Brad Rafferty

Introduction

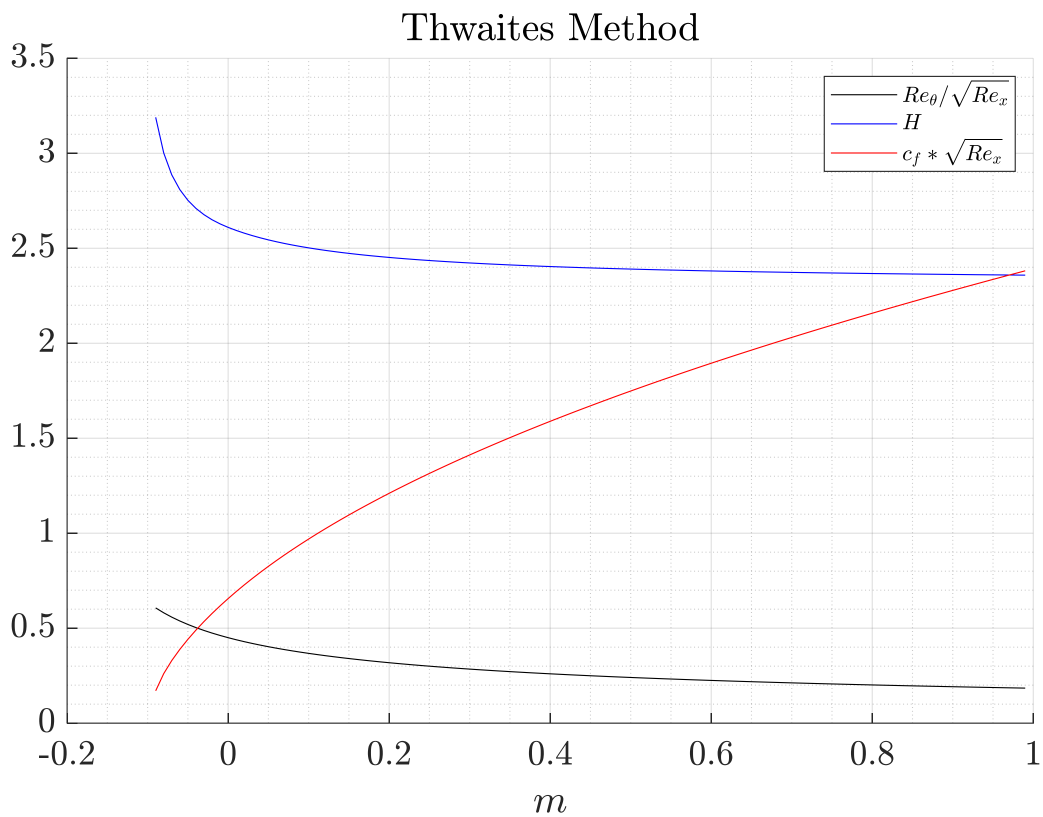
This script serves as an example of how to implement boundary layer analyses using two methods: *Thwaites method* (for integrating the Von Karman equation) and solving the *Falkner-Skan Equations*. The results are then compared for a variety of different nondimensionalized flow parameters

Nomenclature

- `theta` - boundary layer momentum thickness
- `Re` - Reynolds number
- `cf` - skin friction coefficient
- `H` - shape factor (displacement thickness / momentum thickness)
- `m` - dimensionless constant

