Diffuser Augmented Wind Turbine Analysis Code

By

Jonathan Carroll

Submitted to the graduate degree program in the Department of Aerospace Engineering and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Master of Science in Aerospace Engineering.

_____

Chairperson: Dr. Ray Taghavi

_____

Dr. Saeed Farokhi

_____

Dr. Charlie Zheng

Date Defended: 3-7-2014

The Thesis Committee for Jonathan Carroll

certifies that this is the approved version of the following thesis:

Diffuser Augmented Wind Turbine Analysis Code

_____

Chairperson: Dr. Ray Taghavi

Date approved: 03-07-2014

Abstract

Wind Energy is becoming a significant source of energy throughout the world. This ever increasing field will potentially reach the limit of availability and practicality with the wind farm sites and size of the turbine itself. Therefore, it is necessary to develop innovative wind capturing devices that can produce energy in the locations where large conventional horizontal axis wind turbines (HAWTs) are too impractical to install and operate. A diffuser augmented wind turbine (DAWT) is one such innovation. DAWTs increase the power output of the rotor by increasing the wind speed into the rotor using a duct. Currently, developing these turbines is an involved process using time consuming Computational Fluid Dynamics codes. A simple and quick design tool is necessary for designers to develop efficient energy capturing devices. This work lays out the theory for a quick analysis tool for DAWTs using an axisymmetric surface vorticity method. This method allows for quick analysis of duct, hubs and rotors giving designers a general idea of the power output of the proposed hub, blade and duct geometry. The method would be similar to the way blade element momentum theory is used to design conventional HAWTs. It is determined that the presented method is viable for preliminary design of DAWTs.

Acknowledgements

The author would like to thank Dr. Ray Taghavi, Dr. Kyle Wetzel and Dr. Willem Anemaat for their discussions and guidance throughout this project. The author would also like to thank my family and friends who have provided support in my ventures.

Table of Contents

List of Figures

List of Tables

List of Symbols

Symbol	Description	Units
A	Cross Sectional Area	m^2
A_{disc}	Disc Swept Area	m^2
A_H	Percentage of Total Area Occupied by Roughness Elements	~
a	Axial Induction Factor	~
a'	Angular Induction Factor	~
B	Number of Blades	~
$C_{F_{Duct}}$	Duct Force Coefficient	~
C_p	Pressure Coefficient	~
c	Blade Chord	m
c_d	Sectional Drag Coefficient	~
c_l	Sectional Lift Coefficient	~
$c_{z0}(\sigma_H)$	Constant Based on Roughness Element's Standard Deviation	~
d	Displacement Height	m
F_D	Drag Force	N
F_{Duct}	Diffuser Thrust	N
F_L	Lift Force	N
F_N	Normal Force	N
F_{rotor}	Rotor Thrust	N
F_T	Tangential Force	N
H	Total Enthalpy	~
$\bar{H}$	Average Height of Roughness Objects	m

K ..... Coupling Coefficient ..... ~

K' ..... Self-induced Coupling Coefficient ..... ~

k ..... Coupling Coefficient ..... ~

l ..... Length of Vortex Element ..... m

$\dot{m}$ ..... Mass Flow Rate ..... kg/s

P ..... Power ..... W

p ..... Pressure ..... Pa

Q ..... Torque ..... N-m

q ..... Induced Velocity ..... m/s

R_{Hub} ..... Hub Radius ..... m

R_{Tip} ..... Tip Radius ..... m

r ..... Local Radius ..... m

s ..... Distance Along Surface ..... m

T ..... Thrust ..... N

U ..... Horizontal Wind Speed ..... m/s

V ..... Vertical Wind Speed ..... m/s

u_* ..... Friction Velocity ..... m/s

v_s ..... Body Surface Velocity ..... m/s

W ..... Relative Wind Speed ..... m/s

x ..... X-coordinate ..... m

y ..... Y-coordinate ..... m

z ..... Height ..... m

z_{min} ..... Minimum Height for Log-Law ..... m

z_o ..... Roughness Height ..... m

α_∞ ..... Angle of Attack ..... deg

β ..... Local Body Surface Angle ..... deg

Δ ..... Differential ..... ~

ϕ ..... Angle of Relative Wind ..... deg

Γ ..... Vorticity Strength ..... m^2/s

γ ..... Vorticity Strength per Unit Length ..... m/s

κ ..... Von Karman Constant ..... ~

η ..... System Efficiency ..... ~

Ω ..... Rotor Rotational Speed ..... rad/s

ω ..... Local Rotational Speed ..... rad/s

ρ ..... Fluid Density ..... kg/m^3

σ ..... Source Strength ..... ~

Subscript	**Description**	**Units**
blade	Blade	~
D	Drag	~
Duct	Duct/Diffuser	~
k	Section k	~
L	Lift	~
Hub	Rotor Hub	~
m	Point m	~
N	Normal	~
n	Point n	~
Rotor	Rotor	~
rel	Local Relative Parameter	~

T ..... Tangential ..... ~

Tip ..... Blade Tip ..... ~

te ..... Trailing Edge ..... ~

rel ..... Local Relative Parameter ..... ~

wake ..... Wake ..... ~

1 Introduction

In recent years, the realization that the world's seemingly endless supply of fossil fuels is in fact limited, coupled with the increasing environmental concerns, has sparked a fire to explore other avenues to keep up with the ever increasing demand for energy. Two of the major players in this alternative energy competition are nuclear energy sources and renewable energy sources, such as solar and wind energy and biomass power plants. The focus of this thesis is the renewable energy side of this equation, more specifically: wind energy.

The majority of the energy produced from the wind comes from large wind farms containing hundreds of mega-watt wind machines. As with the limited supply of fossil fuels, there are a limited amount of sites on which to put these massive wind farms such that they produce a significant supply of energy and this number is depleting with each passing year. There are less desirable sites, which have a lower average wind speed, but the only way to get a return on the investment is to make the turbine larger to make up for the reduction in speed. This is not always possible as modern land based turbines are almost at the size limit for optimal operation and keeping the impact on the surrounding population and environment to a minimal level. Because of this, some wind turbine manufacturers are turning to off shore wind farms where the size of the turbine does not matter to the human population as much. The focus of this research is to take the opposite approach and make the wind turbine smaller to make it available to be installed in an urban setting increasing the viable wind energy sites. The main challenge for the urban wind turbine installations is the low average wind speed and higher turbulence intensity due to increased surface roughness of incoming winds. There is a potential to benefit from the urban environment. The urban installed wind turbines can take advantage of the accelerated flow around buildings and obstructions, which could potentially increase the power output of the wind machine.

1.1 Historical Background and Problem Definition

A review of the history behind conventional wind turbines and the development of the diffuser augmented wind turbine are discussed in the subsequent subsections. For more information on the historical background of wind turbines, see Reference 1 and Reference 2.

1.1.1 Wind Turbines: A Brief History

Wind turbines/windmills are not a very new concept. The first historical reference to a windmill is given by Hero of Alexandria in the first century, B.C. or A.D. Hero describes a device, which provides air to an organ using a windmill. There have been some debates whether this machine actually existed. The next time that windmills are referenced is in the ninth century A.D. by Al Masudi. These windmills were located in the eastern part of what is now Iran and were vertical axis wind turbines as shown in Figure 1.1.

Figure 1.1 Ninth Century Windmills[3]

Windmills made their way to Europe by the twelfth century A.D. These windmills were considerably different from their middle-eastern counterparts. All European windmills were horizontal axis wind turbines and were used for almost any mechanical task (pumping water, sanding wood, grinding grain, etc.). An interesting feature of the early windmills was that they were built on posts such that the whole windmill could be turned into the wind when the

direction changed. Also, these European horizontal axis windmills were lift-based machines and were more efficient than the middle-eastern vertical axis windmills, which are drag based machines. As the European windmill evolved the construction evolved as well and, prior to the abandoning of wind energy for more portable and storable energy resources (coal and water), the windmill technology developed such that only the top of the windmill yawed into the wind and the lower portion of the mill was stationary. In the 18^{th} century the scientific testing and evaluation of windmills was introduced. An English man by the name of John Smeaton discovered the following rules for windmills than are still applicable to today's windmills and wind turbines:

- The tip speed of the blade is ideally proportional to the speed of the wind
- The maximum torque is proportional to the square of the wind speed
- The maximum power is proportional to the cube of the wind speed

A popular windmill that was primarily used in the American west is the fan mill. This windmill was used primarily to pump water and was developed in Europe during the final years of the age of the European windmills. This windmill had a simple regulating system that allows the mill to be run unattended for long periods of time.

Electrical generators appeared at the end of the nineteenth century and an American by the name of Charles Brush built one of the first wind turbines to generate electricity in 1888. These wind turbines started to resemble the wind turbines used today with three blades and true airfoil shapes on the blades. These early electric wind turbines did not establish an interest in wind energy and developments in wind energy waned until the late 1960s. After the re-emergence of wind energy in the late 1960s, the interest in wind energy has developed the wind turbines that are around today. Figure 1.2 shows the evolution of the size and power extraction capabilities of today's wind turbines. The evolutionary period of wind turbines is far from over. While higher velocity on-shore sites are getting fewer and fewer, a new avenue of off-shore wind sites has opened up, but these sites are not without challenges to overcome.

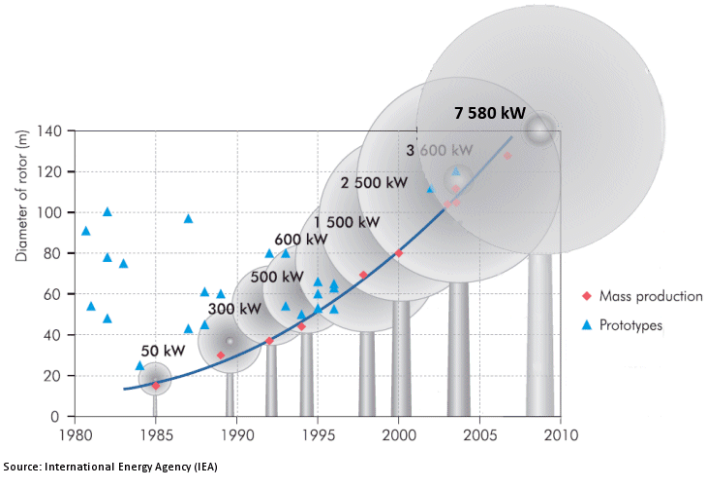

Figure 1.2 Wind Generator Evolution

1.2 Problem Definition

The aim of this research is to develop a preliminary aerodynamic design code for DAWTs. This preliminary design analysis will include the duct and rotor influences on each other to give the designer a better idea of the power output that the system could generate. The intent is to develop a methodology for DAWTs much like the Blade Element Momentum that is used for bare horizontal axis wind turbines (HAWTs). The idea is to sacrifice some accuracy for simplicity and speed to narrow down the design field.

