ME135 Final Report

Ava Smith, Neha Nepal, and Lucky Babcock-Chi

A Set up

For this project, we decided to use 4 blades with a rotor diameter of 50 meters. The rated wind speed is 12 m/s, and the tip speed ratio is 6. Our group decided on the S8037 Blade within the S8 family. To calculate the Reynolds number at $r = \frac{2}{3}R$, we took the chord to be about R/10 m for an intial guess of the chord.

B Rotor Shape Refinement

$$Re = \frac{cV_0}{\nu} = \frac{(1.23)(2)\left((12)^2 + (2.88 \times \frac{2}{3} \times 25)^2\right)}{1.81 \times 10^{-5}} = 6.724 \times 10^6 \qquad (1)$$

Using the equation above (with a sample calculation), we found that at every distance along the blade that we calculated, the Reynold's number was well above 1000000, therefore we will take the closest approximation for this turbine at a Reynolds number of 1 million at $r = \frac{2}{3}R$. Using optimal rotor theory to get the rotor shape in terms of chord and pitch, we found this tip shape, correcting manually for the region $r < 0.25R$.

The normalized chord distribution can be visualized in the figure below. Since, the Reynolds number did not fall below 1 million, so the chord and pitch were not updated. The x interval began at an x value of 0.2 because the chord begins to exponentially increase when getting too close to 0.

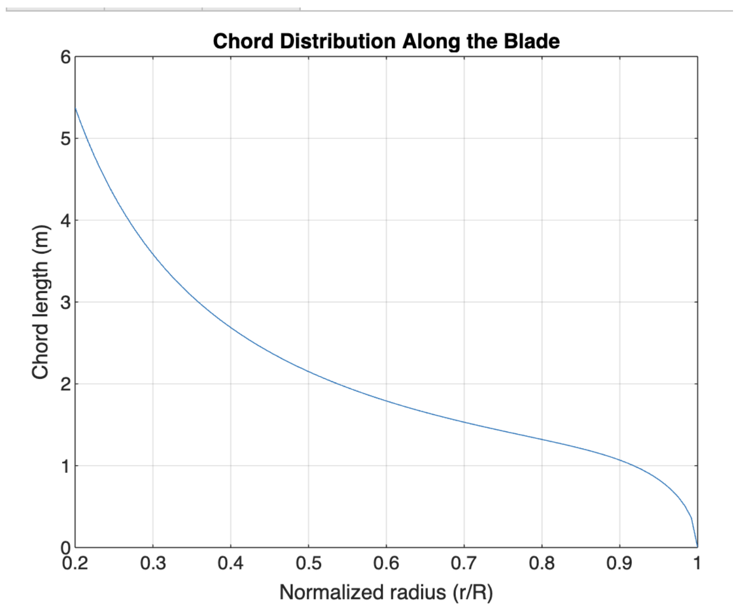


Figure 1: Plot of chord distribution

C Power calculations at rated wind speed

Considering the 12 m/s rated wind speed, power, torque, and thrust coefficients were calculated. To ensure the axial induction factors were correct for the model, iterations over the length of the turbine were done to see where they converged. The value of a was then recorded as 0.33, and the value of a' was about 0.1. The axial induction factor over the normalized radius is included below, as well as torque, power, thrust, and their coefficients.

