project2

November 18, 2022

[1]:
```
import math
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import pyplot as plt
```

1 DATA

as you know we have below equations:

$\frac{\overline{C_p}}{R_u} = a + bT + cT^2 + dT^3 + eT^4$

$\frac{\overline{h}}{R_u T} = a + \frac{b}{2}T + \frac{c}{3}T^2 + \frac{d}{4}T^3 + \frac{e}{5}T^4$

$\frac{\overline{S}^0}{R_u} = a\ln T + bT + \frac{c}{2}T^2 + \frac{d}{3}T^3 + \frac{e}{4}T^4$

1.1 Air (dry)

For Air We have:

[2]:
```
a = 3.653
b = -1.334*10**(-3)
c = 3.291*10**(-6)
d = -1.91*10**(-9)
e = 0.275*10**(-12)
```

[3]:
```
R_u = 8.314             #kj/kmol-k
R = R_u/28.9647         #kj/kg-k
P_ref = 1               #bar
T_ref = 298.15          #k
```

[4]:
```
TIT = 1000              #k
etta_is_c = 0.84
etta_is_t = 0.86
```

2 Part 1

[5]:
```
def properties(data):

    #Scenario1: when P and T are given:

    if ('T' in data) & ('P' in data):
        T = data['T']
        P = data['P']
        C_p_bar = R_u*(a + b*T + c*T**2 + d*T**3 + e*T**4)
        h_bar = R_u*T*(a + b/2*T + c/3*T**2 + d/4*T**3 + e/5*T**4)
        s_bar_above0 = R_u*(a*math.log(T) + b*T + c/2*T**2 + d/3*T**3 + e/
↪4*T**4)
        C_p = R/R_u*C_p_bar
        h = R/R_u*h_bar
        s_above0 = R/R_u*s_bar_above0
        s = -R*math.log(P/P_ref)+s_above0
        return {'P[bar]':P, 'T[k]':T, 'C_p[kj/kg-k]':C_p, 'h[kj/kg]':h, 's[kj/
↪kg-k]':s}

    #Scenario2: when P and s are given:

    if ('P' in data) & ('s' in data):
        P = data['P']
        T = np.array(range(300, 1000))
        s_bar_above0 = R_u*(a*np.log(T) + b*T + c/2*T**2 + d/3*T**3 + e/4*T**4)
        s_above0 = R/R_u*s_bar_above0
        difference_of_s = np.abs(s_above0-R*math.log(P/P_ref)-data['s'])
        T = np.linspace(T[difference_of_s.argmin()]-3, T[difference_of_s.
↪argmin()]+3, 600)
        s_bar_above0 = R_u*(a*np.log(T) + b*T + c/2*T**2 + d/3*T**3 + e/4*T**4)
        s_above0 = R/R_u*s_bar_above0
        difference_of_s = np.abs(s_above0-R*math.log(P/P_ref)-data['s'])
        T_answer = T[difference_of_s.argmin()]
        dic = properties({'P':P , 'T':T_answer})
        C_p, h = dic['C_p[kj/kg-k]'], dic['h[kj/kg]']
        return {'P[bar]':P, 'T[k]':T_answer, 'C_p[kj/kg-k]':C_p, 'h[kj/kg]':h,␣
↪'s[kj/kg-k]':data['s']}

    #Scenario3: When P and h are given:

    if ('P' in data) & ('h' in data):
        P = data['P']
        T = np.array(range(300, 1000))
        h_bar = R_u*T*(a + b/2*T + c/3*T**2 + d/4*T**3 + e/5*T**4)
        h = R/R_u*h_bar
        difference_of_h = np.abs(h-data['h'])
```

```python
        T = np.linspace(T[difference_of_h.argmin()]-3, T[difference_of_h.
↪argmin()]+3, 600)
        h_bar = R_u*T*(a + b/2*T + c/3*T**2 + d/4*T**3 + e/5*T**4)
        h = R/R_u*h_bar
        difference_of_h = np.abs(h-data['h'])
        T_answer = T[difference_of_h.argmin()]
        dic = properties({'P':P , 'T':T_answer})
        C_p, s = dic['C_p[kj/kg-k]'], dic['s[kj/kg-k]']
        return {'P[bar]':P, 'T[k]':T_answer, 'C_p[kj/kg-k]':C_p, 'h[kj/kg]':
↪data['h'], 's[kj/kg-k]':s}

    return {}
```

