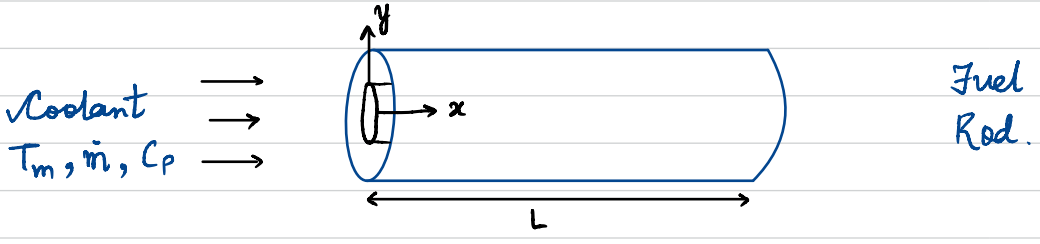
ME341: Assignment 3

→ Abhinav Maheshwari
→ 190028

Q. The volumetric generation rate is given by:

$$q^*(x) = q_0^* \left(\frac{x}{L}\right)^2$$



a) Consider an element with thickness dx:

⇒ Curved surface area: $dA = (\pi D)\, dx$

⇒ Vol. of the element: $dV = \frac{\pi D^2}{4} dx$

By applying energy conservation:

⇒ $E_{in} - E_{out} + E_g = 0$

⇒ $-dq + E_g = 0$

- dq - net heat transfer in the element
- E_g - energy generated

$$\therefore -\left\{q'' \times (\pi D\, dx)\right\} + \left\{q_0^* \left(\frac{x}{L}\right)^2 \left(\frac{\pi D^2}{4} \times dx\right)\right\} = 0.$$

$$\Rightarrow \quad \boxed{q'' = q_0^* \left(\frac{x}{L}\right)^2 \left(\frac{D}{4}\right)} \quad \text{Ans.}$$

∴ Net heat transfer from the rod:

$$q = \int_0^L q'' \, (\pi D) \, dx$$

$$\Rightarrow \quad q = \int_0^L q_0^* \left(\frac{D}{4}\right) \cdot \frac{x^2}{L^2} \cdot \pi D \cdot dx$$

$$\Rightarrow \quad q = q_0^* \frac{\pi D^2}{4L^2} \int_0^L x^2 dx = q_0^* \frac{\pi D^2}{4L^2} \left[\frac{x^3}{3}\right]_0^L$$

$$\Rightarrow \quad \boxed{q = q_0^* \frac{\pi D^2 L}{12}} \quad \text{Ans.}$$

b) Applying energy conservation for the coolant:

$$\Rightarrow \quad dq + \dot{m} C_p T_m = \dot{m} C_p (T_m + dT_m)$$

$$\Rightarrow \quad dq = \dot{m} C_p \, dT_m$$

$$\therefore \quad q'' \times (\pi D \, dx) = \dot{m} C_p \, dT_m$$

$$\Rightarrow \frac{dT_m}{dx} = q'' \times \frac{\pi D}{\dot{m} C_p} = q_0^* \left(\frac{x}{L}\right)^2 \left(\frac{D}{4}\right) \times \frac{\pi D}{\dot{m} C_p}$$

$$\therefore \frac{dT_m}{dx} = q_0^* \frac{\pi D^2}{4 \dot{m} C_p} \frac{x^2}{L^2}$$

$$\Rightarrow \int_{T_{m_i}}^{T_m(x)} dT_m = q_0^* \frac{\pi D^2}{4 \dot{m} C_p L^2} \int_0^L x^2 \, dx$$

$$\Rightarrow T_m(x) - T_{m_i} = q_0^* \frac{\pi D^2}{4 \dot{m} C_p L^2} \left(\frac{L^3}{3}\right)$$

$$\Rightarrow \boxed{T_m(x) = T_{m_i} + q_0^* \frac{\pi D^2 L}{12 \dot{m} C_p}} \quad \text{Ans.}$$

c) Newton's Law of Cooling : $q'' = h (T_s - T_m)$

$$\Rightarrow T_s = T_m + \frac{q''}{h}$$

$$\therefore T_s = T_m + \frac{1}{h} \left\{ q_0^* \left(\frac{x}{L}\right)^2 \left(\frac{D}{4}\right) \right\}$$

$$\Rightarrow \boxed{T_s = T_{m_i} + q_0^* \left(\frac{\pi D^2 L}{12 \dot{m} C_p}\right) + q_0^* \left(\frac{D}{4L^2 h}\right) x^2} \quad \text{Ans.}$$

