ME 236 - Ideal Turbine Blade Shape

Colin Hodge

May 17, 2021

Contents

Nomenclature

Latin

c = Chord length (m)

C_L = Lift Coefficient $(Unitless)$

n = Number of Blades (m)

t = time (s)

S = Length of Disturbed Wind Stream (m)

r = Radial Distance (m)

R = Blade Length (m)

V = Wind Velocity (m/s)

Greek

α = angle of attack $(radians)$

ϕ = Inflow Angle $(radians)$

λ = Tip Speed Ratio $(unitless)$

ω = Angular Velocity (rad/s)

Subscript

opt = Optimum (*unitless*)

w = wind re-establishment (*unitless*)

r = relative (*unitless*)

s = blade rotation into next blade position (*unitless*)

1 Theory

1.1 Assumptions

All assumptions made were based on average wind conditions that could be applied to a standard wind turbine. The angle of attack was chosen from research which led to the article Aerodynamics of Wind Turbine Blades, (cited below, 3). Within this article it states that the angle of attack on turbine blades are usually in between 10 and 15 degrees, this is to create the most ideal amount of lift, while simultaneously reducing the drag. To stay true to this, an angle of attack was picked in the middle at α =12°.

The coefficient of lift was assumed based on the first article seen in the bibliography portion of this report, (cited below 3), titled Fluctuations of angle of attack and lift coefficient and the resultant fatigue loads for a large Horizontal Axis Wind turbine. A graph within this article (Figure 1) relates the coefficient of lift to the angle of attack, the bar on the right is the probability of occurrence. Based on the choice made for angle of attack, the graph was used to match the chosen α =12° to $C_L = 1.5$.

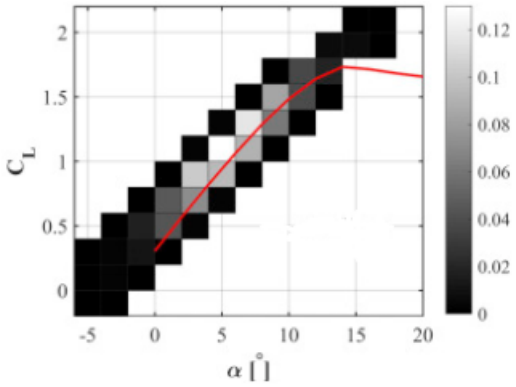

Figure 1: Relationship Between lift Coefficient and Angle of Attack

Because of industry standard for horizontal axis wind turbines, the amount of blades taken into consideration was 3, ($n = 3$). The problem statement gave a blade length of 10 ($R = 10$), which makes the radial

component a vector that goes from 0 to 10 ($r = 1{:}r = 10$).

1.2 Equations

When designing an ideal blade shape for a wind turbine the tip speed ratio (1) of the blades needs to be first taken into account. To find the optimal angular velocity used in this equation the time for the disturbed wind to re-establish itself (2) and the time for a blade to rotate into a previous blades position (3) need to be set equal to each other (4). If $t_s > t_w$ then some of the wind is not being used, and if $t_s < t_w$ it is not as efficient as it could be.

$$\lambda_{opt} = \frac{\omega_{opt} R}{V} \tag{1}$$

$$t_s = \frac{2\pi}{n\omega_{opt}} \tag{2}$$

$$t_w = \frac{S}{V} \tag{3}$$

$$t_s = t_w \tag{4}$$

$$\frac{2\pi}{n\omega_{opt}} = \frac{S}{V} \tag{5}$$

After these values are set equal to each other the optimal angular velocity can be isolated (6) and plugged into the equation for tip speed ratio making it the optimal tip speed ratio equation (8). A simplification is made to the equation, when solving for optima tip speed a term in the equation is isolated as the blade

length (R) over the length of disturbed wind speed (s). It is known that this value is estimated to be close to 2, and is substituted for that isolated value in the derivation.

$$\omega_{opt} = \frac{2\pi V}{nS} \tag{6}$$

$$\lambda_{opt} = \frac{2\pi}{n} \cdot \frac{R}{S} \tag{7}$$

$$\lambda_{opt} = \frac{4\pi}{n} \tag{8}$$

In the literature supplied a different tip speed ratio is listed, (9). This is a product of the optimal tip speed ratio (8), and the quotient of the radial distance over the blade length. This creates different tip speeds based on the distance down the blade. This tip speed ratio is used when solving for the chord length of the blade (10) and the angle of relative wind (11), both of these equations were supplied in the literature provided.

$$\lambda_{opt,r} = \lambda_{opt} \cdot \left(\frac{r}{R}\right) \tag{9}$$

$$c = \frac{8\pi r sin(\phi)}{3nC_L\lambda_{opt,r}} \tag{10}$$

$$\phi = tan^{-1}\left(\frac{2}{3\lambda_{opt,r}}\right) \tag{11}$$

1.3 Code Explanation

Packages were implemented at the top of the code so that a graph could be made of the turbine blade in 3D, trigonometry functions could be used, and so that the number of points in the radial distance vector could be chosen at the users digression. After all packages were implemented and constants were defined as stated above (1.1), a list was made of 100 points evenly spaced between 0 and the 10 and defined as the radial distance, r. The optimal tip speed ratio as defined above (8) was also calculated in the beginning of the code.

The way a majority of the code works is empty lists are created so that values can be appended to them in for loops. For example, the first for loop calculation for the relative tip speed iterates through all 100 points in the r list and calculates the relative tip speed. These values are appended to the empty list created prior at the same index as the r value used that iteration. This process is done for relative tip speed ($\lambda_{opt,r}$), relative wind angle (ϕ), and chord length (c).

