Water Layer Present

April 23, 2021

1 Case 2: Water Layer

- Author: **Team G15**
- Attempt: 3

1.1 Analysis

1.1.1 To find

1. Temperature of Water Surface (T_w)
2. Total heat flux entering the house through the roof, (q_t) when a water layer is present

1.1.2 Nomenclature

- S = Intensity of Solar Radiation (i.e. solar constant)
- v_w = water velocity
- v_a = wind velocity
- ϵ_w = emissivity of water surface
- σ = Stefan-Boltzmann constant ($5.67 * 10^{-8}\ W/m^2K^4$)
- T_r = room temperature (inside)
- T_w = water surface temperature (outside)
- T_a = ambient air temperature (outside)
- $\overline{T}_w$ = average water surface temperature (outside)
- $\overline{T}_a$ = average air temperature (outside)
- τ_w = fraction of solar radiation absorbed by water
- k_w = thermal conductivity of water
- L_w = length of water layer
- h_w = convection coefficient of water layer
- h_r = radiative heat transfer coefficient
- h_c = convective heat transfer coefficient
- h_e = evaporative heat transfer coefficient

1.1.3 Assumptions

1. Steady state with room maintained at fixed ambient temperature
2. Water is still ($v_w = 0$) but gentle breeze is present ($v_a = 10\ km/h$)
3. Dry Surroundings

1.1.4 Equations

Energy balance,

$$q_t = q_c + q_r - q_e$$

Radiation heat transfer,

$$q_r = \tau_w \cdot S - h_r \cdot (T_a - T_w)$$

$$h_r = \epsilon_w \cdot \sigma \cdot \frac{(\overline{T}_w)^4 - (\overline{T}_a - 12)^4}{\overline{T}_a - \overline{T}_w}$$

Convection heat transfer,

$$q_c = h_c \cdot (T_a - T_w)$$

$$h_c = 5.678 \cdot (1 + 0.85 \cdot (v_a - v_w))$$

Evaporative heat transfer,

$$q_e = 0.013 \cdot h_c \cdot (p(\overline{T}_w) - \gamma \cdot p(\overline{T}_a))$$

$$p(T) = R_1 \cdot T + R_2$$

Total heat transfer,

$$q_t = \frac{T_w - T_r}{R_{net}}$$

$$R_{net} = \frac{1}{h_r} + \sum_{i=1}^{3} \frac{L_i}{k_i} + \frac{1}{h_w}$$

$$h_w = \frac{k_w}{L_w} \cdot (0.14 \cdot (Gr \cdot Pr)^{1/3} + 0.644 \cdot (Pr \cdot Re)^{1/3})$$

$$Gr = \frac{g \cdot \beta \cdot (T_w - T_a) \cdot (L_w)^3}{\nu^2}$$

1.1.5 Properties

Outside Air

- Mild breeze $v_a = 2.78\ m/s$
- $T_a \in [305, 320]K$
- $T_f = 320K$
- $\beta = \frac{1}{T_f} = 0.0031\ K^{-1}$

- Table A.4, air (T_f):
  - $\nu = 18 \cdot 10^{-6}\ m^2/s$
  - $\alpha = 25 \cdot 10^{-6}\ m^2/s$
  - $Pr = 0.702$
  - $k = 27.7 \cdot 10^{-3}\ W/m \cdot K$
- $S = 1366\ W/m^2$
- $R_1 = 325\ Pa/°C$ and $R_2 = -5155\ Pa$ (*from reference* **#1**)
- $\gamma = 0.27$ (approx average over a day)

Water layer

- $L_w = 0.1\ m$ (approx thickness of water layer)
- Table A.6, water (T_w):
  - $\nu = 18 \cdot 10^{-6}\ m^2/s$
- Still water $v_w = 0$
- $\epsilon_w = 0.95$
- $\tau_w = 0.6$

Roof

- $t = 0.2\ m$ thick with,
  - Cement $= 5\ cm$
  - Brick $= 10\ cm$
  - Lime $= 5\ cm$
- K_i, Conductivity of each layer,
  - Cement $= 0.72\ W/m \cdot K$
  - Brick $= 0.71\ W/m \cdot K$
  - Lime $= 0.73\ W/m \cdot K$

Inside air

- $T_r = 300K$ (Room Temperature)
- $h_r = 8.4\ W/m^2 \cdot K$

1.1.6 Tools used

- **Python**
- **SymPy** for creating symbolic equations and solving them
- **NumPy**
- **Matplotlib** for plotting results