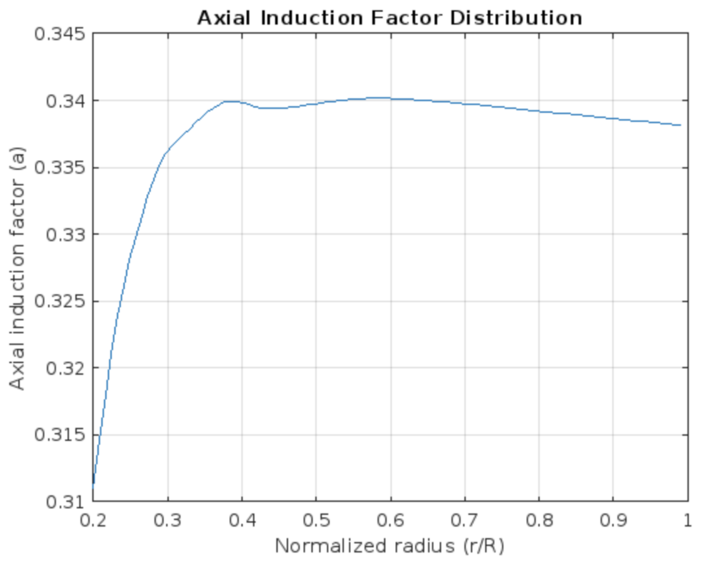


Figure 2: Plot of axial induction factor

Integration was used to carry out the formulas for power, torque, and thrust:

$$P = \frac{1}{2}\rho V_0^3 B R^2 \lambda \int_0^1 \Lambda \left(\frac{w}{V_0}\right)^2 x \left(\sin(\Phi(x)) - (\Phi\cos(x))\frac{C_D}{C_L}\right) dx \tag{2}$$

$$Q = \frac{PR}{\lambda V_0} \tag{3}$$

$$T = \frac{1}{2}\rho V_0^2 A C_t \tag{4}$$

$$C_P = \frac{2P}{\rho V_0^3 A} \tag{5}$$

$$C_q = \frac{C_P}{\lambda} \tag{6}$$

$$C_t = \int_0^1 8a(1-a)Fx\,dx \tag{7}$$

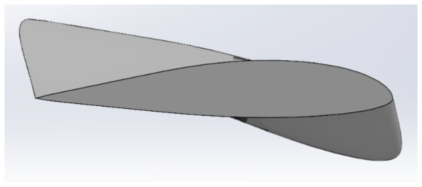
From which we find the following values:

Wind Speed (m/s)	Power (W)	Torque (Nm)	Thrust (N)	C_p	C_t	C_q
12	1,025,792.0	356177.8	139977.8	0.5	0.8	0.1

Table 1: Performance metrics at rated wind speed

D Design

Here is our updated rotor blade design for the S8037 airfoil:


(a) Base View

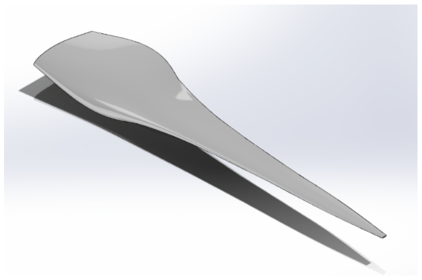

(b) Isometric View

Figure 3: Comparison of Base and Isometric Views

$$\text{Root Bending Moment} = T \times r \quad (\text{approximated at } x = \frac{2}{3}R) \tag{8}$$

From Blade Element Theory:

$$\text{Torque } = dQ = \frac{1}{2} \times \rho \times W^2 \times cdr(C_L \sin\phi - C_D \cos\phi) \tag{9}$$

$$W = \sqrt{V_0^2 + (r\omega)^2}, \quad \omega = \frac{\lambda V_0}{R} = 2.88 \tag{10}$$

$$W = \sqrt{(12)^2 + \left(2.88 \times \frac{2}{3} \times 25\right)^2} \text{ m/s} = 49.48 \text{ m/s} \tag{11}$$

$$\phi = \alpha(\text{angle of attack}) + \theta(\text{pitch}) = 5.5 + 4.07 = 9.57^\circ \tag{12}$$

$$C_L = 0.9019, \quad C_D = 0.00816 \quad (\text{using } C_L\&C_D \text{ values at max } C_L/C_D \text{ value} = 5.5) \tag{13}$$

$$dQ = \frac{1}{2}(1.23 \text{ kg/m}^3)(49.48 \text{ m/s})^2(1.6 \text{ m})\left(\frac{2}{3}(25 \text{ m})\right)(0.9019\sin(9.57){-}0.00816\cos(9.57)) \tag{14}$$

$$dQ = 5.6974 \text{ kN} \tag{15}$$

$$\text{Root Bending Moment} = 5.6974 \text{ kN} \times \left(\frac{2}{3}25 \text{ m}\right) = 94.96 \text{ kNm} \tag{16}$$

To find the wind speed where there is significant deflection in the 10-inch diameter model of the turbine, we will scale our SolidWorks model down to the new dimensions and change the material to ABS plastic. Then, we will run a static study in SolidWorks to find the minimum amount of force applied to a single blade that will result in significant deflection, or displacement at the tip.