Diffuser Augmented Wind Turbine (DAWT) physics and historical background will be discussed. The main application of these DAWTs is expected to be small scale and in urban environments, therefore wind characteristics are discussed in Section 3. The HAWT and DAWT aerodynamic models are discussed in Section 4, followed by the methodology of the developed design code. After this, a validation of the code is presented followed by comparison with experimental data.

2 Diffuser Augmented Wind Turbine

DAWTs use a duct or shroud around the wind turbine to increase the mass flow through the turbine. A modern DAWT is shown in Figure 2.1.

Figure 2.1 Diffuser Augmented Wind Turbine

Phillips[4] outlines advantages of DAWTs are as follows:

- Smaller rotor diameters to produce the same amount of power as conventional horizontal axis wind turbines.
- Lower cut-in wind speeds than conventional wind turbines
- Lower rotor axial loads
- Lower turbulence levels at the rotor plane due to the contraction of the flow
- Increase rotor RPM resulting in a reduction of gearbox ratios
- Reduced noise levels
- Reduction in tip losses of the rotor
- Reduction in yaw sensitivity

All of these benefits come at a price. The construction of the duct increases the material, fabrication, transportation and installation costs. DAWTs are also more susceptible to environmental effects such as the aggregation of ice, snow, temperature fluctuations and

windborne particulates. The diffusers also could have some aeroelastic instabilities caused by flow separation. The flow separation would also lead to fluctuating power output.

DAWTs also suffer from increased tower structure from the weight of the whole system as well as increase tower top loads due to the drag of the duct. The visual impact of the DAWT system could potentially limit the number of available installation sites. All of these drawbacks may cause the only potential commercially viable large-scale DAWT sites to be off shore. However, small scale DAWTs could be commercially viable in urban environments, depending on the siting of the wind turbine.

2.1 DAWT Physics

During the operation of ideal HAWTs, the velocity at the rotor plane is reduced to 2/3 of the free stream velocity. The reduction in flow velocity causes an increase in pressure and this causes some of the mass flow in the "captured" stream tube to be pushed radially outward and around the rotor. This loss in mass flow reduces the effective area of the rotor to 2/3 of the swept rotor surface. DAWTs increase the effective area of the rotor by adding a radial force to counteract the pressure force pushing the airflow around the rotor. The radial force is generated by placing an annular lifting surface with the suction side toward the hub of the rotor. The addition of the duct/diffuser causes a force imbalance and the only way for the system to reach an equilibrium state is to increase the mass flow through the duct/diffuser increasing the capture area of the turbine. The increase in mass flow causes an increase in flow velocity through the rotor and since the power output is proportional to the cube of the velocity causes an increase in power extracted from the wind.

2.2 DAWT Historical Review

The initial development and debates of the merits of the diffuser augmented wind turbine occurred in the 1920s. A well-known figure in the wind industry, Alexander Betz, was one of the first to develop a DAWT theory, where he assumed the pressure at the diffuser exit plane was equal to ambient pressure[5]. Betz acknowledged the potential of the ducted/shrouded wind turbine although he concluded that the designs and applications at that time were uneconomical

for production. Based on these conclusions, further investigations were discarded until the 1950s when shrouded turbine research was revisited by Japanese researchers Sanuki[6] and Iwasaki[7] as well as English researchers Lilley and Rainbird[8]. Sanuki performed some experiments in which an increase in power of 88% relative to the Betz limit was obtained. Iwasaki achieved a power increase of 30% by placing a cylindrical duct around a rotor. Lilley and Rainbird mainly worked on theoretical studies of ducted wind turbines. During this work, Lilley and Rainbird determined that the increase in axial velocity and decrease in tip losses were major contributors to the increase in extracted power. They also suggested that the addition of an aerodynamic surface at the diffuser exit, similar to a flap, could increase the power extracted even further. Through the theoretical research Lilley and Rainbird concluded that the gains in power extraction would be at least 65% of the Betz limit with a properly designed duct.

By the end of the 1950s and into the 1960s, investigations into DAWTs were considered by a Isreali research team headed by Kogan[9,10]. Kogan did a proof of concept study for a ducted wind turbine. One of the first ducts that Kogan considered is shown in Figure 2.2.

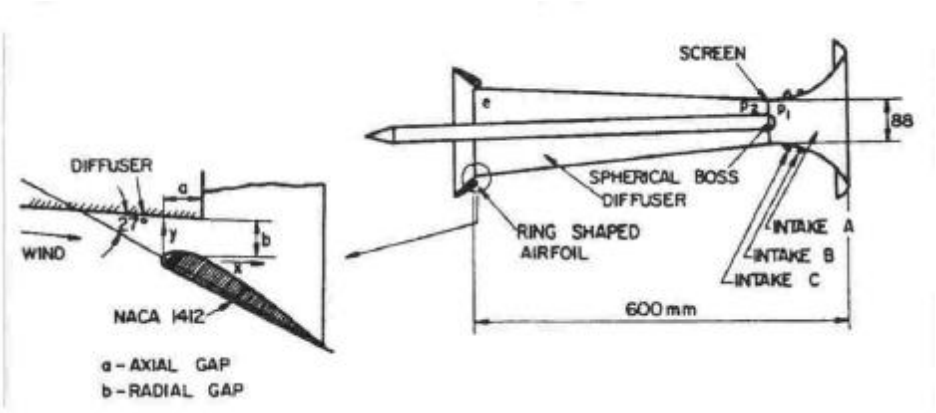

Figure 2.2 First Generation Duct Researched by Kogan[11]

During this research, power augmentation factors of about 3.5 were achieved. However, the size of the duct geometry proved too large to provide a practical wind energy machine. By the end of Kogan's research work in 1967, the size of the duct was reduced while maintaining large power augmentation factors. The later designs employed ring shaped flaps at the duct exit plane to reduce the exit pressure and overcome the positive pressure gradients in the diffuser.

In 1974, a student of Kogan, Igra[11,12], continued on with the DAWT research. Igra suggests that the increase in mass flow and therefore, power is a result of the pressure drop at the diffuser exit, essentially "sucking" air through the diffuser. By the end of the 1970s, Igra had constructed a prototype DAWT. A cross section of Igra's third generation diffuser model is given in Figure 2.3.

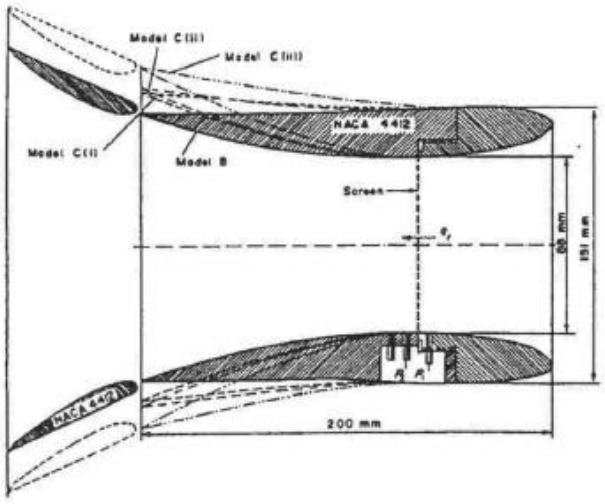

Figure 2.3 Third Generation Diffuser Models with NACA Cross Section[12]

Igra was not alone in 1974 in DAWT research. Foreman[13,14,15,16] of Grumman Aerospace in the United States joined the research effort due to an increase in alternative energy sources caused by the oil crisis in 1974. Foreman's research focused on controlling the boundary layer in the duct using slots. Foreman tested over 100 models in the wind tunnel. In some of Foreman's later research it was determined that swirling flow in the rotor wake increased the energy in the boundary layer and therefore delayed separation in the diffuser due to momentum transfer. Foreman also analyzed the economic and design implications of DAWTs. By 1976 an English researcher by the name of Lewis[17] joined Igra and Foreman in DAWT research and conducted some theoretical and experimental analyses. Lewis determined through a momentum analysis that a DAWT could extract 42.86% more energy from the flow than a similar sized HAWT. Lewis, Foreman and Igra recognized the importance of reducing cost of the duct in order to have a commercially viable product and worked to reduce the costs associated with the use of a diffuser. Due to the promising results of Lewis, Foreman and Igra, a renewed interest in

DAWTs was ignited and many theoretical and experimental investigations into DAWT operations were conducted during the 1970s and 1980s.

Up to this point, no one had tried to market a commercial DAWT, despite all the promising results of the research presented above. In 1995, the first private company tried to bring the DAWT concept to market. Vortec Energy Limited, bought the rights to the DAWT design from Grumman Aerospace[4]. The 17.3 meter prototype Vortec 7 was built in 1997. The Vortec 7 was based off of Foreman's design and optimized with the use of CFD and a series of small scale experiments. The Vortec 7 was expected to reach power augmentation levels of about 9, but the actual data was far from the predicted at about 2.4. The velocity profile at the rotor plane showed lower speeds closer to the hub and a high speed region beyond the blade tips[18]. This produced a lower power output than predicted as the initial assumption of a uniform velocity profile across the blade plane was inadequate to model the problem accurately. This miscalculation led to the eventual closure of Vortec Energy Limited. Despite the commercial miscalculation of Vortec, several companies are attempting to take smaller scale DAWTs to market, but have yet to achieve big commercial success. The Vortec 7 DAWT is shown in Figure 2.4.

Figure 2.4 Vortec 7 DAWT

Research into DAWT theory has been continually evolving. Van Bussel, Jameison, Werle and Pretz [19, 20, 21] have reviewed the DAWT theories and have attempted to reignite interest in the DAWT through attempts to use momentum theory to calculate the power production of a DAWT. With modern computing power, more CFD studies are being performed on DAWTs including blade design and shroud design[22,23,24,25,26,27]. There has also been an attempt to reduce the pressure in the wake downstream of the duct by putting a flange around the trailing edge of the diffuser, which causes the flow to separate and creates a large low pressure region downstream of the duct effectively creating a suction force through the duct[28,29]. In 2008, a FloDesign[30] developed a DAWT with a lobed trailing edge to promote wake mixing aft of the rotor and increase mass flow through the rotor. FloDesign's DAWT is shown in Figure 2.5.