[6]:
```python
point1 = {'P':1 , 'T':300}  #P in bar and T in k
```

[7]:
```python
Properties1 = properties(point1)
```

[8]:
```python
Properties1
```

[8]:
```
{'P[bar]': 1,
 'T[k]': 300,
 'C_p[kj/kg-k]': 1.0045354971741467,
 'h[kj/kg]': 304.76511887228247,
 's[kj/kg-k]': 5.903583614724041}
```

[9]:
```python
point2 = {'P':1 , 'T':500} #P in bar and T in k
```

[10]:
```python
Properties2 = properties(point2)
```

[11]:
```python
Properties2
```

[11]:
```
{'P[bar]': 1,
 'T[k]': 500,
 'C_p[kj/kg-k]': 1.0296628956971763,
 'h[kj/kg]': 507.70032142573547,
 's[kj/kg-k]': 6.421365514096027}
```

[12]:
```python
delta_h = Properties2['h[kj/kg]']-Properties1['h[kj/kg]']
```

[13]:
```python
delta_s = Properties2['s[kj/kg-k]']-Properties1['s[kj/kg-k]']
```

[14]:
```python
print(f'delta_h is equal to >>> {delta_h:.5} kj/kg and delta_s is equal to >>>␣
↪{delta_s:.5} kj/kg-k')
```

```
delta_h is equal to >>> 202.94 kj/kg and delta_s is equal to >>> 0.51778 kj/kg-k
```

3 Part 2

3.1 Point 1

at the inlet of the compressor we have P01 and T01:

[15]:
```
P01 = 1       #bar
T01 = 300     #k
```

[16]:
```
point_01 ={'P':P01,'T':T01}
```

[17]:
```
Properties_01 = properties(point_01)
```

[18]:
```
Properties_01
```

[18]:
```
{'P[bar]': 1,
 'T[k]': 300,
 'C_p[kj/kg-k]': 1.0045354971741467,
 'h[kj/kg]': 304.76511887228247,
 's[kj/kg-k]': 5.903583614724041}
```

3.2 Point 2

The Compressor Compression Ratio is given and is equal to:

[19]:
```
r_c = 12
```

So We can find P02:

[20]:
```
P02 = r_c*P01
```

for this Point We can act like this:

[21]:
```
s02_s = Properties_01['s[kj/kg-k]']
```

[22]:
```
point_02s = {'P':P02, 's':s02_s}
```

[23]:
```
Properties_02s = properties(point_02s)
```

[24]:
```
Properties_02s
```

[24]:
```
{'P[bar]': 12,
 'T[k]': 603.4056761268781,
 'C_p[kj/kg-k]': 1.0514629718103226,
 'h[kj/kg]': 615.2669601538392,
 's[kj/kg-k]': 5.903583614724041}
```

from the equation of isentropic efficiency we can find h_{0_2} and by that we are able to find the properties of the flow at the point 2 (outlet of the compressor):

[25]:
```python
h02s =Properties_02s['h[kj/kg]']
h01 = Properties_01['h[kj/kg]']
```

[26]:
```python
h02 = h01+(h02s-h01)/etta_is_c
```

Now we can act like this:

[27]:
```python
point_02 = {'P':P02, 'h':h02}
```

[28]:
```python
Properties_02 = properties(point_02)
```

[29]:
```python
Properties_02
```

[29]:
```
{'P[bar]': 12,
 'T[k]': 659.3055091819699,
 'C_p[kj/kg-k]': 1.0645147997300364,
 'h[kj/kg]': 674.4101680169929,
 's[kj/kg-k]': 5.997303946481227}
```