Making the assumption that significant deflection occurs when x (displacement) is:

$$x = \frac{1}{20} R_{\text{model}} = 6.35 \text{ mm}$$

where $R_{\text{new}} = 5$ in $= 0.127$ m. The force applied to the model airfoil to achieve this displacement is approximately:

$$F \approx 1.1 \text{ N}$$

Here's the displacement distrubution:

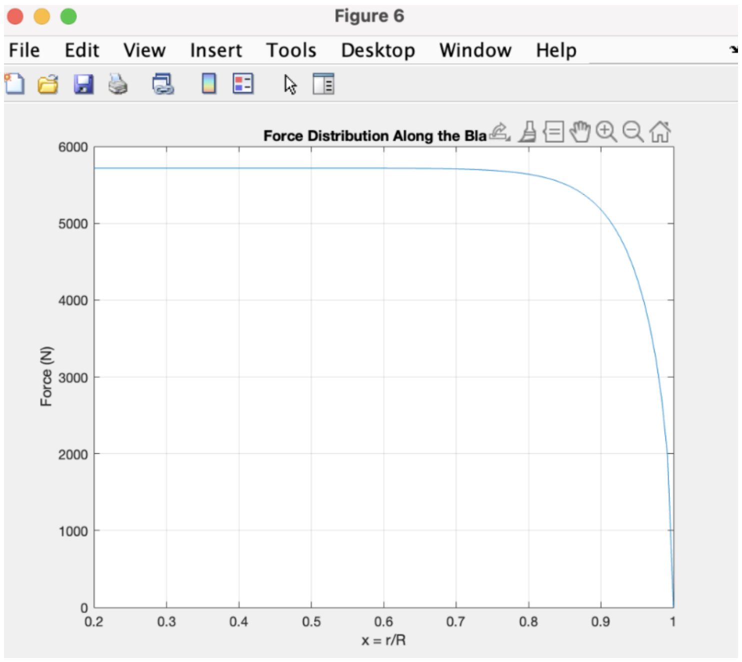


Figure 4: Plot of Force Distribution

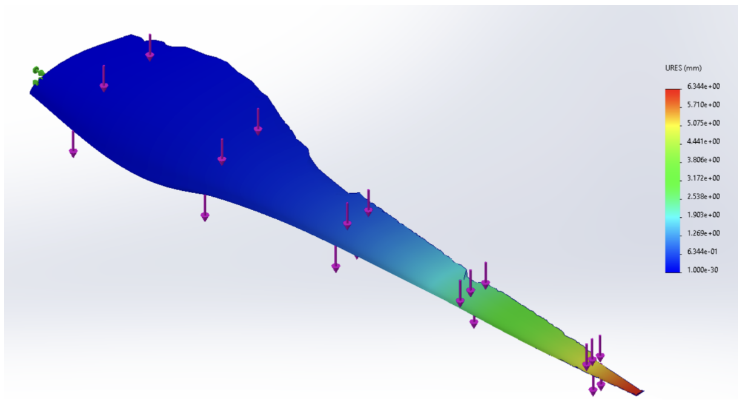


Figure 5: Simulation with Distributed Force

Now, integrating the torque equation from 0 to 1:

$$\int_0^1 dQ = \int_0^1 \frac{1}{2} \times \rho \times W^2 \times c\, dr (C_L \sin\phi - C_D \cos\phi)$$

$$1.1N \cdot R = \int_0^1 0.5(1.225)\sqrt{(}(V_0^2 + (xR)^2))^2 \times c(x) \times R\, dx (C_L \sin\phi - C_D \cos\phi)$$

To reduce this, we can condense the constants and create two separate integrals:

$$C = \frac{(1.1 \cdot 0.127}{0.5(1.225)0.127(0.9019 \sin(9.57°) - 0.00816 \cos(9.57°))}$$

$$C = \int_0^1 V_0^2 c(x)\, dx + \int_0^1 (xR)^2 c(x)\, dx$$

$$V_0 = \sqrt{(}\frac{C - I_1}{I_2})$$

The wind speed V_0 that causes significant deflection is 38.41 m/s.