Figure 2.5 FloDesign's Lobed Diffuser[30]

Recently, DAWT research has been done by Ten Hoopen[5] and van Dorst[31] of Delft University of Technology and Widnall[32] of the Massachusetts Institute of Technology, using axisymmetric vorticity method to calculate the influence of the duct and rotor on the flow field. This thesis builds on the work done by Ten Hoopen and van Dorst. Ten Hoopen's main focus was the effect of placing vortex generators at the trailing edge of the duct to enhance the wake mixing behind the rotor. Van Dorst improved the blades of an existing DAWT. Both, van Dorst and Ten Hoopen utilized the surface vorticity method in the development and analysis of the DAWT.

3 Wind Characteristics

The wind resources and characteristics of a particular site are instrumental in the system design, performance evaluation, siting and operations of a wind turbine. The following sections deal with the wind characteristics.

3.1 Rural Wind Characteristics

Out in the plains of today's large wind farms the wind profile can be assumed to have the following relationship because of the minimal surface roughness introduced (i.e. minimal obstructions such as buildings, trees, mountains etc.) at the ground level. The following equation is known as the Log-law for wind velocity profiles.

$$U(z) = \frac{u_*}{\kappa} \ln\left(\frac{z}{z_o}\right) \tag{1}$$

This equation comes from atmospheric turbulence and, due to the no-slip condition at the surface of the earth an atmospheric boundary layer is formed. A graphical representation of this wind velocity profile is shown in Figure 3.1.

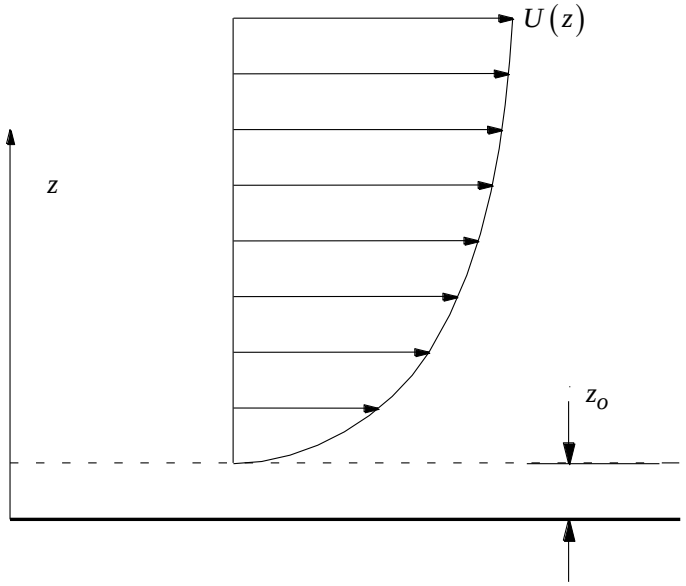

Figure 3.1 Wind Velocity Profile for Log-law

Manwell[1] gives typical surface roughness parameters for various relatively smooth terrains. These are given in Table 3.1.

Table 3.1 Surface Roughness for Relatively Smooth Terrains

Terrain	z_o [mm]
Very smooth (ice or mud)	0.01
Calm open sea	0.20
Blown sea	0.50
Snow surface	3.00
Lawn grass	8.00
Rough pasture	10.00
Fallow field	30.00
Crops	50.00

Most wind maps, as shown in Figure 3.2, give the average wind speed at a reference height of 10 meters with an assumed surface roughness height of 0.3 meters.

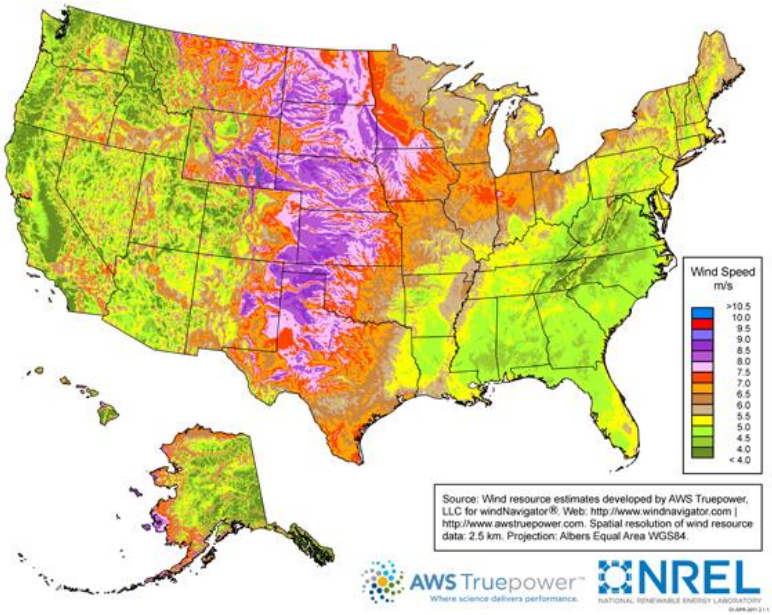

Figure 3.2 American Wind Map[33]

3.2 Urban Wind Characteristics

The Log-law wind shear profile can be extended to urban environments by increasing the value for the surface roughness length, z_o. Manwell[1] gives typical values for obstructed flow surface roughness lengths. These are given in Table 3.2.

Table 3.2 Typical Surface Roughness Values for Various Terrain Types

Terrain	z_o [m]
Few trees	0.10
Many trees, hedges, few buildings	0.25
Forest and woodlands	0.50
Suburbs	1.50
Centers of cities with tall buildings	3.00

Figure 3.3 shows a sketch of the average velocity profile in an urban setting, where the surface roughness is very large. The average velocity profile for low surface roughness areas is also shown for comparison. It is interesting to note the slight velocity increase at the top of the building structure. This is a local flow feature and will be discussed in more detail in the subsequent sections.

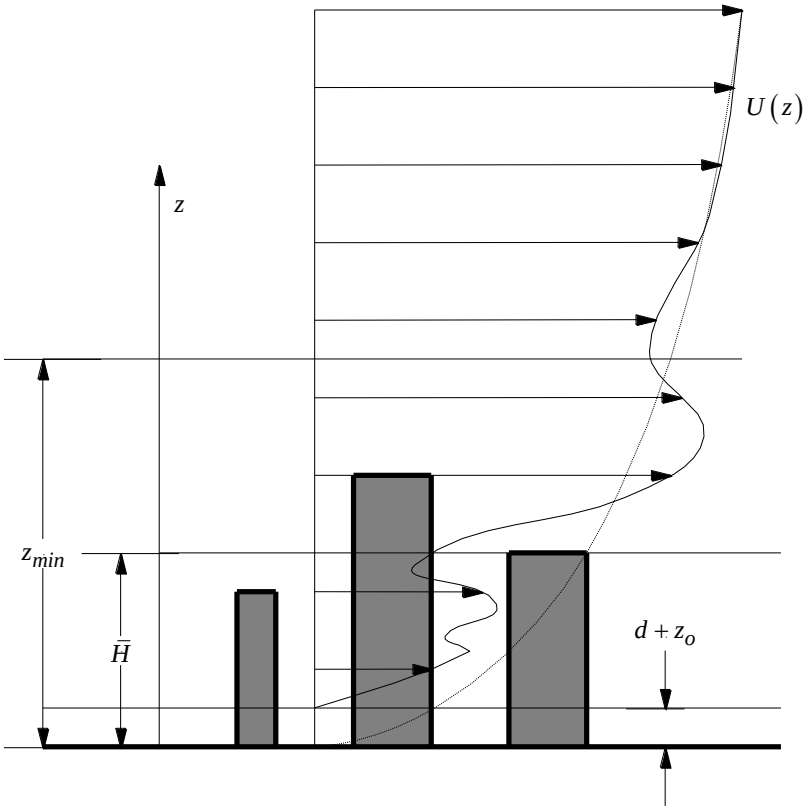

Figure 3.3 Sketch of Average Velocity Profile in an Urban Environment

Reference 34 presents another modification to the Log-law for urban environments. This modification of the grassland log-law adds a virtual displacement height, d, to the surface roughness height. The log-law for the urban environment is then:

$$U(z) = \frac{u_*}{\kappa} \ln\left(\frac{z-d}{z_o}\right) \tag{2}$$

Where the displacement height, d, is given as a function of the average height of the roughness objects, $\bar{H}$, the roughness height, z_o, and the percentage of the total area occupied by the roughness elements, A_H:

$$d = \bar{H} - 4.3 z_o \left(1 - A_H\right) \; for \; 0.2 \le A_h \le 0.8 \tag{3}$$

The roughness height is given as follows:

$$z_o = c_{z0}(\sigma_H) A_H \bar{H} \quad (4)$$

$$z_{min} = 1.5d \quad (5)$$

Table 3.3 shows typical values for the parameters used in the above equations for built environments[34].

Table 3.3 Typical Values for Wind Shear Parameters

Parameter	Value
A_H	0.42
$\bar{H}$	25
$c_{z0}(\sigma_H)$	0.08

Great care should be utilized when these equations are used. The log-law equations are based on semi-empirical data, but are inadequate to approximate the complex flow characteristics of flow around and through a built environment.

3.3 Wind Turbines in the Urban Environment

It is important to understand the wind characteristics around buildings to adequately site the wind turbine. Unlike the rural counterparts, wind turbines in the urban settings have to operate in regions of flow influenced by the surrounding buildings. As stated earlier, the airflow around buildings and/or obstructions is accelerated due to the low-pressure zones around the obstruction. These regions of accelerated flow can be used to further augment the power output of a DAWT. The power about a cylindrical shaped building could be increased by as much as a factor of 8.[34] This value is less for rectangular buildings and obstructions, but there is still significant acceleration of the airflow around these obstructions. The proximity and shapes of the buildings also have the ability to increase the flow velocity similar to the way a duct increases the velocity at the rotor plane. Figure 3.4 shows the velocity vectors around a 2D building. As the figure shows, there is a region of accelerated flow outside of the region of separated flow.