1.2 Solving (Python Code)

1.2.1 Initialize Values

[1]:
```
import sympy as sp
import numpy as np
```

Outside Air

- Saturation pressure of water p = R_1*T + R_2

[2]:
```
v_a = 2.78 # Velocity (m / s)

# Temperatures
T_f = 320 # (K)
beta = 1/T_f # (K)
T_a = np.array([305.0, 310.0, 315.0, 320.0]) # (K)
T_a_avg = 273 + 37 # (K)

# Constants
sigma = 5.67e-8 # Stefan Boltzmann constant (W / m^2 * K^4)
g = 9.8 # (m^2 / s)
R_1 = 325 # N / m^2 °C
R_2 = -5155 # N / m^2
gamma = 0.27
S = 1366 # Solar constant

def p(T): # Saturation pressure of water as a function of temperature (N / m^2)
    return R_1 * (T-273) + R_2
```

Water Layer

[3]:
```
v_w = 0 # Velocity (m / s)
L_w = 5 # Dimensions (m)

# Temperatures
T_w = sp.symbols('T_w') # (K)
T_w_avg = 273 + 32 # (K)

# Constants
epsilon_w = 0.95 # Emissivity of water surface
tau_w = 0.6 # Water's solar absorbtivity
```

- Table A.6 used (*from reference* **#2**)
- Upon analysing the below data, we can approximate h_w to 950 W/m^2

[4]:
```
rho_w = 990 # density (kg / m^3)
k_w = 0.63 # thermal conductivity (W / m * K)
mu_w = 1e-6 * np.array([769, 695, 631, 577]) # viscosity (N * s / m^2)
nu_w = mu_w / rho_w # dynamic visosity (m^2 / s)

Pr_w = np.array([5.20, 4.62, 4.16, 3.77]) # Prandtl number
Re_w = 0 # Reynolds number, still water
Gr_w = g * beta * (T_a - T_w) * L_w**3 / nu_w**2 # Grashof number

# Water free convection coeffecient
```

```
h_w = (k_w/L_w) * (0.14 * (Gr_w*Pr_w)**(1/3) + 0.644 * (Pr_w*Re_w)**(1/3))

# Example at T_a = 310K and T_w = 306K
h_w_test = h_w[1].replace(T_w, 306)
print('Approximate min value of h_w = %.2f' % (h_w_test))
```

```
Approximate min value of h_w = 923.62
```

Roof Layers

[5]:
```
# Layer 1: Concrete
k_1 = 0.72 # (W / m * K)
L_1 = 0.05 # (m)

# Layer 2: Brick
k_2 = 0.71 # (W / m * K)
L_2 = 0.10 # (m)

# Layer 3: Lime
k_3 = 0.73 # (W / m * K)
L_3 = 0.05 # (m)
```

Inside Air

[6]:
```
h_r = 8.4 # (W / m^2 * K)
T_r = 300 # (K)
```

1.2.2 Equations

Radiation Heat

[7]:
```
h_r = epsilon_w * sigma * (T_w_avg**4 - (T_a_avg - 12)**4)/(T_a_avg - T_w_avg)␣
 ↪# (W / m^2 * K)
q_r = tau_w * S - h_r * (T_a - T_w) # (W / m^2)

# Example at T_a = 310K and T_w = 306K
q_r_test = q_r[1].replace(T_w, 306)
print('Approximate value of q_r = %.2f' % (q_r_test))
```

```
Approximate value of q_r = 786.53
```

Convection Heat

- Forced convection and free convection both have been used

[8]:
```
h_c = 5.678 * (1 + 0.85 * (v_a - v_w))
print('h_c = %.2f' % (h_c))

q_c = h_c * (T_a - T_w) # (W / m^2)

# Example at T_a = 310K and T_w = 306K
```

```
q_c_test = q_c[1].replace(T_w, 306)
print('Approximate value of q_c = %.2f' % (q_c_test))
```

```
h_c = 19.10
Approximate value of q_c = 76.38
```

Evaporation Heat:

```
[9]: q_e = 0.013 * h_c * (p(T_w_avg) - gamma * p(T_a_avg)) # function p defined
     ↪above, (W / m^2)

     # Example at T_a = 310K and T_w = 306K
     print('Approximate value of q_e = %.2f' % (q_e))
```

```
Approximate value of q_e = 841.55
```

Total Heat:

```
[10]: h_w = 1200 # from above approximation (W / m^2 * K)
      R = 1/h_r + L_1/k_1 + L_2/k_2 + L_3/k_3 + 1/h_w # (m^2 * K / W)

      q_t = (T_w - T_r) / R # (W / m^2)

      # Example at T_a = 310K and T_w = 306K
      q_t_test = q_t.replace(T_w, 306)
      print('Approximate value of q_t = %.2f' % (q_t_test))
```

```
Approximate value of q_t = 14.98
```