E Wind Below the Rated Wind Speed

In Part E, it was required to find the Reynolds number on the turbine with different rated wind speeds of 4 m/s and 8 m/s. Building on the code used in earlier parts, new relative velocities were found in order to reassess the Reynolds number.

Every Reynold's number calculation stayed above a million for the wind speeds, so no further interpolation was needed and the data for Re=1E6 was used for all the calculations.

The same formulas were updated and then used for the integration values (our code with the variation from parts B and C for part E are included at the end). From our calculations, the following values were found for each rated wind speed:

Wind Speed (m/s)	Power (W)	Torque (Nm)	Thrust (N)	C_p	C_t	C_q
4	25,036.00	26079.17	10467.18	0.3253	0.5440	0.0542
8	200,288.03	104316.68	41868.72	0.3253	0.5440	0.0542

Table 2: Performance metrics at 4 m/s and 8 m/s wind speeds

F Wind Exceeding the Rated Wind Speed

To find the pitch adjustments at different wind speeds, we needed to recalculate our tip speed ratio. To maintain constant power, we need to the angle ψ necessary to keep theta the same, to keep power the same. From our new value of the λ, we are calculate the able to find a new ϕ. From our new angle, we calculate a new angle of attack from which we are able to find the anlge ψ. The pitch adjustment is found in the table below, at each location, for every wind speed.

Table 3: Pitch adjustment (ψ) required for different wind speeds

	x_s_1	**x_s_2**	**x_s_3**	**x_s_4**	**x_s_5**
Vf_{14}	0.12773	0	-0.042064	-0.085958	-0.10786
Vf_{16}	0.19259	0	-0.0092635	-0.06064	-0.087333
Vf_{18}	0.25107	0	0	-0.035589	-0.066942
Vf_{20}	0.30372	0	0	-0.010831	-0.046708

G Wake Calculations

To find the wind speeds in the wake as a function of downstream distance for rated conditions, we use the following equations.

$$\frac{V_{w,\text{Park}}}{V_0} = 1 - \frac{2a}{\left(1 + \frac{2k_{\text{park}}x}{D}\left(\sqrt{\frac{1-2a}{1-a}}\right)\right)^2} \tag{17}$$

$$\frac{V_{w,\text{RevisedPark}}}{V_0} = \frac{1}{2} + \frac{1}{2}\sqrt{1 - \frac{8a(1-2a)}{\left(\sqrt{\frac{1-a}{1-2a}} + \frac{2kx}{D}\right)^2}} \tag{18}$$

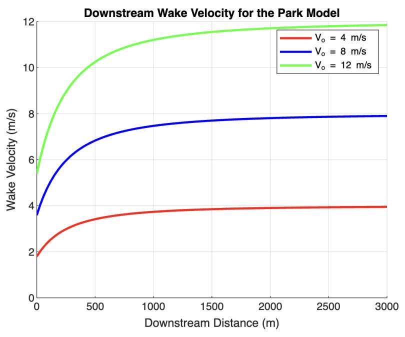
$$x_2 = \left(\frac{1-2a}{2a}\right)^{\frac{2}{3}} \tag{19}$$

$$x_o = x_2 - \frac{D}{6E}\left(\frac{(1-2a)^{\frac{3}{2}}(1-a)^{\frac{1}{2}}}{2a}\right) \tag{20}$$

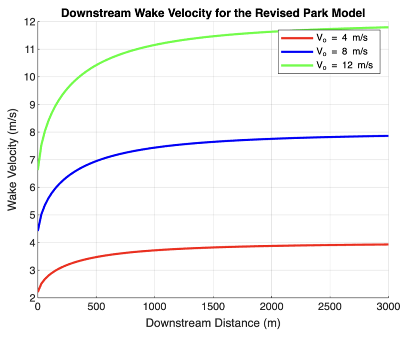
$$X = \frac{6E}{(2a(1-a))^{\frac{1}{2}}}\left(\frac{x - x_o}{D}\right) \tag{21}$$

$$\frac{V_{\text{w,Entrainment}}}{V_0} = \frac{X^{\frac{2}{3}}}{X^{\frac{2}{3}} + 1} \tag{22}$$