3.3 Point 3

Because We don't have any loss of pressure in the combustion chamber, We will have:

[30]:
```python
P03 = P02
```

and also we have the Turbine Inlet Temperature(TIT):

[31]:
```python
T03 = TIT
```

[32]:
```python
point_03 = {'P':P03, 'T':T03}
```

[33]:
```python
Properties_03 = properties(point_03)
```

[34]:
```python
Properties_03
```

[34]:
```
{'P[bar]': 12,
 'T[k]': 1000,
 'C_p[kj/kg-k]': 1.140980227656423,
 'h[kj/kg]': 1050.706446122349,
 's[kj/kg-k]': 6.456285190272438}
```

3.4 Point 4

We have total pressure at the outlet of the turbine:

[35]:
```python
P04 = 1.5    #bar
```

[36]:
```python
s04_s = Properties_03['s[kj/kg-k]']
```

[37]:
```
point_04s = {'P':P04, 's':s04_s}
```

[38]:
```
Properties_04s = properties(point_04s)
```

[39]:
```
Properties_04s
```

[39]:
```
{'P[bar]': 1.5,
 'T[k]': 578.5141903171954,
 'C_p[kj/kg-k]': 1.045879890344605,
 'h[kj/kg]': 589.1643250564591,
 's[kj/kg-k]': 6.456285190272438}
```

from the equation of isentropic efficiency we can find h_{04} and by that we are able to find the properties of the flow at the point 4 (outlet of the turbine):

[40]:
```
h03 = Properties_03['h[kj/kg]']
h04s = Properties_04s['h[kj/kg]']
```

[41]:
```
h04 = h03-etta_is_t*(h03-h04s)
```

[42]:
```
point_04 = {'P':P04, 'h':h04}
```

[43]:
```
Properties_04 = properties(point_04)
```

[44]:
```
Properties_04
```

[44]:
```
{'P[bar]': 1.5,
 'T[k]': 639.8848080133556,
 'C_p[kj/kg-k]': 1.0599150808497755,
 'h[kj/kg]': 653.7802220056838,
 's[kj/kg-k]': 6.562426354210015}
```

3.5 total efficiency

[45]:
```
W_net_is = (h03-h04s)-(h02s-h01)
```

[46]:
```
W_net_ac = (h03-h04)-(h02-h01)
```

[47]:
```
etta_total = W_net_ac/W_net_is
```

[48]:
```
print(f"""
total efficiency is equal to >>> {etta_total*100:.4}%
W_net_is is equal to         >>> {W_net_is:.5} kj/kg
W_net_ac is equal to         >>> {W_net_ac:.5} kj/kg
"""
     )
```

```
total efficiency is equal to >>> 18.06%
W_net_is is equal to         >>> 151.04 kj/kg
W_net_ac is equal to         >>> 27.281 kj/kg
```

3.6 Properties

[49]:
```python
Properties = {
    'Point 1' : Properties_01,
    'Point 2s': Properties_02s,
    'Point 2' : Properties_02,
    'Point 3' : Properties_03,
    'Point 4s': Properties_04s,
    'Point 4' : Properties_04
}
```

[50]:
```python
df = pd.DataFrame(Properties).T
```

[51]:
```python
df
```

[51]:

	P[bar]	T[k]	C_p[kj/kg-k]	h[kj/kg]	s[kj/kg-k]
Point 1	1.0	300.000000	1.004535	304.765119	5.903584
Point 2s	12.0	603.405676	1.051463	615.266960	5.903584
Point 2	12.0	659.305509	1.064515	674.410168	5.997304
Point 3	12.0	1000.000000	1.140980	1050.706446	6.456285
Point 4s	1.5	578.514190	1.045880	589.164325	6.456285
Point 4	1.5	639.884808	1.059915	653.780222	6.562426