1.2.3 Solving

$$q_c + q_r - q_e = q_t$$
$$\therefore\ q_c + q_r - q_e - q_t = 0$$

Calculate T_w

```
[11]: eq = q_c + q_r - q_e - q_t

      n = len(eq)
      T_w_calc = np.empty(n, dtype=object)

      for i in range(n):
          T_w_calc[i] = round(sp.solve(eq[i], T_w)[0], 2)

      for i in range(n):
          print('T_w = %.1f K for T_a = %.1f K' % (T_w_calc[i], T_a[i]))
```

```
T_w = 302.4 K for T_a = 305.0 K
T_w = 306.5 K for T_a = 310.0 K
T_w = 310.5 K for T_a = 315.0 K
T_w = 314.6 K for T_a = 320.0 K
```

Calculate q_t

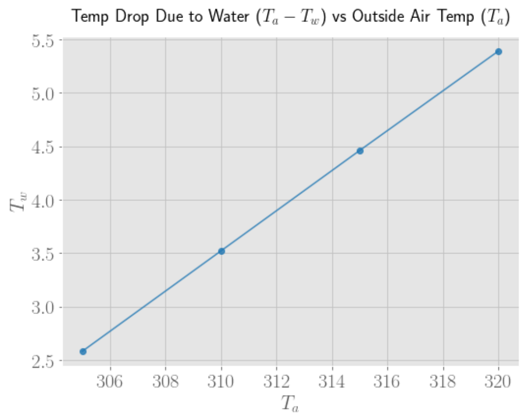
```
[12]: q_t_calc = np.empty(n, dtype=object)

for i in range(n):
    q_t_calc[i] = q_t.replace(T_w, T_w_calc[i])

for i in range(n):
    print('Heat entering = %.1f W/m^2 for T_a = %.1f K' % (q_t_calc[i], T_a[i]))
```

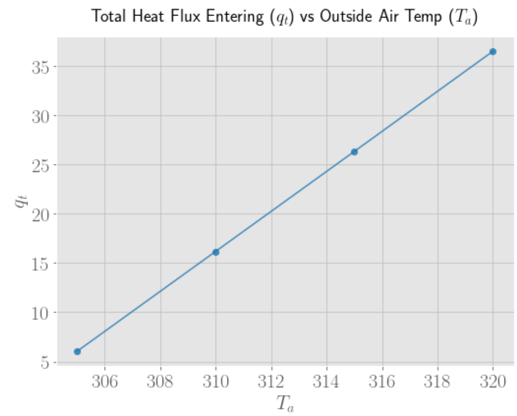
```
Heat entering = 6.0 W/m^2 for T_a = 305.0 K
Heat entering = 16.2 W/m^2 for T_a = 310.0 K
Heat entering = 26.3 W/m^2 for T_a = 315.0 K
Heat entering = 36.5 W/m^2 for T_a = 320.0 K
```

1.2.4 Plot

- Temp Drop Due to Water $(T_a - T_w)$ vs Outside Air Temp (T_a)
- Total Heat Flux Entering (q_t) vs Outside Air Temp (T_a)

```
[13]: %matplotlib inline
import matplotlib.pyplot as plt

# Initialize matplotlib
plt.rc('text', usetex=True) # Unnecessary
plt.style.use('ggplot')
plt.rcParams['grid.color'] = '#C0C0C0'

fig = plt.figure(figsize=(16, 6))

ax1 = fig.add_subplot(121)
plt.plot(T_a, T_a-T_w_calc, color='#1F77B4cc', marker='o')
plt.xticks(fontsize=20)
plt.yticks(fontsize=20)
plt.xlabel('$T_a$', fontsize=20)
plt.ylabel('$T_w$', fontsize=20)
plt.title('Temp Drop Due to Water ($T_a - T_w$) vs Outside Air Temp ($T_a$)',␣
 ↪fontsize=18, pad=15)

ax2 = fig.add_subplot(122)
plt.plot(T_a, q_t_calc, color='#1F77B4cc', marker='o')
plt.xticks(fontsize=20)
plt.yticks(fontsize=20)
plt.xlabel('$T_a$', fontsize=20)
plt.ylabel('$q_t$', fontsize=20)
plt.title('Total Heat Flux Entering ($q_t$) vs Outside Air Temp ($T_a$)',␣
 ↪fontsize=18, pad=15)

fig.tight_layout(w_pad=10)
```





1.3 References

1. A. Shrivastava *et al.* "Evaporative cooling model..." (1984)
2. F. Incropera *et al.* "Fundamentals of Heat and Mass Transfer."