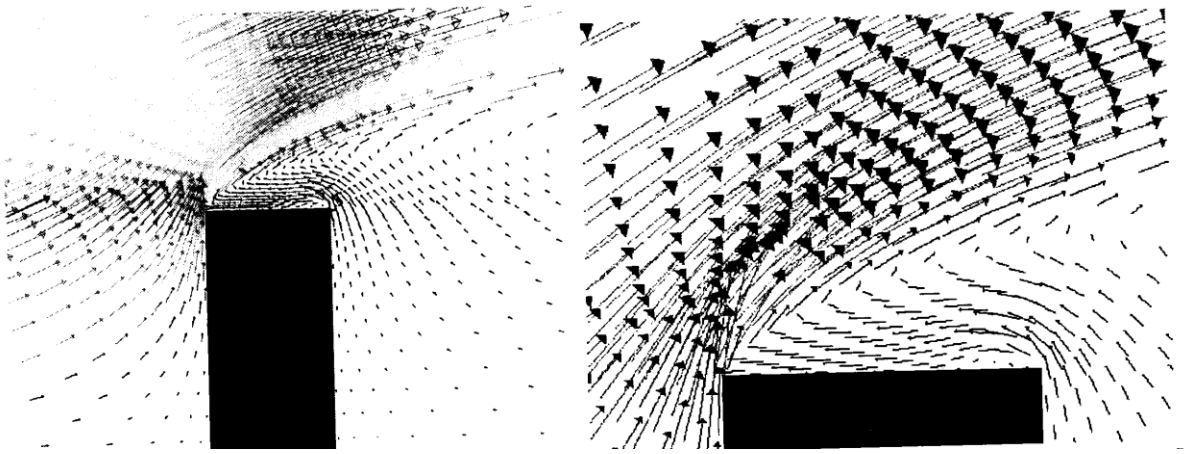

Figure 3.4 2D CFD Simulation of the Flow Around a Building[34]

While there is significant acceleration of the flow around buildings the flow phenomenon around the obstruction will affect the siting of the wind turbine. The boundary layer separates at the edge of the building. These regions of separation cause low flow velocities and high turbulence levels. It is important to avoid these regions of separated flow when siting the wind turbine.

Grant et al.[35] discusses the potential of wind turbines in the urban environment, specifically ducted wind turbines mounted on buildings. While public concerns over safety, noise and vibration are valid, most of these points are reduced through the use of DAWTs. While, Grant's ducted turbine is not conventional, data is provided comparing a conventional wind turbine (WTG), two DAWTs (DWT(S) and DWT(W)) and a photovoltaic system (PV). This is shown in Figure 3.5.

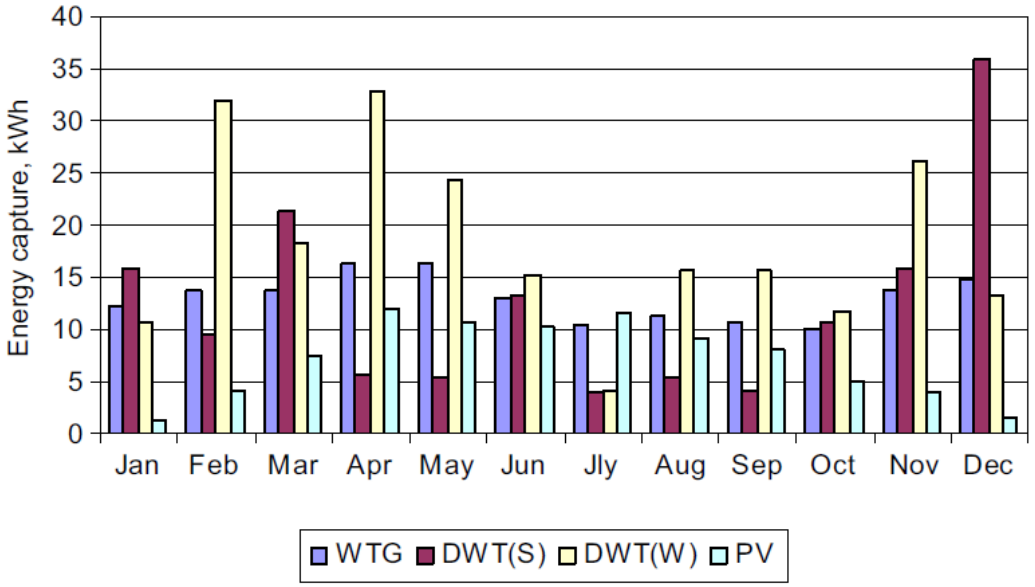

Figure 3.5 Urban Energy Capture Predictions[35]

The difference between the ducted wind turbines, DWT(S) and DWT(W) is the direction the wind turbine is facing. As Figure 3.5 shows, the power output of the DAWT is highly dependent on the siting of the wind turbine and it is better to have a rotating DAWT that can rotate into the oncoming wind for urban applications.

4 Methodology

Preliminary design methods are a compromise between accuracy and computational time. At the beginning stages of design it is necessary to run through numerous designs to determine the merits of each design. Therefore a numerical model of the DAWT is created to facilitate design iterations. The numerical model selected is the axisymmetric surface vorticity method. The initial incoming velocity is determined using the axisymetric surface vorticity method for a duct and center body without the presence of a rotor. The initial incoming flow is then used to calculate the forces on the blades of the turbine and the wake structure behind the rotor. Using the updated forces on the blades and the rotor wake, the velocity at the rotor plane is recalculated and the process is iterated until convergence on a solution within a specified tolerance. Each process outlined above is described in the following subsections.

4.1 Horizontal Axis Wind Turbine Momentum Method

The DAWT momentum method is often similar to and often based on the axial momentum method developed for bare wind turbines. Therefore the axial momentum of a bare wind turbine is discussed and then the momentum theory for DAWTs is discussed.

4.1.1 Bare Wind Turbine Blade Element Momentum Method

The preliminary performance analysis and design of traditional bare Horizontal Axis Wind Turbines (HAWTs) is usually performed by a Blade Element Momentum (BEM) analysis method. The BEM analysis method uses strip theory and momentum analysis to solve for the blade strip forces and the inflow at the rotor plane. Initially, the momentum method assumes the following:

- Homogeneous, incompressible, steady state fluid flow
- No frictional drag
- Rotor has an infinite number of blades
- Uniform thrust over disc or rotor area
- Non-rotating wake
- Static pressure far upstream and far downstream of the rotor is equal to the ambient static pressure

Wake rotation, number of blades, drag and non-uniform thrust is taken into account later on in the analysis. Momentum analysis of the bare HAWT assumes a control volume where the boundaries are the rotor captured stream tube surfaces and two cross sections of the stream tube far enough upstream and downstream, such that the static pressure in the stream tube is equal to the ambient static pressure, as shown in Figure 4.1.

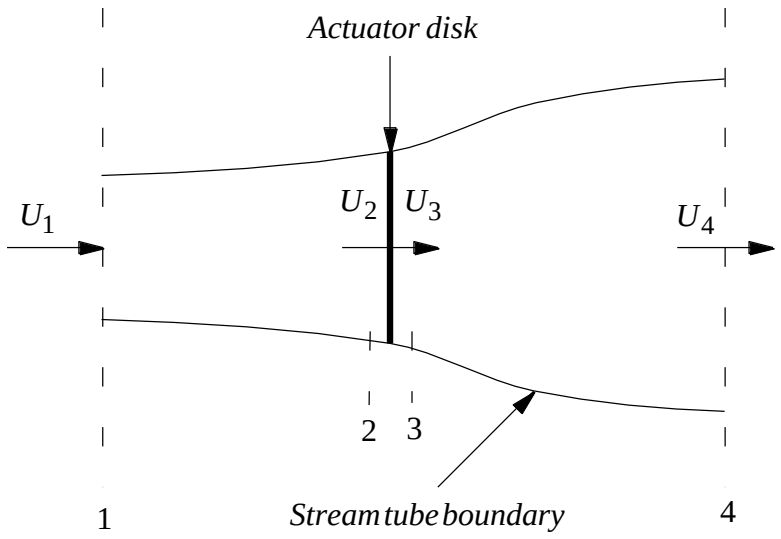

Figure 4.1 HAWT Momentum Analysis Definition Sketch

From the conservation of linear momentum, the thrust of the rotor is equal and opposite to the rate of change of momentum in the stream tube, namely:

$$T = U_1(\rho AU)_1 - U_4(\rho AU)_4 \tag{6}$$

Since the mass flow rate, $\dot{m} = (\rho AU)_1 = (\rho AU)_4$, is constant in the control volume, Equation (6) is:

$$T = \dot{m}(U_1 - U_4) \tag{7}$$

No work is done upstream or downstream of the rotor plane, therefore Bernoulli's equation can be applied upstream and downstream of the rotor. Bernoulli's equation for upstream of the rotor is:

$$p_1 + \frac{1}{2}\rho U_1^2 = p_2 + \frac{1}{2}\rho U_2^2 \tag{8}$$

Applying Bernoulli's equation downstream of the rotor plane gives:

$$p_3 + \frac{1}{2}\rho U_3^2 = p_4 + \frac{1}{2}\rho U_4^2 \tag{9}$$

Here it is assumed that the upstream and downstream pressures are equal ($p_1 = p_4$) and the velocity just before and just behind the rotor plane is equal ($U_2 = U_3$). The thrust of the disc is also defined by the pressure differential and the area of the disc, namely:

$$T = A_{disc}\left(p_2 - p_3\right) \tag{10}$$

Using the Bernoulli equations upstream and downstream of the rotor, the term $\left(p_2 - p_3\right)$ can be determined. Substituting this into Equation (10), the rotor disc thrust becomes:

$$T = \frac{1}{2}\rho A_{disc}\left(U_1^2 - U_4^2\right) \tag{11}$$

Equating the thrust equations (Equation (7) and Equation (11)) produces the following relationship:

$$U_{disc} = \frac{U_1 + U_4}{2} \tag{12}$$

Defining the axial induction factor as:

$$a = \frac{U_1 - U_2}{U_1} \tag{13}$$

The velocity at the rotor disc plane is then given as:

$$U_2 = U_1\left(1 - a\right) \tag{14}$$

The velocity far downstream of the rotor is then:

$$U_4 = U_1\left(1 - 2a\right) \tag{15}$$

The power of the disc is equal to the thrust of the disc times the velocity at the disc, namely:

$$P = \frac{1}{2}\rho A_{disc} U_2\left(U_1 + U_4\right)\left(U_1 - U_4\right) \tag{16}$$

Substitution of Equation (14) and Equation (15) into Equation (16) yields the following equation for the power output of the rotor disc.

$$P = \frac{1}{2}\rho A_{disc} U_1^3 4a\left(1-a\right)^2 \tag{17}$$

Substitution of Equation (14) and Equation (15) into Equation (11) produces the following thrust equation.

$$T = \frac{1}{2}\rho A_{disc} U_1^2 \left[4a\left(1-a\right)\right] \tag{18}$$

The thrust on an annular element is then:

$$dT = 4a\left(1-a\right)\frac{1}{2}\rho U_1^2 2\pi r dr \tag{19}$$

To include the effects of wake rotation to the analysis, a control volume that moves with the angular velocity of the blades must be used. For more information on the derivation of this, see Glauert[36].