1. We need the relation between u/U_∞ and η using the Range-Kutta method, the root finding method and the Bisection method.

$$\Rightarrow \boxed{\eta = \frac{y}{\delta} = \frac{y}{\sqrt{\frac{\nu x}{U_\infty}}}}$$

From the similarity law for Blassius eq^n :

$$\frac{u}{U_\infty} = F\left[\frac{y}{\delta}\right] = F\left[\frac{y}{\sqrt{\frac{\nu x}{U_\infty}}}\right] = F(\eta)$$

Governing eq^n : $\boxed{2f'''(\eta) + f''(\eta) f'(\eta) = 0}$

where $f'(\eta) = F(\eta)$

$$\Rightarrow \frac{u}{U_\infty} = f'(\eta)$$

- Converting this into 3 first order diff. eq^{n}s:

$$\boxed{f' = G \;;\quad G' = H \;;\quad H' = -\frac{1}{2} fH}$$

Boundary Conditions : $f(0) = 0$

$f'(0) = G(0) = 0$

$f'(\infty) = G(\infty) = 1$

→ We will assume an initial value of H
→ Then we will solve using the Range - Kutta Method to get $G(\infty)$ or $G(L) = 1$ for large L value.
→ On every subsequent step, we will update the value of $H(0)$ using the root-finding method and bisection method.

Implementation of the Range - Kutta method can be found in the MATLAB code given below :

(We take old values of $H(0)$, arr_f, arr_g and arr_h ; update them using $H(0)$ and then return new values of $G(L)$, arr_f, arr_g, arr_h)

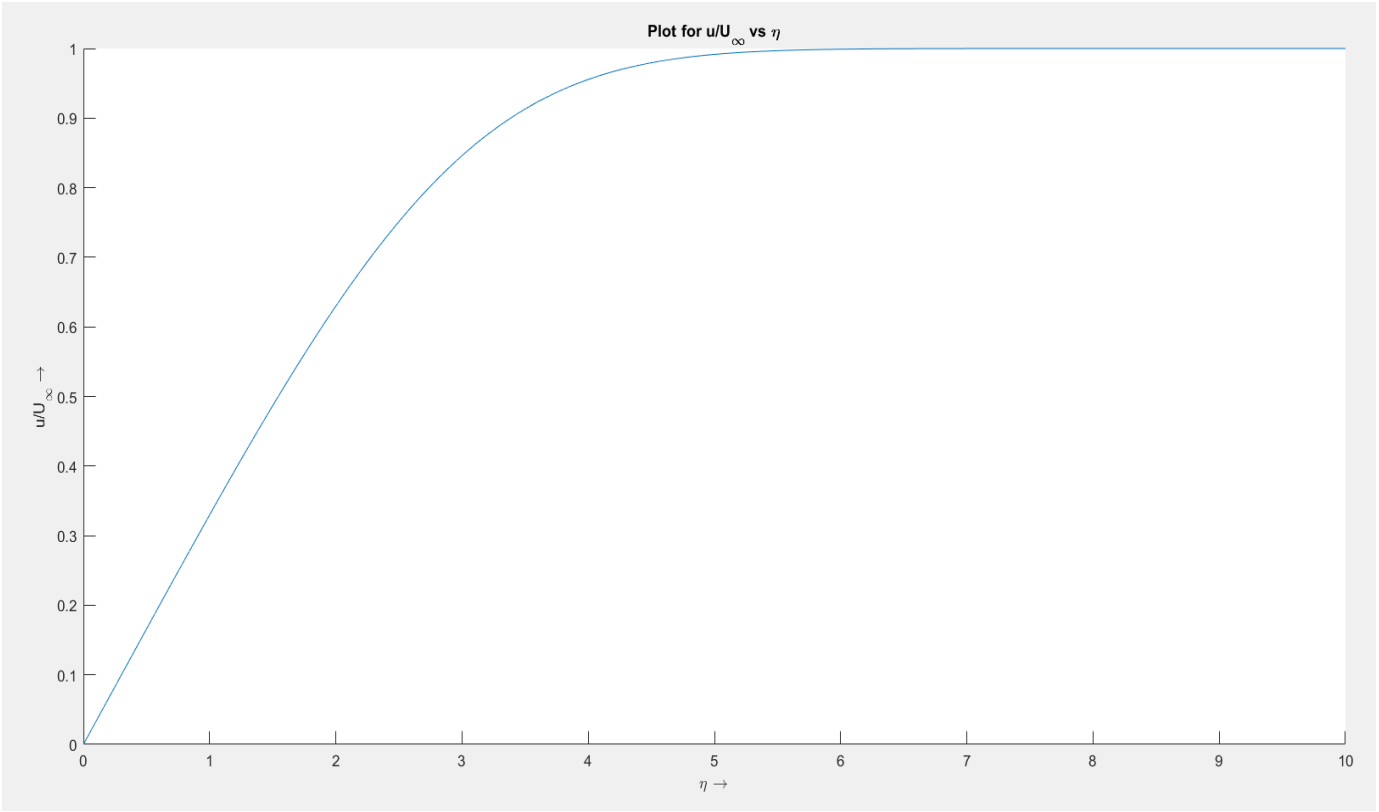
The plots can be found on the last page.