Calculations from Thwaites' Governing Equations

```
clearvars

% % Declare array of m values over which to loop
m_array = (-0.09:0.01:0.99)';

% % Calculate Re_theta / sqrt(Re_x)
Re_theta_div_sqrt_Re_x = (0.45 ./ (5*m_array + 1).^0.5);

for i = 1:length(m_array)
    m = m_array(i);

    % % Calculate H(lambda) and l(lambda)
    lambda = 0.45*m / (5*m + 1);
    if 0 <= lambda && lambda <= 0.1
        H(i,1) = 2.61 - 3.75*lambda + 5.24*lambda^2;
        l(i,1) = 0.22 + 1.57*lambda - 1.80*lambda^2;
    elseif -0.1 <= lambda && lambda < 0
        H(i,1) = 2.088 + 0.0731/(lambda + 0.14);
        l(i,1) = 0.220 + 1.4020*lambda + 0.018*lambda/(0.107 + lambda);
    else
        warning('lambda not within -0.1 and 0.1!')
    end

end

% % Calculate cf*sqrt(Re_x)
cf_sqrt_Re_x = 2*l ./ (0.45 ./ (5*m_array + 1)).^0.5;

% % Plots
plot_results(m_array, Re_theta_div_sqrt_Re_x, H, cf_sqrt_Re_x)
title('Thwaites Method')
saveas(gcf, 'fig1.emf')
```