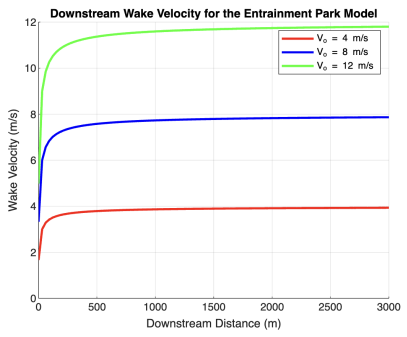
Plotting these we get the following three graphs for the three models:



(a) Park Model



(b) Revised Park Model



(c) Entrainment Model

Figure 6: Park vs Revised Park vs Entrainment Model

From there, we find the power associated with one turbine, from which we find 10 turbines spaced 6 diameters apart, by summing the 10 turbines. We get the following values for the power:

V_0 (m/s)	Park Model (W)	Revised Park Model (W)	Entrainment Model (W)
4	209037.3914	293782.5835	472114.5786
8	1672299.1315	2350260.6682	3776916.6291
12	5644009.5689	7932129.7553	12747093.6232

Table 4: Power output for different wind speeds using the Park, Revised Park, and Entrainment models

H Code

H.1 Part A

```
    clear; clc; close all;
B = 4;lambda = 6;V0 = 12;R = 25; Vrel = sqrt ((V0*lambda/R*(2/3 *R))^2 + V0^2);
```

```
D = 50;R = D/2;g = 9.81;rho = 1.225;mu = 1.81e-5;
% airfoil data at different Reynolds numbers to interpolate later
alpha_max_1_mil = 5.5*pi/180;
alpha_max_500_thou = 6.25*pi/180;
CL_max_1_mil = 0.9019;
CD_max_1_mil = 0.00816;
CL_max_500 = 1.0708;
CD_max_500 = 0.01072;
a_initial = 1/3; %guess
%% Part A
%calculating chord and pitch at specific locations
x_s = [4/R; 8/R; 12/R; 16/R; 20/R]; %these will work for part A and E&F
for i = 1:length(x_s)
    F_s(i) = (2/pi) * acos(exp(-(1-x_s(i))*B*lambda/(2*(1-a_initial))));
    c_s(i) = 16*pi*F_s(i)*R/(9*CL_max_1_mil*B*lambda^2*x_s(i));
    Re_s(i)= (rho * Vrel * lambda * x_s(i) * R * c_s(i)) / mu;
    pitch_s(i) = atan(2/(3*lambda*x_s(i))) - alpha_max_1_mil;
end
```

