Slade Brooks

M13801712

brooksl@mail.uc.edu

AEEM5058 HW#8

Problem 1

The fuselage of a large commercial aircraft consists of a skin reinforced by a system of stringers and frames as shown in Fig. 1. Analysis of load distribution shows that a section of the fuselage of dimension 1800×840 mm^2 (the longest side is parallel to the fuselage axis) is subject to a compressive load of 100 kN. The skin is made of aluminium alloy 2024-T3 and has uniform thickness t =1.6 mm. Propose a distribution of equally spaced stringers and frames that would prevent buckling of the structure whilst limiting its weight. Provide your solutions in terms of required number of stringers and frames including the distances x and y shown in Fig. 1 . Use the following assumptions:

a) The panel can be considered as flat and simply supported along its edges(this is justified by the large radius of a typical fuselage compared to the width of the panel)
b) The stringers are also made of aluminium alloy 2024-T3
c) The properties of 2024-T3 are those available at https://asm.matweb.com/search/SpecificMaterial.asp?bassnum=MA2024T3
d) The warping constant for a T section is given according to Fig. 2
e) The frames can be considered as rigid and each weighs 2.2 kg

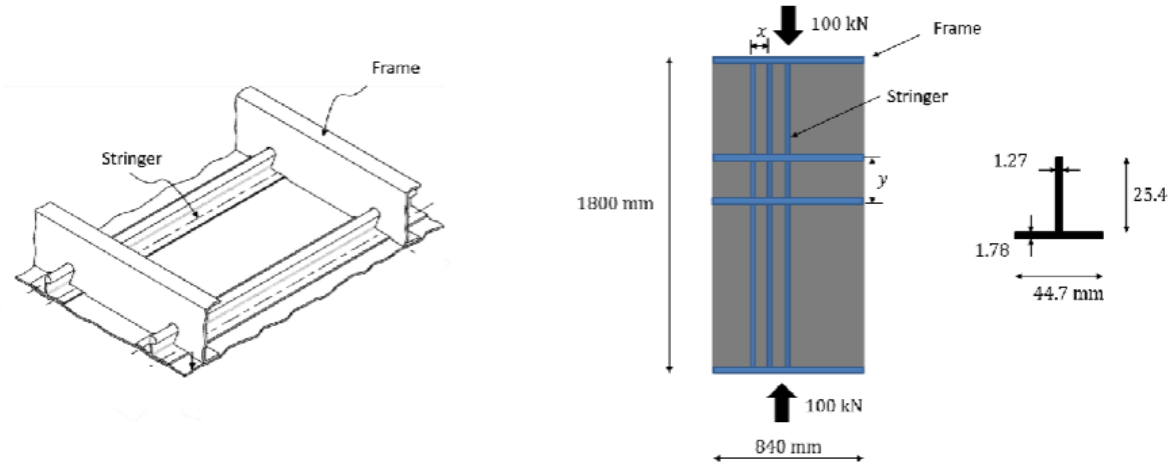
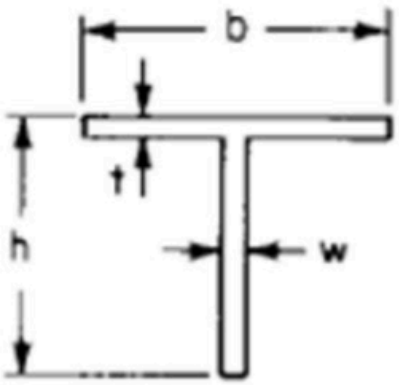


Figure 1



$$I_w = \frac{b^3 t^3}{144} + \frac{h^3 w^3}{36}$$

Figure 2

In [366…
```
import numpy as np
```

First define known values:

In [367…
```
# stringer dimensions
b = 44.7        # stringer b (mm)
t = 1.78        # stringer thick (mm)
h = t + 25.4    # stringer height (mm)
w = 1.27        # stringer width (mm)
ts = 1.6        # skin thickness (mm)
L = 1800.       # section length (mm)
W = 840.        # section width (mm)
F = 100.        # applied load (kN)

# material properties
E = 73.1        # mod of elast (GPa)
rho = 2.78/1e6  # density (kg/m^3)
nu = 0.33       # Poisson's ratio
```

Create a function to determine the moment of area and the cross sectional area for a variable stringer configuration:

In [368…
```
def stringer(x):
    """
    Parameters
    ----------
    x : float
        Spacing of stringers (mm).

    Returns
    -------
    Iy : float
        Second moment of area (mm^4).
    A : float
        X-sec area (mm^2).
    """
    # determine x-sec area
    As = ts*x               # skin area (mm^2)
    Avert = (h - t)*w       # vert. area (mm^2)
```

```
    Ahor = b*t              # horiz. area (mm^2)
    A = As + Avert + Ahor   # total area (mm^2)

    # determine centroid
    yc = 0.                                                          # centroid
    zc = ((Ahor*(h - (t/2))) + (Avert*(h - t)/2) + (As*(h + (ts/2))))/A     # centroid

    # calculate moment of inertia
    Iys = ts**3*x/12 + As*(h + ts/2 - zc)**2                # I of skin (mm^4)
    Iyst = t**3*b/12 + b*t*(h - t + t/2 - zc)**2 + \
        (h - t)**3*w/12 + (h - t)*w*((h - t)/2 - zc)**2     # I of stringer (mm^4)
    Iy = Iyst + Iys                                         # total I (mm^4)

    return Iy, A, As
```

Now set up and loop through numbers of stringers and frames to find the optimal config. Similar to Hw#7, check lowest critical stress to determine if it buckled and if not then stop.

In [369…

```
# number of strings/frames to loop through
nstrings = np.arange(1, 20.1, 1)
nframes = np.arange(2, 10.1, 1)
wmin = np.inf
nso = 0.
nfo = 0.
xo = 0.
yo = 0.

for i, ns in enumerate(nstrings):
    for k, nf in enumerate(nframes):
        x = W/ns              # stringer spacing (mm)
        y = L/(nf - 1)        # frame spacing (mm)

        # determine Iy and A with function
        Iy, A, As = stringer(x)

        # 1-D column approx
        sigma1 = E*Iy*np.pi**2/(A*y**2)

        # individual plates approx
        k = 4.
        sigma2 = k*np.pi**2*E/(12*(1 - nu**2))*(ts/x)**2

        # individual stiff plates approx
        k = 0.43
        sigma3 = k*np.pi**2*E/(12*(1 - nu**2))*(w/(h - t))**2

        # critical stress is minimum
        sigcr = np.min([sigma1, sigma2, sigma3])

        # equivalent skin thickness
        tbar = ((A - As)/x) + ts

        # determine stress
        Nx = F/W
        sigmaF = Nx/tbar

        # determine total weight
        wf = 2.2
        ws = (A - As)*L*rho
```

```
        wsk = ts*L*W*rho
        wtot = nf*wf + ns*ws + wsk

        # make sure that it works and then store if minimum
        if sigmaF <= sigcr:
            if wtot < wmin:
                wmin = wtot
                nso = ns
                nfo = nf
                xo = x
                yo = y
```

Now just print the final answers.

In [370…

```
print(f"Optimal Setup:")
print(f"# stringers: {nso:.0f}, # frames: {nfo:.0f}")
print(f"stringer spacing: {xo:.1f} mm, frame spacing: {yo:.1f} mm")
print(f"mass: {wmin:.1f} kg")
```

```
Optimal Setup:
# stringers: 7, # frames: 4
stringer spacing: 120.0 mm, frame spacing: 600.0 mm
mass: 19.4 kg
```