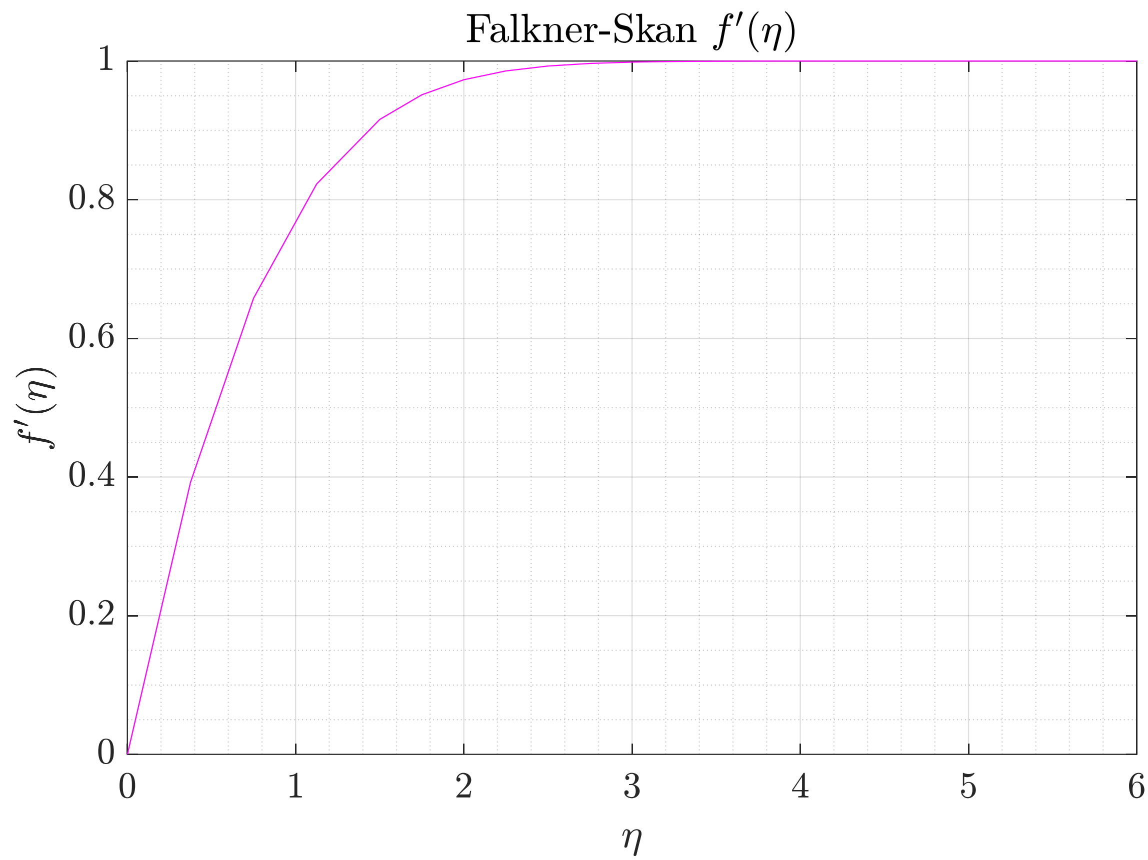


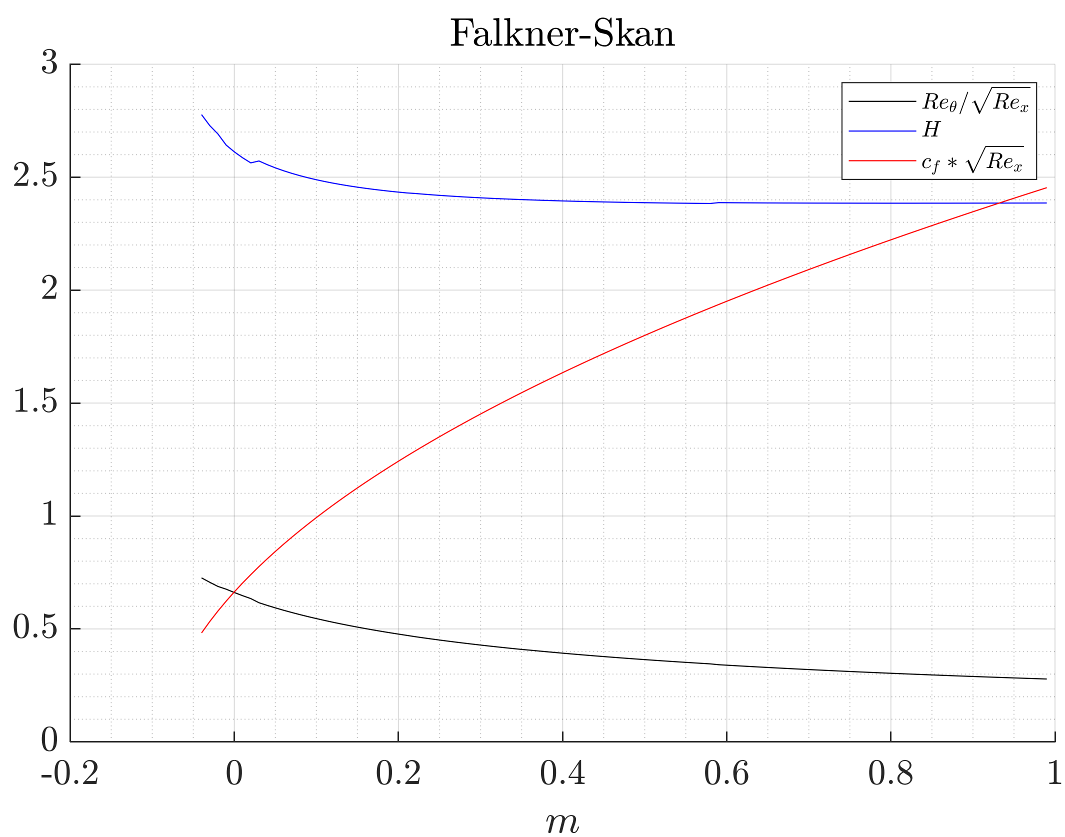
Falkner-Skan Calculations

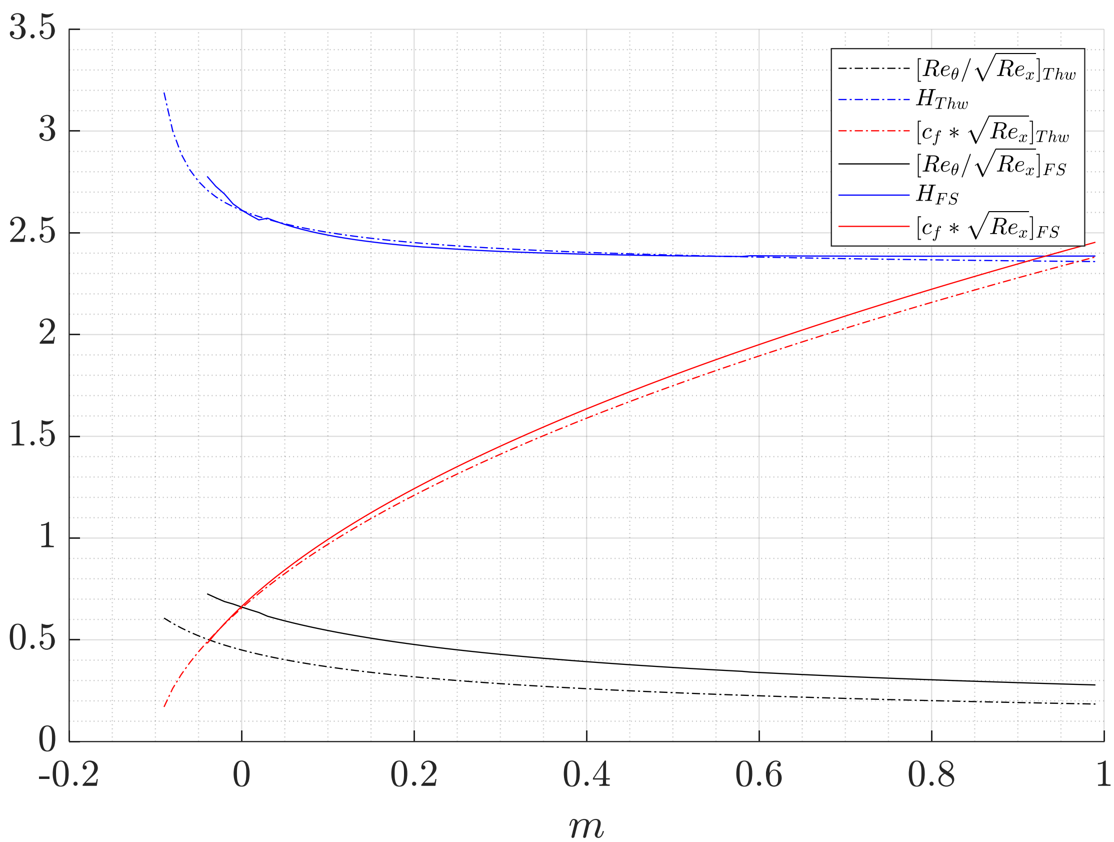
```
% % Use a large-enough value for eta limit to prevent numerical problems at too high of values
eta_limit   = 6;

% % Declare m values to loop through
m_array_2     = (-0.04:0.01:0.99);

% % Initialize the Falkner-Skan solution (may take some trial and error)
% and set the options for the boundary value problem solver
solinit = bvpinit(linspace(0, eta_limit, 5), [.005, .005, .005]);
options = bvpset('stats', 'off'); % Set options for bvp4c

% % Loop through the array of m values
for i = 1:length(m_array_2)
    m = m_array_2(i);

    % % Solve the boundary value problem
    sol = bvp4c(@fsode, @fsbc, solinit, options, m);

    % % Contents of sol:
    % |sol.x| - mesh selected by bvp4c
    % |sol.y| - approximation to y(x) at the mesh points of sol.x

    % % Pull values from the sol structure
    eta = sol.x;
```

```
    f   = sol.y(1, :); % f
    fp  = sol.y(2, :); % f'
    fpp = sol.y(3, :); % f''

    % % Assign integrand for definition of theta and calculate the integral
    int_term = fp.*(1-fp);
    theta(i) = 1/sqrt((m+1)/2) * trapz(eta, int_term);

    % % Assign integrand for definition of del_star and calculate the integral
    int_term    = (1-fp); % Integrand for delta_star definition