H.2 Part B

```
%% Part B: General shape of the turbine blade
x = linspace(0.2, 1, 100);
c = zeros(size(x));
pitch = zeros(size(x));
F = zeros(size(x));
Re = zeros(size(x));
for i = 1:length(x)
    F(i) = (2/pi) * acos(exp(-(1-x(i))*B*lambda/(2*(1-a_initial))));
    c_guess = R/10;
    Re_guess = (rho * Vrel * lambda * x(i) * R * c_guess) / mu;
    if Re_guess < 750000
        CL_design = CL_max_500;
        alpha_design = alpha_max_500_thou;
    else
        CL_design = CL_max_1_mil;
        alpha_design = alpha_max_1_mil;
    end
    c(i) = 16*pi*F(i)*R/(9*CL_design*B*lambda^2*x(i));
    % Recalculate Reynolds number with the actual chord
    Re(i) = (rho * Vrel * lambda * x(i) * R * c(i)) / mu;
    pitch(i) = atan(2/(3*lambda*x(i))) - alpha_design;
end
% Plot the chord distribution
figure(1);
plot(x, c);
```

```
xlabel('Normalized radius (r/R)');
ylabel('Chord length (m)');
title('Chord Distribution Along the Blade');
grid on;
figure(2);
plot(x, pitch*180/pi);
xlabel('Normalized radius (r/R)');
ylabel('Pitch angle (degrees)');
title('Pitch Angle Distribution Along the Blade');
grid on;
% Part C: BEM iteration to find axial induction factors
a = ones(size(x)) * a_initial;
a_prime = zeros(size(x));
phi = zeros(size(x));
CL_values = zeros(size(x));
CD_values = zeros(size(x));
sigma = zeros(size(x));
max_iterations = 100;
tolerance = 1e-5;
for i = 1:length(x)
    r = x(i) * R;
    sigma(i) = (B * c(i)) / (2 * pi * r);
    error = 1;
    iterations = 0;
    a_prime(i) = 0.01;
    while error > tolerance && iterations < max_iterations
        phi(i) = atan((1 - a(i)) / ((1 + a_prime(i)) * lambda * x(i)));
        alpha = phi(i) - pitch(i);
        [CL_values(i), CD_values(i)] = airfoilData(alpha);
        a_new = 1 / ((4 * F(i) * sin(phi(i)) * sin(phi(i))) / (sigma(i) * (CL_values(i) * co
        a_prime_new = 1 / ((4 * F(i) * sin(phi(i)) * cos(phi(i))) / (sigma(i) * (CL_values(i
        error = max(abs(a_new - a(i)), abs(a_prime_new - a_prime(i)));
        a(i) = 0.75 * a(i) + 0.25 * a_new;
        a_prime(i) = 0.75 * a_prime(i) + 0.25 * a_prime_new;
        iterations = iterations + 1;
    end
    if iterations >= max_iterations
        fprintf('Warning: Maximum iterations reached at x = %.2f without convergence\n', x(i
    end
end
plot(x,a_prime)
figure(3);
plot(x, a);
xlabel('Normalized radius (r/R)');
ylabel('Axial induction factor (a)');
title('Axial Induction Factor Distribution');
```

```
grid on;
```

H.3 Part C

```
%% Part C
a = 1/3;
a_prime = 0.1;
% CL_max_1_mil = 0.9019;
% CD_max_1_mil = 0.00816;
x = linspace(0.01, 1, 100);
integrand_values = zeros(size(x));
for i = 1:length(x)
    F_val = (2/pi) * acos(exp(-(1-x(i)) * B * lambda / (2 * (1 - a))));
    phi_val = atan((1 - a) / (lambda * x(i) * (1 + a_prime)));
    w_val = V0 * sqrt((lambda * x(i))^2 + (1 - a)^2);
    c_val = 16 * pi * F_val * R / (9 * CL_max_1_mil * B * lambda^2 * x(i));
    L_lambda_val = c_val * CL_max_1_mil / R;
    integrand_values(i) = (w_val / V0)^2 .* L_lambda_val * x(i) * ...
        (sin(phi_val) - cos(phi_val) * CD_max_1_mil / CL_max_1_mil);
end
y = trapz(x, integrand_values);
Cp = B * lambda / pi * y;
Cq = Cp/lambda;
Ct = trapz(x, 8*a*(1-a)*F.*x);
Power = Cp*0.5*rho*V0^3*(pi*R^2);
Torque = Power/(lambda*V0/R);
omega = lambda*V0/R; % to use in part F
Thrust = Ct*0.5*rho*V0^2*(pi*R^2);
print('Results for Part C')
fprintf('Results for Cp = %.1f m/s:\n', Cp);
fprintf('Results for Cq = %.1f m/s:\n', Cq);
fprintf('Results for Ct = %.1f m/s:\n', Ct);
fprintf('Results for Power = %.1f m/s:\n', Power);
fprintf('Results for Torque = %.1f m/s:\n', Torque);
fprintf('Results for Thrust = %.1f m/s:\n', Thrust);
figure;
plot(x, integrand_values);
xlabel('Normalized radius (r/R)');
ylabel('Integrand value');
title('Power Coefficient Integrand Distribution');
grid on;
```