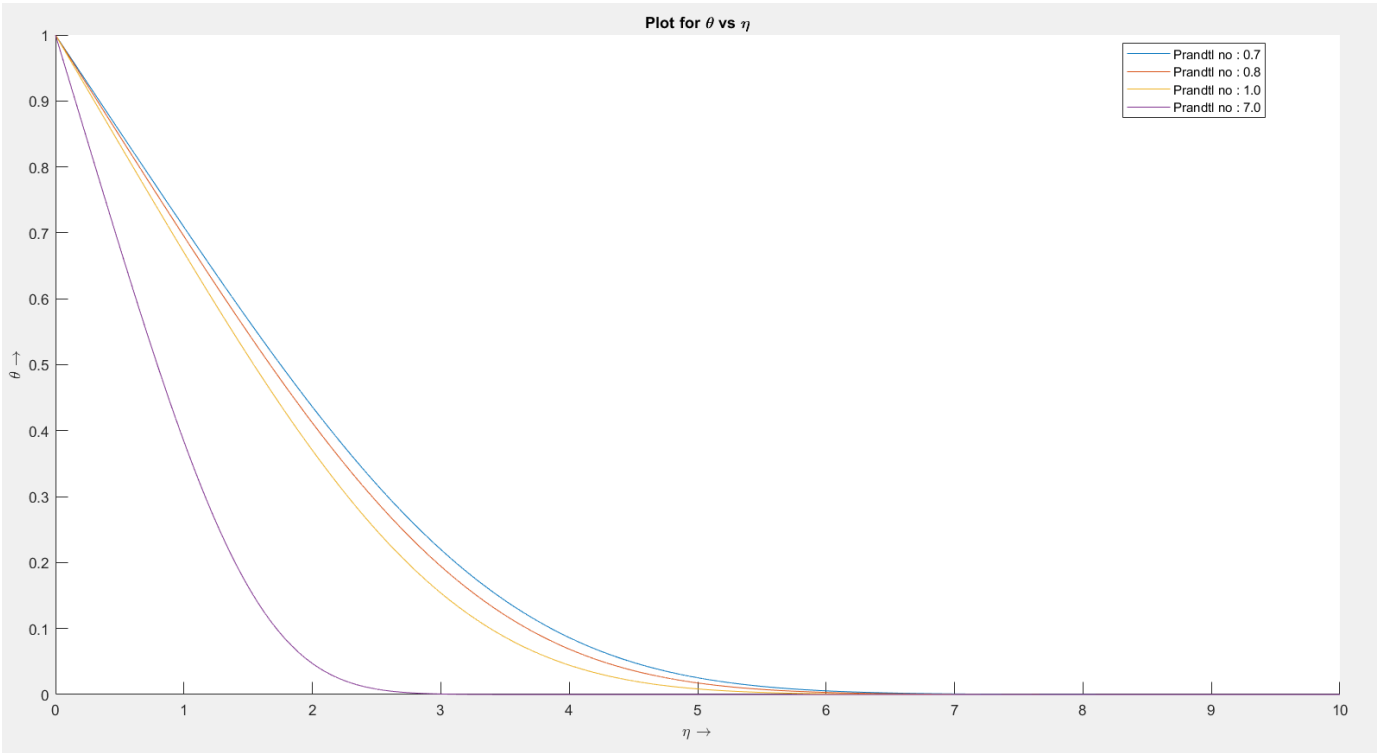
```
%% Abhinav Maheshwari
%% 190028
%% ME341 Assignment 3 Q1

L = 10.0 ; %% length of the domain
N = 100000 ; %% number of grid points
A = 0.0 ; %% guessed value of H at eta = 0
B = 1.0 ; %% guessed value of H at eta = 0
convergence = 0.00001 ; %% criteria for convergence
h = L /(N - 1) ; %% size of each step
arr_f = zeros(1, N) ;
arr_g = zeros(1, N) ;
arr_h = zeros(1, N) ;
Theta = zeros(1, N) ;
Theta_Des = zeros(1, N) ;


iterations = 0 ;
c_old = 1.0 ;

a = A ; %% guessed values of H, mentioned above
b = B ; %% guessed values of H, mentioned above

%% bisection method
err = 1 ;
eta = linspace(0, L, N) ;

figure(1)
title("Plot for u/U_{\infty} vs \eta")
xlabel("\eta \rightarrow")
ylabel("u/U_{\infty} \rightarrow")
hold on

figure(2)
title("Plot for \theta vs \eta")
xlabel("\eta \rightarrow")
ylabel("\theta \rightarrow")
hold on

while(err > convergence)

    iterations = iterations + 1 ;
    c_new = (a + b) / 2.0 ; %% new guessed value

        %% solving using shooting velocity technique
    [f_a, arr_f, arr_g, arr_h] = shoot_vel(a, arr_f, arr_g, arr_h, h, N) ;
    [f_b, arr_f, arr_g, arr_h] = shoot_vel(b, arr_f, arr_g, arr_h, h, N) ;
    [f_c, arr_f, arr_g, arr_h] = shoot_vel(c_new, arr_f, arr_g, arr_h, h, N) ;

    if(((f_a - 1.0)*(f_c - 1.0)) > 0) %% checking location of root
        a = c_new ;
    else
        b = c_new ;
    end

    err = abs((c_new - c_old) / c_old) ; %% error calculation
    c_old = c_new ; %% updating the guessed value
end

figure(1)
plot(eta, arr_g)
```

```
for Pr = [0.7 0.8 1 7]
    a = A ; %% guessed value of Theta_Des(0), mentioned above
    b = B;
    c_old = 1.0 ;
    err = 1 ;
    Theta = zeros(1, N) ;