    del_star(i) = 1/sqrt((m+1)/2) * trapz(eta, int_term);

    % % Calculate shape factor
    H_2(i) = del_star(i)/theta(i);

    Re_theta_div_sqrt_Re_x_2(i) = theta(i);

    cf_sqrt_Re_x_2(i) = 2*sqrt((m+1)/2)*fpp(eta==0);
end


% % Plots
% f'(eta) vs eta - confirm it ranges from 0 to 1
figure('Position', [100 100 750 500])
    plot(eta, fp, 'm')
    xlabel('$$\eta$$')
    ylabel('$$f''(\eta)$$')
    set(gca, 'FontSize', 16)
    set(gcf, 'Color', 'w')
    grid on; grid minor;
    title('Falkner-Skan $f''(\eta)$')
    saveas(gcf, 'fig2.emf')

plot_results(m_array_2, Re_theta_div_sqrt_Re_x_2, H_2, cf_sqrt_Re_x_2)
title('Falkner-Skan')
saveas(gcf, 'fig3.emf')

% % Add Falkner-Skan solutions to Thwaites' Method plot
figure('Position', [100 100 750 500])
    hold on
    plot(m_array, Re_theta_div_sqrt_Re_x, 'k-.')
    plot(m_array, H, 'b-.')
    plot(m_array, cf_sqrt_Re_x, 'r-.')
    plot(m_array_2, Re_theta_div_sqrt_Re_x_2, 'k-')
    plot(m_array_2, H_2, 'b-')
    plot(m_array_2, cf_sqrt_Re_x_2, 'r-')
    xlabel('$m$')
    set(gca, 'fontsize', 16)
    set(gcf, 'color', 'w')
    grid on; grid minor
    legend({'$$[Re_{\theta} / \sqrt{Re_x}]_{Thw}$$', '$$H_{Thw}$$',...
            '$$[c_f*\sqrt{Re_x}]_{Thw}$$', '$$[Re_{\theta} / \sqrt{Re_x}]_{FS}$$',...
```

```
        '$$H_{FS}$$', '$$[c_f*\sqrt{Re_x}]_{FS}$$'}, 'fontsize', 10, 'location', 'northeast')
    saveas(gcf, 'fig4.emf')
```