The angular induction factor is defined by:

$$a' = \frac{\omega}{2\Omega} \tag{20}$$

Using the conservation of angular momentum, the effect of the wake rotation can be included in the calculation of the torque of the rotor disc. The torque on the rotor disc must be equal to the change in angular momentum of the wake. The torque on an elemental annulus is given as:

$$dQ = d\dot{m}\left(\omega r\right)r = \rho U_2 2\pi r dr\left(\omega r\right)r \tag{21}$$

Substituting Equation (14) and Equation (20) into Equation (21) produces the following equation for the elemental torque.

$$dQ = 4a'\left(1-a\right)\frac{1}{2}\rho U\Omega r^2 2\pi r dr \tag{22}$$

The elemental power is given as:

$$dP = \Omega dQ = 4a'\left(1-a\right)\frac{1}{2}\rho U\Omega^2 r^3 2\pi r dr \tag{23}$$

The blade element forces need to be determined to use the momentum analysis for a particular HAWT blade. The blade element portion of this method assumes the following:

- There is no aerodynamic interaction of the blade elements
- The lift and drag characteristics are based on 2D analysis of the airfoils (i.e. no 3D effects are accounted for)

Figure 4.2 shows the blade geometry used in the Blade Element analysis.

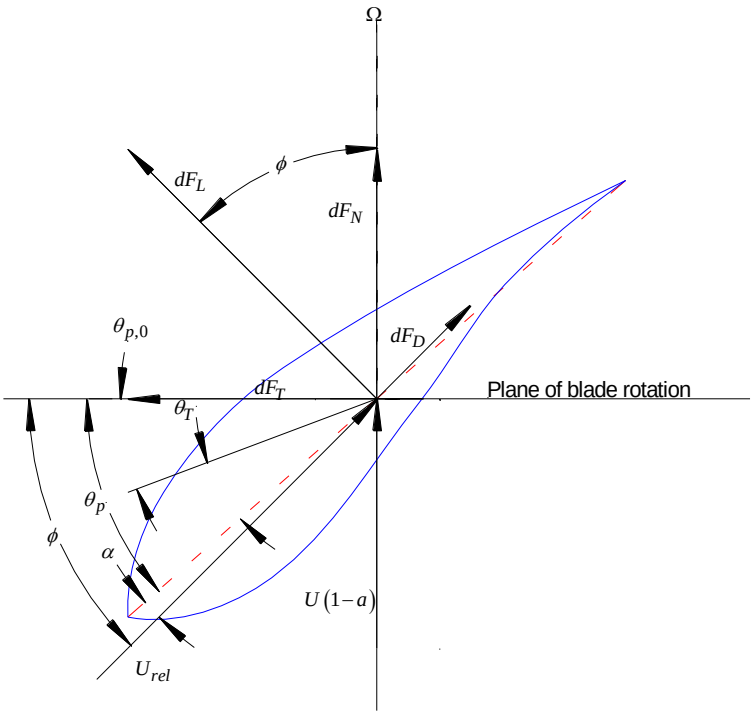

Figure 4.2 Blade Section Geometry Definition

From Figure 4.2 the following relationships can be determined:

$$\tan\phi = \frac{U\left(1-a\right)}{\Omega r\left(1+a'\right)} \tag{24}$$

$$U_{rel} = \frac{U(1-a)}{\sin\phi} \tag{25}$$

$$dF_L = c_l \frac{1}{2} \rho U_{rel}^2 c dr \tag{26}$$

$$dF_D = c_d \frac{1}{2} \rho U_{rel}^2 c dr \tag{27}$$

$$dF_N = dF_L \cos\phi + dF_D \sin\phi \tag{28}$$

$$dF_T = dF_L \sin\phi - dF_D \cos\phi \tag{29}$$

The elemental normal force of a rotor with number of blades, B, is then given by:

$$dF_N = B \frac{1}{2} \rho U_{rel}^2 \left(c_l \cos\phi + c_d \sin\phi\right) c dr \tag{30}$$

The elemental torque is then given by:

$$dQ = B r dF_T = B \frac{1}{2} \rho U_{rel}^2 \left(c_l \sin\phi - c_d \cos\phi\right) c r dr \tag{31}$$

Assuming the normal force from the blade element analysis is equal to the thrust from momentum analysis and equating the torque equations, the axial and angular induction factors can be determined through an iterative solution method. This iteration is initiated by assuming an initial value for the induction factors and iterating until the induction factors converge to a solution. The total power and thrust of the rotor can then be calculated as follows:

$$P = \int_{R_{Hub}}^{R_{Tip}} dP = \int_{R_{Hub}}^{R_{Tip}} \Omega dQ \tag{32}$$

$$T = \int_{R_{Hub}}^{R_{Tip}} dT \tag{33}$$

4.1.2 Ducted Wind Turbine Momentum Method

Werle and Perez[21] developed a momentum theory for ducted turbines incorporating the duct influence by including the axial force exerted on the flow by the duct.

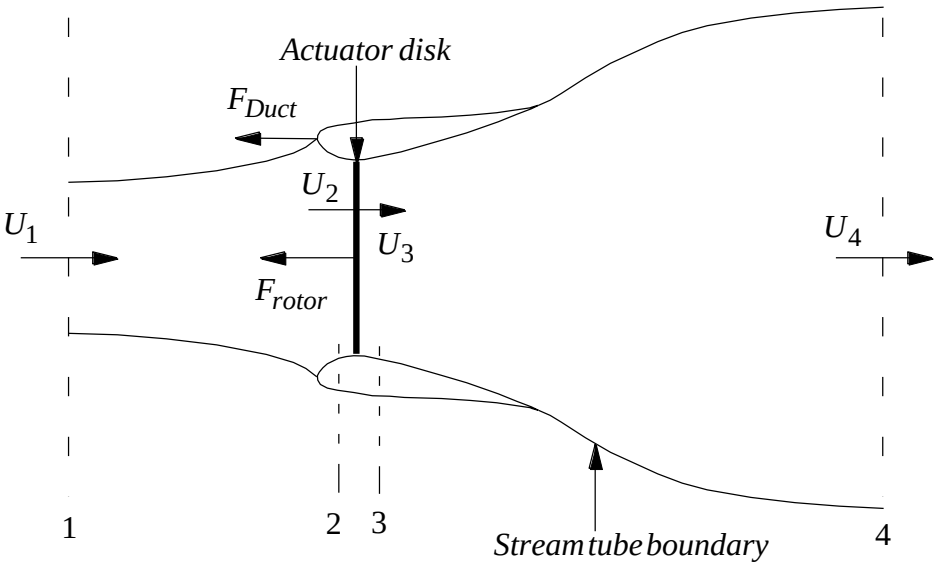

Figure 4.3 DAWT Momentum Theory Definition Sketch

The thrust on a control volume enclosed by the stream tube and planes 1 and 4 defined by Figure 4.3 is calculated by:

$$T = F_{Rotor} + F_{Duct} = \dot{m}\left(U_1 - U_4\right) = A_{Rotor}\left(p_3 - p_2\right) + F_{Duct} \tag{34}$$

From a Bernoulli analysis of Figure 4.3, the pressure differential over the rotor is given as:

$$\left(p_2 - p_3\right) = \frac{1}{2}\rho\left(U_1^2 - U_4^2\right) \tag{35}$$

The duct force can be normalized as follows:

$$C_{F_{Duct}} = \frac{F_{Duct}}{\frac{1}{2}\rho A_{Duct}\left(U_4^2 - U_1^2\right)} \tag{36}$$

Using Equation (36) and Equation (35) yields the following equation for the velocity at the rotor plane.

$$U_2 = \frac{1}{2}\left(1+C_{F_{Duct}}\right)\left(U_4+U_1\right) \tag{37}$$

Using the following relationship, the velocity at the rotor plane can be calculated by Equation (39)

$$\frac{U_4}{U_1} = 1-2a \tag{38}$$

$$U_2 = U_1\left(1+C_{F_{Duct}}\right)\left(U_4^2-U_1^2\right) \tag{39}$$

Using induction factors, the force on the control volume is then:

$$T = \frac{1}{2}\rho A_{rotor}U_1^2\left(4a\left(1-a\right)+C_{F_{Duct}}\right) \tag{40}$$

The power is then:

$$P = \frac{1}{2}\rho A_{rotor}U_1^3\left(4a\left(1-a\right)^2\left(1+C_{F_{Duct}}\right)\right) \tag{41}$$

The force on the duct is required for this method. The duct force can be determined from experimental data, surface vorticity method or Computational Fluid Dynamics (CFD). This method does not take into account the rotor influence on the duct, although the experimental data utilizes a screen mesh or gauze to simulate the rotor.

4.2 Current Method

The current method tries to make up for some of the restrictions of the DAWT momentum analysis by solving for the forces acting on the duct and calculating the influence of the duct on the incoming flow as well as the effect of the rotor on the incoming flow. The duct, hub and

rotor interactions on the incoming flow are the most important part in analyzing a DAWTs performance.

4.2.1 Duct and Hub Model

The duct and hub influence on the flow field is approximated using the axisymmetric surface vorticity method outlined in Reference 37. This section discusses the formulation of the 2D surface vorticity method and its extension to 3 dimensional flow assuming axisymmetric flow fields.

4.2.1.1 Physical Significance of Surface Vorticity Model

The surface vorticity model has an advantage over source models as it represents an actual simulation of an ideal fluid flow. All real fluid flows contain a region near the body where viscous effects dominate the fluid flow, known as the boundary layer. The boundary layer has enough vorticity to reduce the free stream velocity just outside of the boundary layer to zero at the body surface. As the Reynolds number approaches infinity, i.e. the flow becomes inviscid, the boundary layer is compacted down onto the surface of the body generating a surface vorticity sheet of strength, $\gamma(s)$. The flow would then go from zero directly beneath the body surface vorticity sheet to v_s above and parallel to the body surface. If it is assumed that there are no separation effects or turbulent effects that are present in real flows at high Reynolds numbers, it can be seen that the inviscid flow model can be seen as a special case of an infinite Reynolds number flow.