    Theta_Des = zeros(1, N) ;
    while(err > convergence)
        iterations = iterations + 1;
        c_new = (a + b) / 2.0 ; %% new guessed value

                %% solve using shoooting thermal technique
        [f_a, Theta, Theta_Des]= shoot_therm(arr_f, a, Theta, Theta_Des, Pr, h, N) ;
        [f_c, Theta, Theta_Des]= shoot_therm(arr_f, c_new, Theta, Theta_Des, Pr, h, N) ;
        [f_b, Theta, Theta_Des]= shoot_therm(arr_f, b, Theta, Theta_Des, Pr, h, N) ;

        if((f_a - 1.0)*(f_c - 1.0) > 0) %% checking location of root
            a = c_new ;
        else
            b = c_new ;
        end
        err = abs((c_new - c_old) / c_old) ; %% error calculation
        c_old = c_new ; %% updating the guessed value
    end

    figure(2)
    plot(eta, 1 - Theta) %% for theta = (T - T_inf) / (T_w - T_inf) = 1 - (T_w - T) / (T_w - T_inf)
    hold on
end



%% Define Functions

%% shoots using guessed value of h(0), returns g(L) value
```

```
function [g_new, arr_f, arr_g, arr_h] = shoot_vel(h_old, arr_f, arr_g, arr_h, h, N)
    f_old = 0 ; %% f(0) = 0
    g_old = 0 ; %% g(0) = 0
    for i = 1:N %% calculating the coefficients of Runge-Kutta method
        k1 = h*g_old ;
        l1 = h*h_old ;
        m1 = h*(-0.5)*f_old*h_old ;

        k2 = h*(g_old + 0.5*l1) ;
        l2 = h*(h_old + 0.5*m1) ;
        m2 = h*(-0.5)*(f_old + 0.5*k1)*(h_old + 0.5*m1) ;

        k3 = h*(g_old + 0.5*l2) ;
        l3 = h*(h_old + 0.5*m2) ;
        m3 = h*(-0.5)*(f_old + 0.5*k2)*(h_old + 0.5*m2) ;

        k4 = h*(g_old + l3) ;
        l4 = h*(h_old + m3) ;
        m4 = h*(-0.5)*(f_old + k3)*(h_old + m3) ;

        %% values of f,g,h for the next position
