, mass of the payload
theta = 15 # deg., angle between fastener plane and ascent direction
N_limit = 3  # g
FS = 1.3
g = 9.81  # m/s^2
```

As the launch vehicle accelerates upwards there is an inertial load that pulls the payload down in the opposite direction. Our ultimate load, P, on the payload is determined by multiplying the limit load, factor of safety, mass, and gravity.

$P = FS * N_{limit} * g * M$

Normally for a fastener analysis, we take into account the distance between the center of all fasteners and the center of mass of the payload because this will create an overturning moment that increase the tension in some of the fasteners. Because the problem does not include detailed dimensions, I'm going to assume that this overturning moment is negligible and outside the scope of this undergraduate course.

The reaction load, R, is broken into components, R_s and R_t, for the shear and tension reaction loads on the fasteners. The free-body diagram below shows the applied load and the reaction load components for this problem.



[3]:
```python
# Calculating reaction loads
P = FS*N_limit*g*M
R = P
Rs = R*np.cos(np.radians(theta))
Rt = R*np.sin(np.radians(theta))
print(f"Applied Load, P = {P:.0f} N")
print(f"Shear Reaction, Rs = {Rs:.0f} N")
print(f"Tension Reaction, Rt = {Rt:.0f} N")
```

```
Applied Load, P = 80344 N
Shear Reaction, Rs = 77606 N
Tension Reaction, Rt = 20795 N
```

[4]:
```python
# create a pandas dataframe with the bolt information
bolt_types = ['A', 'B', 'C', 'D', 'E', 'F']
weight = [200, 200, 150, 125, 125, 100]
shears = [7000, 8000, 6000, 5000, 6000, 5000]
tensions = [7000, 6000, 4000, 2500, 5000, 3000]
price = [20, 25, 10, 20, 115, 50]
d = {"Type":bolt_types, "Weight":weight, "Shear": shears, "Tension":tensions,␣
 ↪"Price":price}
fastener_df = pd.DataFrame(data=d)
fastener_df
```

[4]:

	Type	Weight	Shear	Tension	Price
0	A	200	7000	7000	20
1	B	200	8000	6000	25
2	C	150	6000	4000	10
3	D	125	5000	2500	20
4	E	125	6000	5000	115
5	F	100	5000	3000	50

From the fastener shear and tension allowables we can calculate the minimum fasteners required to have a positive margin of safety with the interaction of shear and tension using the following interaction equation from Flabel:

$Ratio_t^2 + Ratio_s^2 = 1$

Where the ratio is the applied load divided by the allowable. If we include the number of fasteners, n, we can re-write the equation to solve for n.

$\left(\frac{R_t}{n*Tension}\right)^2 + \left(\frac{R_s}{n*Shear}\right)^2 = 1$

$n = \sqrt{\left(\frac{R_t}{Tension}\right)^2 + \left(\frac{R_s}{Shear}\right)^2}$

Then based on this quantity we calculate the margin of safety (MS), the total weight of all fasteners in grams, and the total cost for all fasteners.

[5]:

```
# calculate the min qty for the load, MS, total weight, and total cost
# qty based on interaction equation Ratio_shear^2 + Ratio_tension^2 = 1
fastener_df['Min Qty'] = np.ceil(np.sqrt((Rt/fastener_df['Tension'])**2 +
                                 (Rs/fastener_df['Shear'])**2))
n = fastener_df['Min Qty'].to_numpy()
shear = fastener_df['Shear'].to_numpy()
tension = fastener_df['Tension'].to_numpy()
ratio_shear = Rs/(n*shear)
ratio_tension = Rt/(n*tension)
fastener_df['MS'] = 1/np.sqrt(ratio_shear**2 + ratio_tension**2) - 1
fastener_df['MS'] = fastener_df['MS'].round(decimals=2)

fastener_df['Total Weight'] = fastener_df['Weight']*fastener_df['Min Qty']
fastener_df['Total Cost'] = (fastener_df['Total Weight']/28/
 ↪16)*fastener_df['Price']
fastener_df['Total Cost'] = fastener_df['Total Cost'].round(decimals=2)
fastener_df
```

[5]:

	Type	Weight	Shear	Tension	Price	Min Qty	MS	Total Weight	Total Cost
0	A	200	7000	7000	20	12.0	0.05	2400.0	107.14
1	B	200	8000	6000	25	11.0	0.07	2200.0	122.77
2	C	150	6000	4000	10	14.0	0.00	2100.0	46.88
3	D	125	5000	2500	20	18.0	0.02	2250.0	100.45
4	E	125	6000	5000	115	14.0	0.03	1750.0	449.22
5	F	100	5000	3000	50	17.0	0.00	1700.0	189.73

I recommend choosing type F as it minimizes weight, even with an additional fastener to avoid such a low MS. The same would apply to the next lightest option too, so F is still the lightest total weight.

Weight is the critical factor for launching something into orbit, and for every additional pound it costs \$10,000 - \$20,000. So even a 50 gram savings comes to about to at least \$1,000 in savings.

[6]:

```
n = 18
shear = 5000
tension = 3000
ratio_shear = Rs/(n*shear)
ratio_tension = Rt/(n*tension)
MS = 1/np.sqrt(ratio_shear**2 + ratio_tension**2) - 1
print(f"MS for {n} type F fasteners = +{MS:.2f}")
```

```
MS for 18 type F fasteners = +0.06
```