Falkner-Skan $f'(\eta)$
$f'(\eta)$
η
0
0.2
0.4
0.6
0.8
1
0
1
2
3
4
5
6


Falkner-Skan
$Re_\theta/\sqrt{Re_x}$
H
$c_f * \sqrt{Re_x}$
m
0
0.5
1
1.5
2
2.5
3
-0.2
0
0.2
0.4
0.6
0.8
1



Functions

```
function [] = plot_results(m_array, Re_theta_div_sqrt_Re_x, H, cf_sqrt_Re_x)
% % Various parameters vs m
figure('Position', [100 100 750 500])
    hold on
    % % Re_theta / sqrt(Re_x) vs m
    plot(m_array, Re_theta_div_sqrt_Re_x, 'k')
    % % H vs m
    plot(m_array, H, 'b')
    % % cf*sqrt(Re_x) vs m
    plot(m_array, cf_sqrt_Re_x, 'r')
    xlabel('$$m$$')
    set(gca, 'FontSize', 16)
    set(gcf, 'Color', 'w')
    grid on; grid minor;
    legend('$$Re_{\theta} / \sqrt{Re_x}$$', '$$H$$', '$$c_f*\sqrt{Re_x}$$', 'fontsize', 10)
end

function dfdeta = fsode(eta, f, m)
%%fsode is the Falkner-Skan ODE

beta = 2*m/(m+1);
dfdeta = [  f(2) % f'
```

```
             f(3) % f''
            -f(1)*f(3) - beta*(1 - f(2)^2) ];
end

function res = fsbc(f0, finf, m)
%%fsbc are the boundary conditions for the Falkner-Skan equations

res = [ f0(1) % f(0) = 0
        f0(2) % f'(0) = 0
        finf(2) - 1 ]; % f'(infinity) - 1 = 0
end
```