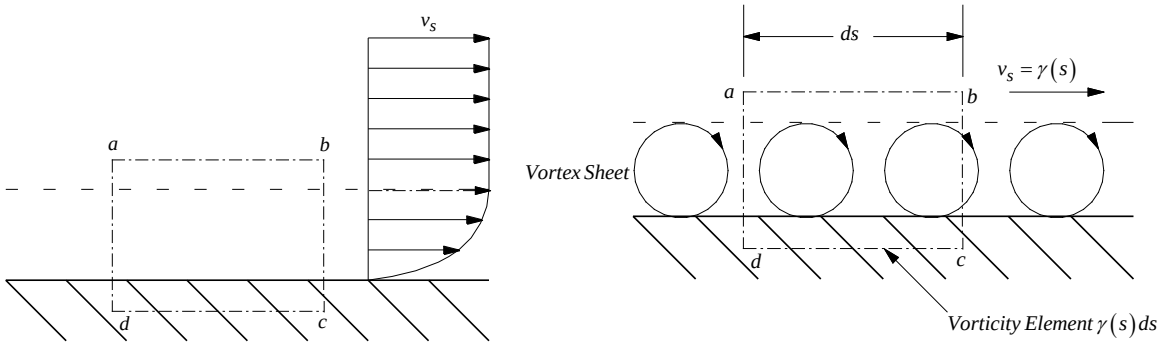

Figure 4.4 Boundary Layer and Surface Vorticity Approximation

It should be noted that viscous effects including turbulence and separation are important parameters in determining the flow field. These effects will be ignored in the current model, but should be considered and more research into methods for including these effects in the current method is to be done.

4.2.1.2 Two Dimensional Surface Vorticity Formulation

The flow past a two-dimensional body in the (x, y) plane in a flow field with a uniform free stream velocity of W_∞ and at an angle of attack α_∞ can be represented by covering the surface of the body in a vorticity sheet of strength $\gamma(s)$. The problem is visualized in Figure 4.5.

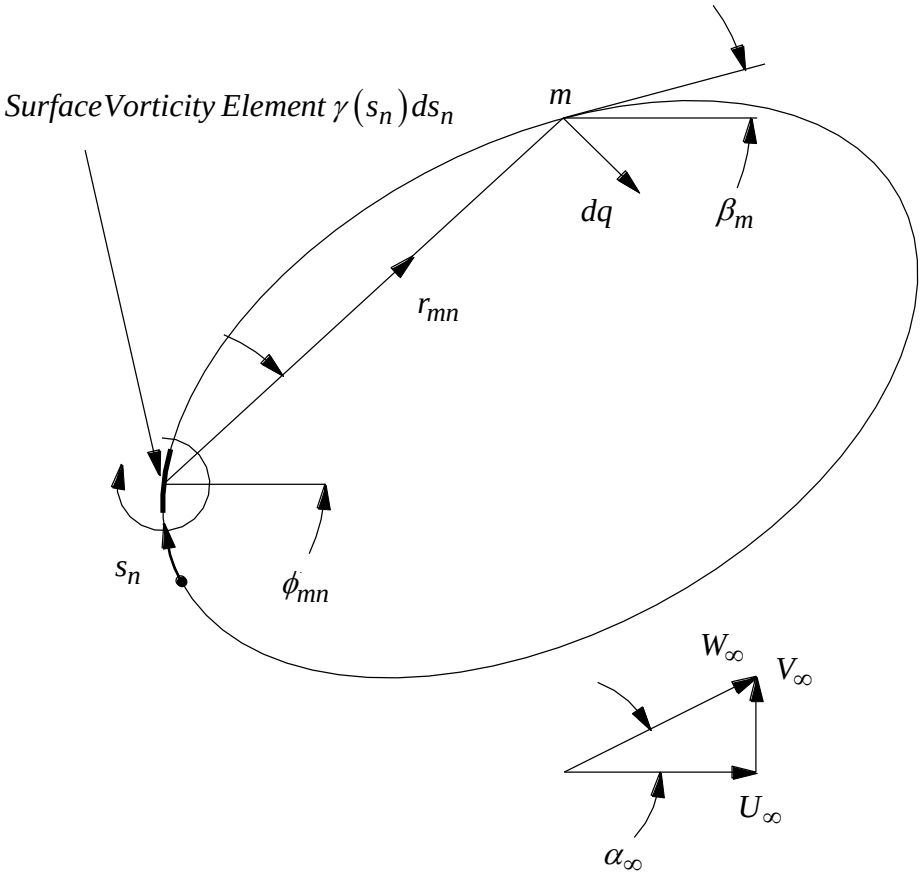

Figure 4.5 2D Surface Vorticity Sketch

The distance s, is measured from some datum point clockwise around the body. For airfoils, the datum is usually defined as the leading edge of the airfoil. The velocity induced at point m by a small line vortex element at n of strength Γ_n per unit length and of length dl_n is given by the Biot-Savart Law:

$$d\vec{v}_{mn} = \frac{\vec{\Gamma}_{mn} dl_n \times \vec{r}_{mn}}{4\pi r_{mn}^3} \tag{42}$$

For the two dimensional case the induced velocity is given as:

$$dq_{mn} = \frac{\gamma(s_n)ds_n}{2\pi r_{mn}} \tag{43}$$

Decomposing the induced velocity given in Equation (42) into its x and y directional components in terms of the geometric locations yields the following:

$$dU_{mn} = \left(\frac{y_m - y_n}{2\pi r_{mn}^2} \right) \gamma(s_n)\, ds_n \tag{44}$$

$$dV_{mn} = -\left(\frac{x_m - x_n}{2\pi r_{mn}^2} \right) \gamma(s_n)\, ds_n \tag{45}$$

The induced velocity parallel to m where the body slope is defined as β_n is given by:

$$dv_{s_{mn}} = \frac{1}{2\pi} \left[\frac{(y_m - y_n)\cos\beta_m - (x_m - x_n)\sin\beta_m}{(x_m - x_n)^2 + (y_m - y_n)^2} \right] \gamma(s_n)\, ds_n \tag{46}$$

The boundary condition of zero velocity flow on and parallel to the body surface at m (Dirichlet boundary condition) is expressed as follows:

$$-\frac{1}{2}\vec{\gamma}_m + \frac{1}{4}\iint_s \frac{\vec{i}_m \times \left(\left(\vec{\gamma}_m \times \vec{r}_{mn} \right) \times \vec{i}_m \right) dS}{r_{mn}^3} + \vec{i}_m \times \left(W_\infty \times \vec{i}_m \right) = 0 \tag{47}$$

For plane two-dimensional flows Equation (47) becomes:

$$-\frac{1}{2}\gamma(s_m) + \oint k(s_m, s_n)\gamma(s_n)\, ds_n + W_\infty \left(\cos\alpha_\infty \cos\beta_m + \sin\alpha_\infty \sin\beta_m \right) = 0 \tag{48}$$

Where the last term is the free stream velocity components parallel to the body at m and the coupling coefficients $k(s_m, s_n)$ is given as:

$$k(s_m, s_n) = \frac{1}{2\pi} \left[\frac{(y_m - y_n)\cos\beta_m - (x_m - x_n)\sin\beta_m}{(x_m - x_n)^2 + (y_m - y_n)^2} \right] \tag{49}$$

Equation (48) is to be true for all points on the body surface, therefore the body surface is approximated by a set number of straight line segments of length, Δs_n.

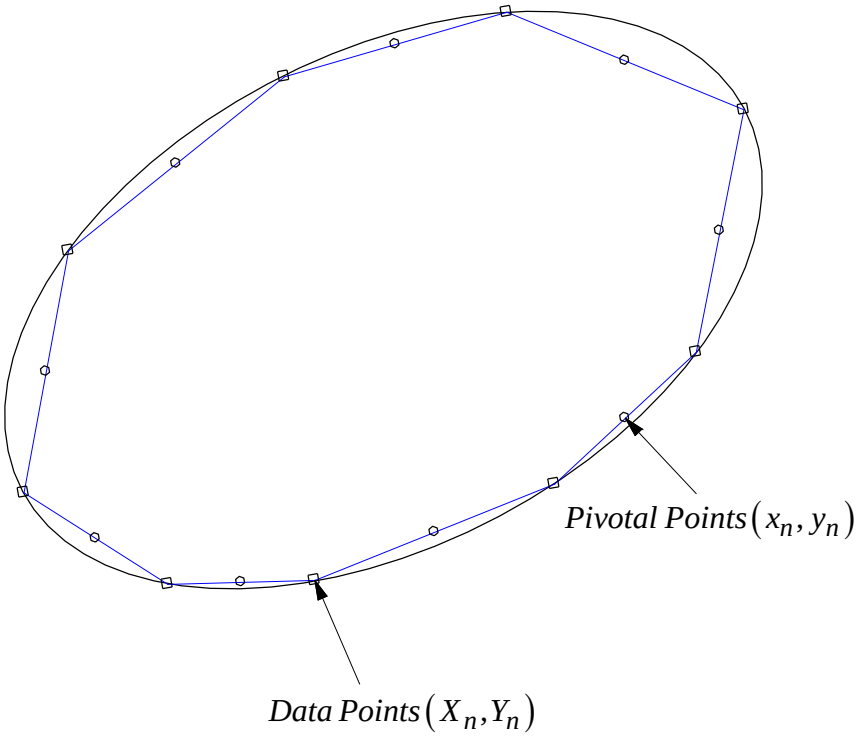

Figure 4.6 Surface Discretization Using Straight Line Elements

With this approximation Equation (48) can then be simplified as follows:

$$\sum_{n=1}^{M} K(s_m, s_n)\gamma(s_n) = -U_\infty \cos\beta_m - V_\infty \sin\beta_m \tag{50}$$

Where U_∞ and V_∞ are components of the free stream velocity W_∞ in the x and y directions, respectively. Also, the coupling coefficients are then given by:

$$K(s_m, s_n) = k(s_m, s_n)\Delta s_n \tag{51}$$

It should be noted that the term $-\frac{1}{2}\gamma(s_m)$ is combined with the coupling coefficients $K(s_m, s_n)$ such that:

$$K(s_m, s_n) = -\frac{1}{2} + K'_{mm} \tag{52}$$

Where $K'_{mm} \cdot \gamma(s_m)$ is the self-induced velocity of the m[th] element. This is necessary because Equation (49) is finite but indeterminate when $m = n$. Therefore the self-induced coupling coefficient is then:

$$K'_{mm} = \frac{\Delta s_m}{2\pi} Lim_{s_m \to s_n} \left[\frac{(y_m - y_n)\cos\beta_m - (x_m - x_n)\sin\beta_m}{(x_m - x_n)^2 + (y_m - y_n)^2} \right] \tag{53}$$

For a straight-line segment, this value is zero, but for a curved body surface there will always be a net induced velocity parallel to the body. The self-induced velocity due to a curved surface is given by:

$$K'_{mm} = \frac{\Delta s_m}{4\pi} \left[\frac{-\frac{d^2 y_m}{dx_m^2}}{\left(1 + \left(\frac{dy_m}{dx_m}\right)^2\right)^{\frac{3}{2}}} \right] = \frac{\Delta s_m}{4\pi r_m} \approx -\frac{\Delta\beta_m}{4\pi} \tag{54}$$

From Kelvin's theorem, the net circulation on the interior of a surface induced by a vorticity element should be zero. To account for this, Lewis[37] applies a so called back diagonal correction to the coupling coefficient matrix, namely:

$$K(s_{M+1-m}, s_m) = -\frac{1}{\Delta s_{M+1-m}} \sum_{\substack{n=1 \\ n \neq M+1-m}}^{M} K(s_n, s_m)\Delta s_n \tag{55}$$

Once this correction is applied, the Kutta condition must be applied to the trailing edge of the section. This is applied by the following equation:

$$\gamma(s_{te}) = -\gamma(s_{te+1}) \tag{56}$$

Once these corrections are applied the surface vorticity can be determined by solving Equation (50) for $\gamma(s)$. The pressure coefficient is then calculated by:

$$C_P = 1 - \left[\frac{\gamma(s)}{W_\infty} \right]^2 \tag{57}$$

4.2.1.3 Axisymmetric Surface Vorticity Formulation

The three dimensional axisymmetric vorticity follows from the two dimensional formulation, with the exception of the formulation of the coupling coefficients. A sheet of ring vorticity now approximates the body. The vortex ring is modeled by the Biot-Savart law is shown in Figure 4.7.