H.4 Part D

```
    % Significant Deflection
omega = 2.88; & Rotational speed (rad/s)|
R_new = 0.127; & Scaled down model radius (in meters)
a_x = @(x) interpl(x_d, a, x, 'linear', 'extrap'); % creating a function a(x)
 = (1.1*0.127)/(0.5*1.225*R_new*(CL_max_1_mil*sind (phi)-CD_max_1_mil*cosd(phi))); % value o
c_d_new = @(x) ((R_new * 16 * pi) ./ (9 * CL_max_1_mil * B * lambda^2 * x)) .* (2/pi) .* acc
% Compute I1 = integral of c(x)|
I1 = integral(c_d_new, 0.2, 1);
% Compute I2 = integral of (omega*x*R)^2 * c(x)
12 = integral (@(x) (omega*x*R_new).^2 |* c_d_new(x), 0.2, 1);
* Display result
printf('The wind speed V_o that causes significant deflection is %.2f m/s\h', V_o);
```

H.5 Part E

```
%% Part E
V0_s = [4; 8];
Cp_results = zeros(size(V0_s));
Cq_results = zeros(size(V0_s));
Ct_results = zeros(size(V0_s));
Power_results = zeros(size(V0_s));
Torque_results = zeros(size(V0_s));
Thrust_results = zeros(size(V0_s));

print('Results for Part E')
for i = 1:length(V0_s)
    V0 = V0_s(i);
    CL = zeros(size(x_s));
    CD = zeros(size(x_s));
    phi_s = zeros(size(x_s));
    w_s = zeros(size(x_s));
    integrand_values_s = zeros(size(x_s));
    for j = 1:length(x_s)
        Vrel_s = sqrt((V0*lambda*x_s(j))^2 + V0^2);
        Re_s = (rho * Vrel_s * c_s(j)) / mu;
        if Re_s < 5E5
            CL(j) = CL_max_500;
            CD(j) = CD_max_500;
        elseif Re_s > 1E6
            CL(j) = CL_max_1_mil;
            CD(j) = CD_max_1_mil;
        else
            CL(j) = interp1([5E5, 1E6], [CL_max_500, CL_max_1_mil], Re_s);
            CD(j) = interp1([5E5, 1E6], [CD_max_500, CD_max_1_mil], Re_s);
        end
        phi_s(j) = atan((1 - a) / (lambda * x_s(j) * (1 + a_prime)));
        w_s(j) = V0 * sqrt((lambda * x_s(j))^2 + (1 - a)^2);
        L_lambda = c_s(j) * CL(j) / R;
        integrand_values_s(j) = (w_s(j) / V0)^2 * L_lambda * x_s(j) * ...
            (sin(phi_s(j)) - cos(phi_s(j)) * CD(j) / CL(j));
        fprintf('Results for Re = %.1f m/s:\n', Re_s);
    end
    y_s = trapz(x_s, integrand_values_s);
    Cp_results(i) = B * lambda / pi * y_s;
    Cq_results(i) = Cp_results(i) / lambda;
    Ct_results(i) = trapz(x_s, 8 * a * (1 - a) * F_s .* x_s');
    A = pi * R^2;
    Power_results(i) = 0.5 * rho * A * V0^3 * Cp_results(i);
    Torque_results(i) = Power_results(i) / (lambda * V0 / R);
    Thrust_results(i) = 0.5 * rho * A * V0^2 * Ct_results(i);
    fprintf('Results for V0 = %.1f m/s:\n', V0);
    fprintf('  Cp = %.4f\n', Cp_results(i));
    fprintf('  Cq = %.4f\n', Cq_results(i));
    fprintf('  Ct = %.4f\n', Ct_results(i));
    fprintf('  Power = %.2f W\n', Power_results(i));
    fprintf('  Torque = %.2f N·m\n', Torque_results(i));
    fprintf('  Thrust = %.2f N\n\n', Thrust_results(i));
end
```