There are now 4 list of length 100 of values specific to that radial position on the blade of the turbine. A new list was made which sums all of the relative wind angles and the constant angle of attack to get a blade angle list (what will be graphed). When the lists of chord lengths and blade angles were created, they were in polar coordinates as r and θ respectively. These values were then converted to Cartesian, and the mid points were isolated in a list to be graphed.

Plotted in 3D were the adjusted blade angle and chord length values, as well as the radial distance vector. Secondary lists were made of the inverse of those values so that the whole blade shape was graphed, (seen in red 2). The connecting lines (seen in blue 2), are created in a for loop that pairs the end points on the connecting lines and graphs them. The more radial distance values set at the beginning will lead to more connecting lines, although unless the user is using a super computer it is recommended that 50 - 100 points are used, so that the computer can handle the plotting. The plotted blade shape can be seen below, (2).

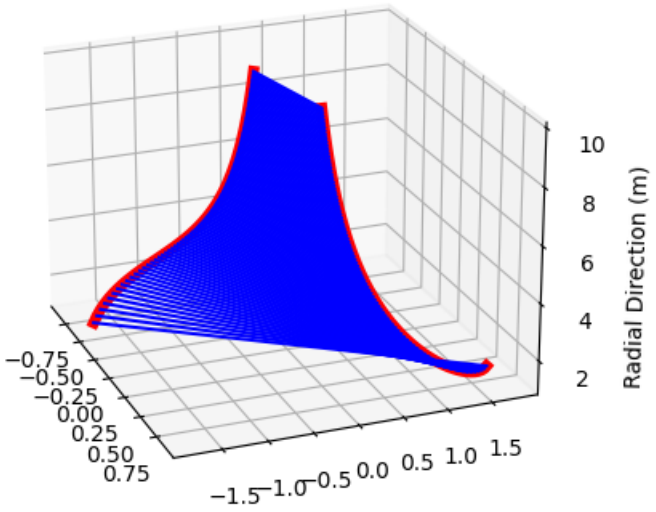

Figure 2: Ideal Blade Shape for a 10 m Wind Turbine Blade

2 Raw Code (Python)

```
# Colin Hodge
# ME 236 - Renewable Energy Harvesting
# Final Project
# Ideal turbine blade shape
#5/15/2021


# Importing requried packages
import matplotlib.pyplot as plt
import math
from mpl_toolkits.mplot3d import Axes3D
import numpy as np

# Defining constants
# Number of blades
N = 3

# Length of blades (m)
R = 10

# List of radial lengths (m)
r = np.linspace(1, 30, num = 100)

# Tip Speed ratio
lambda_opt = (4 * math.pi) / (N)

# Angle of attack/ Lift coefficient (assumed)
alpha = 12
cl = 1.5

# lambda_r list to append to
lambda_r = []

# Angle of relative wind list to append to
phi = []

# Chord length list to append to
chord_length = []

# Blade angle list to append to
blade_angle = []
```

```
# Cartesian points
cartx = []
carty = []

# Max/Min cartesian lists to append to (will be graphed)
x_min = []
x_max = []
y_min = []
y_max = []


# For loop to assign values for lambda_r
for i in r:
    tipspeed = lambda_opt * (i/R)
    lambda_r.append(tipspeed)


# For loop to assign values for phi
for i in lambda_r:
    rel_wind = math.atan(2/(3 * i))
    phi.append(rel_wind)

# For loop to assign values for chord_length
for i in range(0,len(r)):
    c = (8 * math.pi * r[i] * (math.sin(phi[i])) / ( 3 * N * cl * lambda_r[i]))
    chord_length.append(c)

# For loop to assign values for blade angle
for i in phi:
    a = (alpha * (math.pi / 180)) + i
    blade_angle.append(a)

# Converting values from polar to cartesian
for i in range(0,len(r)):
    x = (chord_length[i] * (math.cos(blade_angle[i])))
    y = (chord_length[i] * (math.sin(blade_angle[i])))
    cartx.append(x)
    carty.append(y)
```

```
# Centering values
for i in range(0,len(r)):
    x_small = (-1) * cartx[i] / 2
    x_large = cartx[i] / 2
    y_small = (-1) * carty[i] / 2
    y_large = carty[i] / 2

# Assigning to lists to be graphed
    x_min.append(x_small)
    x_max.append(x_large)
    y_min.append(y_small)
    y_max.append(y_large)


# Assigning variables to plot
# x values
i1 = x_min
i2 = x_max
# y values
j1 = y_min
j2 = y_max
# z values
k1 = r
k2 = r

# Plotting ideal blade shape
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
ax.plot(i1, j1, k1, color = 'red', linewidth= 5)
ax.plot(i2, j2, k2, color = 'red', linewidth= 5)

# Connecting lines
for i in range(len(i1)):
    ax.plot([i1[i], i2[i]], [j1[i], j2[i]], [k1[i], k2[i]], color = 'blue')

# Label and showing plot
ax.set_zlabel('Radial Direction (m)')
plt.show()
```

3 Bibliography

Bibliography

A. Rezaeiha, R. Pereira, and M. Kotsonis, "Fluctuations of angle of attack and lift coefficient and the resultant fatigue loads for a large Horizontal Axis Wind turbine," *Renewable Energy*, 27-Jul-2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0960148117307280. [Accessed: 16-May-2021].

"Aerodynamics of Wind Turbine Blades," *New Mexico MESA*, 2021. [Online]. Available: https://www.nmmesa.org/wp-content/uploads/2019/10/Aerodynamics-of-Wind-Turbine-Blades.pdf. [Accessed: 16-May-2021].