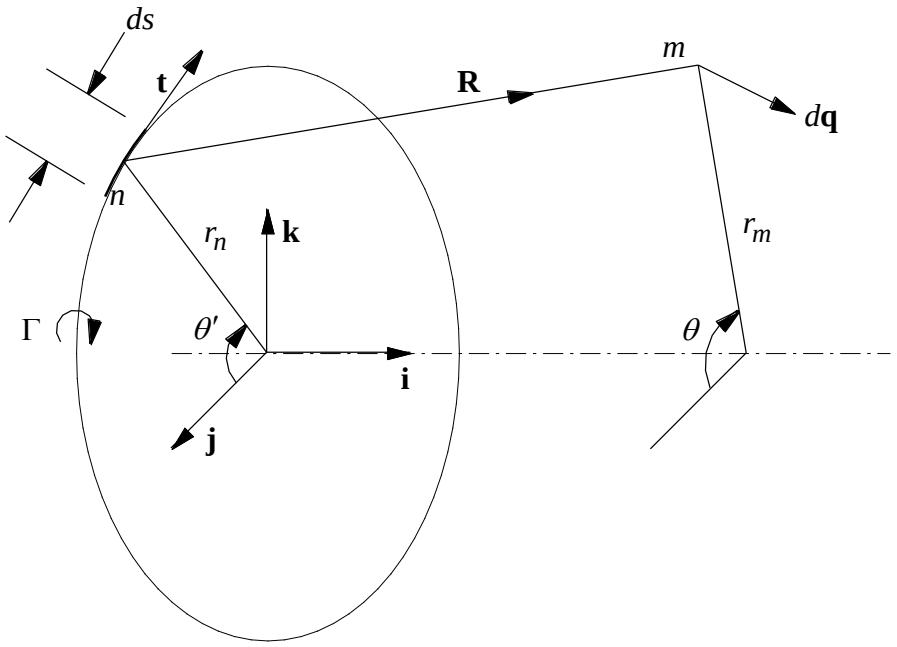

Figure 4.7 Vortex Ring Modeling by Biot-Savart Law

The induced velocity components at point m due to a unit ring vortex at n of strength $\Gamma = 1$ are given by:

$$u_{mn} = \frac{1}{4\pi}\int_0^{2\pi}\left[\frac{r_n - r_m\cos(\theta-\theta')}{\left[(x_m-x_n)^2 + r_m^2 + r_n^2 - 2r_m r_n\cos(\theta-\theta')\right]^{\frac{3}{2}}}\right]d\theta' \tag{58}$$

$$v_{mn} = \frac{1}{4\pi}\int_0^{2\pi}\left[\frac{(x_m-x_n)\cos\theta'}{\left[(x_m-x_n)^2 + r_m^2 + r_n^2 - 2r_m r_n\cos(\theta-\theta')\right]^{\frac{3}{2}}}\right]d\theta' \tag{59}$$

The coupling coefficients for the axisymmetric case are then calculated by:

$$\tilde{K}\left(s_m, s_n\right) = u_{mn}\cos\beta_m + v_{mn}\sin\beta_m \tag{60}$$

The equations for the induced velocities, u_{mn} and v_{mn} can be simplified with the use of elliptic integrals of the first and second kind. The induced velocities are then given by:

$$u_{mn} = -\frac{1}{2\pi r_n\sqrt{\left[x^2+\left(r+1\right)^2\right]}}\left[K\left(k\right)-\left[1+\frac{2\left(r-1\right)}{x^2+\left(r-1\right)^2}\right]E\left(k\right)\right] \tag{61}$$

$$v_{mn} = -\frac{\frac{x}{r}}{2\pi r_n\sqrt{\left[x^2+\left(r+1\right)^2\right]}}\left[K\left(k\right)-\left[1+\frac{2r}{x^2+\left(r-1\right)^2}\right]E\left(k\right)\right] \tag{62}$$

Where K(k) and E(k) are elliptic integrals of the first and second kind and

$$k = \sqrt{\frac{4r}{x^2+\left(r+1\right)^2}} \tag{63}$$

$$x = \frac{x_m - x_n}{r_n} \tag{64}$$

$$r = \frac{r_m}{r_n} \tag{65}$$

Representing the body as by M discrete vortex rings, the potential flow about the axisymmetric body is given as follows

$$\sum_{n=1}^{M}\tilde{K}\left(s_m, s_n\right)\gamma\left(s_n\right)\Delta s_n = -W\cos\beta_m \tag{66}$$

By absorbing the element length into the coupling coefficient, Equation (66) becomes a set of linear equations, namely:

$$\begin{pmatrix} K_{11} & K_{12} & \dots & K_{1M} \\ K_{21} & K_{22} & \dots & K_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ K_{M1} & K_{M2} & \dots & K_{MM} \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_M \end{pmatrix} = \begin{pmatrix} rhs_1 \\ rhs_2 \\ \vdots \\ rhs_M \end{pmatrix} \tag{67}$$

Where:

$$K\left(s_m, s_n\right) = \tilde{K}\left(s_m, s_n\right)\Delta s_n \tag{68}$$

The self-induced coupling coefficient for a body of revolution is given by Lewis[37] as:

$$K\left(s_m, s_m\right) = -\frac{1}{2} + \frac{\Delta s_m}{4\pi R_m} - \frac{\Delta s_m}{4\pi R_m}\left[\ln\frac{8\pi r_m}{\Delta s_m} - \frac{1}{4}\right]\cos\beta_m \tag{69}$$

Where R_m and r_m are given in Figure 4.8.

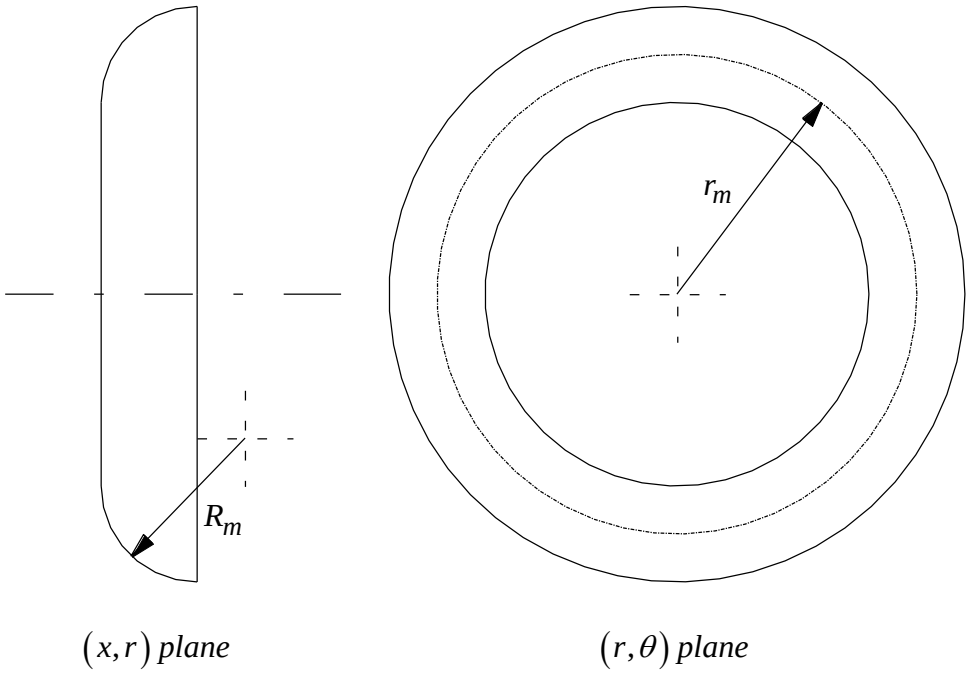

Figure 4.8 Double Curvature of a Ring Vortex Surface Element

The duct and hub are modeled in the same fashion and the potential flow system of equations given by:

$$\begin{pmatrix} K_{Hub} & K_{DuctHub} \\ K_{HubDuct} & K_{Duct} \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \vdots \\ \gamma_m \end{pmatrix} = \begin{pmatrix} rhs_1 \\ \vdots \\ rhs_m \end{pmatrix} \tag{70}$$

The back diagonal correction is still applied to the duct influence coefficients, K_{Duct}. The Kutta condition is also applied, which results in a reduction the number of equations in the system of equations. The pressure coefficient can then be calculated by the equation given in the preceding section, Equation (57).

4.2.2 Rotor and Wake Model

A more detailed derivation of the rotor and wake model is given by Drela[38]. The rotor is modeled using blade lifting line theory with radial variation in the loading of the rotor. At any location along the blade where the circulation changes by $\Delta\Gamma$, there will be a helical vortex filament shed into the flow with strength, $-\Delta\Gamma$. These vortex filaments can be translated into tangential and meridional vortex sheets of strengths gamma and γ_m, respectively. These vortex strengths are the circulation per unit length in the two directions. The vortex strengths are given by:

$$\gamma_m = -\frac{B\Delta\Gamma}{2\pi r} \tag{71}$$

$$\gamma = -\frac{B\Delta\Gamma}{2\pi r}\frac{W_\theta}{W_m} \tag{72}$$

These shed vortex sheets make up the rotor wake. The strength of the wake vortex sheet is related to the circulation and the total enthalpy jump across the sheets by:

$$\gamma_i = \frac{1}{V_{m_i}}\left[-\frac{1}{2}\left(\frac{1}{2\pi r}\right)^2 \Delta\left(\Gamma^2\right) + \Delta\left(H_i\right)\right] \tag{73}$$

Where Γ, is the blade circulation and the total enthalpy jump across the vortex sheet is given by:

$$\Delta H_i = \Omega \frac{B \Delta \Gamma}{2\pi} \tag{74}$$

The blade can then be discretized into M number of blade sections and the blade forces can be computed for that section using sectional properties of the airfoils. The circulation for the blade segment is defined as follows:

$$\Gamma = \frac{1}{2} W c c_l \tag{75}$$

The sectional lifting coefficient, c_l, is dependent on the local angle of attack of the section. Therefore the blade sectional properties are important. Figure 4.9 shows the geometrical definitions for the blade section.