        f_new = f_old + (1.0 / 6.0)*(k1 + 2.0*k2 + 2.0*k3 + k4) ;
        g_new = g_old + (1.0 / 6.0)*(l1 + 2.0*l2 + 2.0*l3 + l4) ;
        h_new = h_old + (1.0 / 6.0)*(m1 + 2.0*m2 + 2.0*m3 + m4) ;

        %% updating the values of f,g,h
        f_old = f_new ;
        g_old = g_new ;
        h_old = h_new ;
```

```
        %% updating the values of f,g,h in the arrays
        arr_f(i) = f_new ;
        arr_g(i) = g_new ;
        arr_h(i) = h_new ;
    end
end


function [Y1_new, Theta, Theta_Des] = shoot_therm(arr_f, Y_in, Theta, Theta_Des, Pr, h, N)
    Y_old = Y_in ; %% guessed value for Theta_Des(0)
    Y1_old = 0.0 ; %% Theta(0) = 0

    for i = 1:N  %% calculating the  coefficients of Runge-Kutta method
        k1 = (-0.5)*Pr*arr_f(i)*Y_old ;
        k2 = (-0.5)*Pr*arr_f(i)*(Y_old + (h / 2.0)*k1) ;
        k3 = (-0.5)*Pr*arr_f(i)*(Y_old + (h / 2.0)*k2) ;
        k4 = (-0.5)*Pr*arr_f(i)*(Y_old + h*k3) ;

        l1 = Y_old ;
        l2 = Y_old + (h / 2.0)*l1 ;
        l3 = Y_old + (h / 2.0)*l2 ;
        l4 = Y_old + h*l3 ;

        %% values of Theta, Theta_Des for the next position
        Y_new = Y_old + (h / 6.0)*(k1 + 2.0*k2 + 2.0*k3 + k4) ;
        Y1_new = Y1_old + (h / 6.0)*(l1 + 2.0*l2 + 2.0*l3 + l4) ;

        %% updating the values Theta, Theta_Des
        Y_old = Y_new ;
        Y1_old = Y1_new ;

        %%update Theta, Theta_Des in array
        Theta(i) = Y1_new ;
        Theta_Des(i) = Y_new ;
    end
end
```


Plot for u/U∞ vs η
u/U∞ →
η →
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
1
2
3
4
5
6
7
8
9
10


Plot for θ vs η
θ →
η →
Prandtl no : 0.7
Prandtl no : 0.8
Prandtl no : 1.0
Prandtl no : 7.0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
1
2
3
4
5
6
7
8
9
10