H.6 Part F

```
%% Part F - Adjusting Pitch Angle () for Wind Speeds Above Rated
V_f = [14; 16; 18; 20];
psi = zeros(length(V_f), length(x_s));

for j = 1:length(V_f)
    lambda_s = omega * R / V_f(j);

    for i = 1:length(x_s)
        phi_new = atan((1 - a) / ((1 + a_prime) * lambda_s * x_s(i)));

        alpha_new = phi_new - pitch(i);

        if alpha_new > alpha_max_1_mil
            alpha_new = alpha_max_1_mil;
        elseif alpha_new < -alpha_max_1_mil
            alpha_new = -alpha_max_1_mil;
        end

        theta_new = phi_new - alpha_new;

        psi(j, i) = theta_new - pitch(i);
    end
end
```

```
disp('Pitch adjustment () required for different wind speeds:');
disp(array2table(psi, 'VariableNames', ...
    strcat('x_s_', string(1:length(x_s))), 'RowNames', strcat('Vf_', string(V_f))));
```

H.7 Part G

```
    % MATLAB Code for Wind Turbine Wake Calculation (Part G)

% Parameters
Ct = 0.8;
n_turbines = 10;
D = 50;
spacing = 6 * D;
a = 0.276;
k_park = 0.06;
k_revised = 0.03;
x = linspace(0, 10*6*D, 100);
E = 0.8;
rho = 1.23;

V_o_values = [4, 8, 12];
colors = ['r', 'b', 'g'];

% Park (Jensen) Model
figure;
hold on;
for i = 1:length(V_o_values)
    V_o = V_o_values(i);
    V_park = V_o * (1 - (2 * a) ./ (1 + 2 * k_park * x ./ D * ((1 - 2 * a) / (1 - a))^0.5).
    plot(x, V_park, 'Color', colors(i), 'LineWidth', 2, 'DisplayName', ['V_o = ', num2str(V

    P_turbine = 0.5 * rho * (pi/4) * D^2 * V_o^3 * (V_park/V_o).^3;
    P_total_park(i) = sum(P_turbine(1:n_turbines));

end
hold off;
legend;
xlabel('Downstream Distance (m)');
ylabel('Wake Velocity (m/s)');
title('Downstream Wake Velocity for the Park Model');
grid on;

% Revised Park Model
figure;
hold on;
```

```
for i = 1:length(V_o_values)
    V_o = V_o_values(i);
    V_revised = V_o * (0.5 + 0.5*sqrt(1 - (8*a*(1-a))./((((1 - a) / (1 - 2 * a))^0.5 + 2 * k
    plot(x, V_revised, 'Color', colors(i), 'LineWidth', 2, 'DisplayName', ['V_o = ', num2str

    P_turbine = 0.5 * rho * (pi/4) * D^2 * V_o^3 * (V_revised/V_o).^3;
    P_total_revised(i) = sum(P_turbine(1:n_turbines));
end
hold off;
legend;
xlabel('Downstream Distance (m)');
ylabel('Wake Velocity (m/s)');
title('Downstream Wake Velocity for the Revised Park Model');
grid on;

% Entrainment Model
figure;
hold on;
for i = 1:length(V_o_values)
    V_o = V_o_values(i);
    x2 = ((1-2*a)/(2*a))^(2/3);
    x_o = x2 - (D/(6*E))*(((1-2*a)^(3/2)*(1-a)^(1/2))/(2*a));
    X = ((6*E)/(2*a*(1-a))^(1/2))*((x-x_o)./D);
    V_entrainment = V_o * (X.^(2/3)./(X.^(2/3)+1));
    plot(x, V_entrainment, 'Color', colors(i), 'LineWidth', 2, 'DisplayName', ['V_o = ', nun

    P_turbine = 0.5 * rho * (pi/4) * D^2 * V_o^3 * (V_entrainment/V_o).^3;
    P_total_entrainment(i) = sum(P_turbine(1:n_turbines));
end
hold off;
legend;
xlabel('Downstream Distance (m)');
ylabel('Wake Velocity (m/s)');
title('Downstream Wake Velocity for the Entrainment Park Model');
grid on;

% Display total power output for each model
disp('Total Power Output for each model:');
for i = 1:length(V_o_values)
    disp(['V_o = ', num2str(V_o_values(i)), ' m/s']);
    disp(['  Park Model: ', num2str(P_total_park(i)), ' W']);
    disp(['  Revised Park Model: ', num2str(P_total_revised(i)), ' W']);
    disp(['  Entrainment Model: ', num2str(P_total_entrainment(i)), ' W']);
end
```