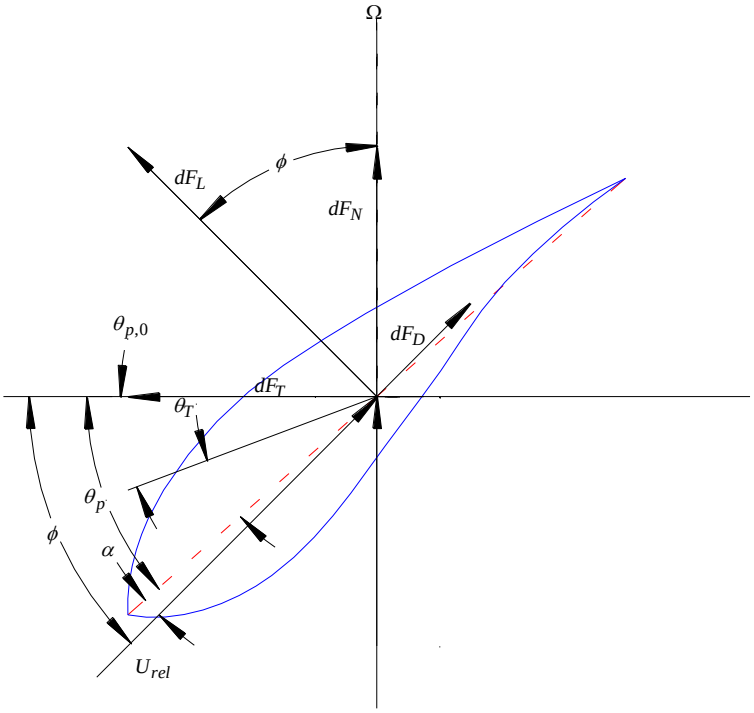

Figure 4.9 Blade Section Geometry Sketch

From Figure 4.9 the following relationships can be determined:

$$\phi = \tan^{-1}\left(\frac{U_{rel}}{\Omega r}\right) \tag{76}$$

$$\alpha = \phi - \theta \tag{77}$$

With these relationships, blade sectional data, blade pitch and chord distributions and Equation (75), the blade circulation as a function of radial position can be calculated.

Once the blade circulation is known, the strengths of the wake vortex sheets can be calculated and included in the analysis. The wake is modeled as vortex ring sheets, similar to that of the duct and hub, with the exception that the strengths of the vortex rings are known and their influence on the duct and hub can be included in the calculation of the duct and hub vorticity values. When the wake influence is included, Equation (70) becomes:

$$\begin{pmatrix} K_{Hub} & K_{DuctHub} \\ K_{HubDuct} & K_{Duct} \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \vdots \\ \gamma_m \end{pmatrix} + K_{wake}\gamma_{wake} = \begin{pmatrix} rhs_1 \\ \vdots \\ rhs_m \end{pmatrix} \tag{78}$$

The surface vorticity values on the duct and hub can then be computed incorporating the influence of the rotor and wake.

The effect of the blade drag can also be taken into account by approximating the blade as a source sheet. The strength of the source is given by:

$$\sigma = \frac{B}{4\pi r} W c c_d \tag{79}$$

The source sheet can be incorporated into Equation (78) as follows:

$$\begin{pmatrix} K_{Hub} & K_{DuctHub} \\ K_{HubDuct} & K_{Duct} \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \vdots \\ \gamma_m \end{pmatrix} + K_{wake}\gamma_{wake} + K_{blade}\sigma_{blade} = \begin{pmatrix} rhs_1 \\ \vdots \\ rhs_m \end{pmatrix} \tag{80}$$

4.2.3 DAWT Potential Flow Solution

The methodology is programmed in MATLAB[39] computing software. The model flow chart is shown in Figure 4.10. The MATLAB code is included in Appendix C.

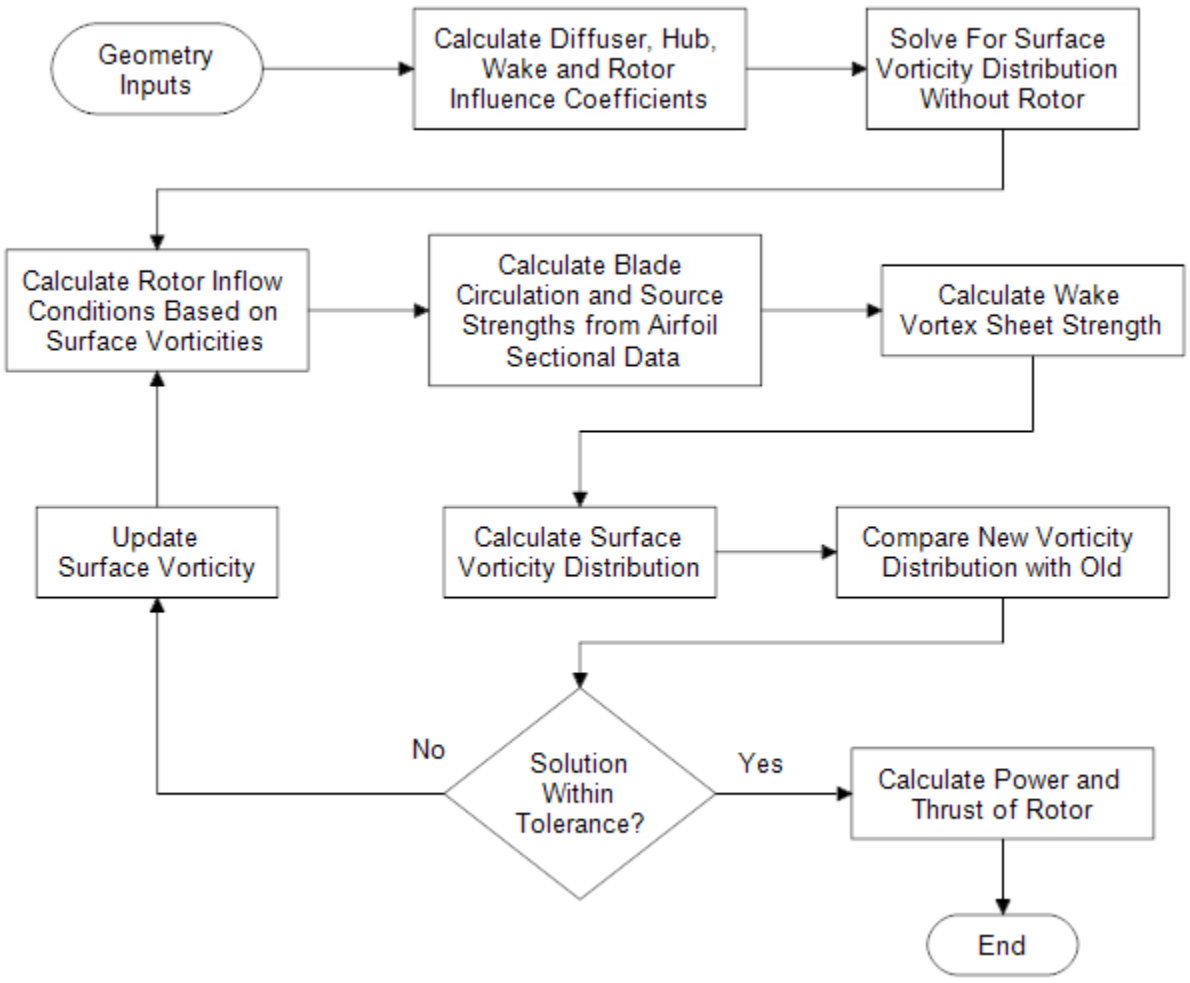

Figure 4.10 Solution Flow Chart

The duct, hub and rotor geometry are known along with the initial operating conditions such as the free stream velocity, global pitch angle (θ_0), rotor RPM, air density and altitude. The body influence coefficients are constant and can be calculated and stored for later use. To calculate the body influence coefficients the geometry is approximated by linear panel elements. The panel elements are defined clockwise from the section leading edge and ending at the same point forming a closed body. The length of the panel is defined by:

$$\Delta s_n = \sqrt{\left(x_{n+1} - x_n\right)^2 + \left(y_{n+1} - y_n\right)^2} \tag{81}$$

The panel slope is calculated by:

$$\tan \beta_n = \frac{y_{n+1} - y_n}{x_{n+1} - x_n} \tag{82}$$

The vortex rings are applied at pivotal points on the panel. These pivotal points are defined as follows:

$$x_{pivotal} = \frac{1}{2}\left(x_{n+1} + x_n\right) \tag{83}$$

$$y_{pivotal} = \frac{1}{2}\left(y_{n+1} + y_n\right) \tag{84}$$

The coupling coefficients are then calculated using Equation (78). These coupling coefficients are then stored for use in subsequent calculations.

The wake geometry assumed to expand and contract parallel to the duct surface and the shed vortices from the blade are assumed to be at the edges of the blade section. The wake is then approximated similarly to the duct and hub geometry with initial constant ring vortex strength of zero. The wake is assumed to extend 8 rotor diameters downstream of the rotor. After this it is assumed that the wake influence is negligible. This is to approximate the rotor wake extending to infinity downstream of the rotor. The influence of the wake on the duct, hub and rotor plane is calculated and the influence coefficients are stored for later use.

The rotor blade is divided into blade segments and the pitch and chord distributions are defined input parameters of the blade. The initial velocity at the rotor plane is calculated using the coupling coefficients of the duct, hub and wake to determine the velocity distribution at the rotor plane. The pivotal points at the rotor plane are at the center points of the blade sections. These induced velocities are then used to determine the circulation for each blade section using the blade section parameters. The strengths of the blade section shed vortices can then be calculated using the circulation of the blade segments. The wake and rotor effects are then updated and new surface vorticity strengths are calculated and the velocity at the rotor plane can then be

updated. This process is repeated until the surface vorticity strengths converge within a desired tolerance. Figure 4.11 shows a sketch of the discretized model.

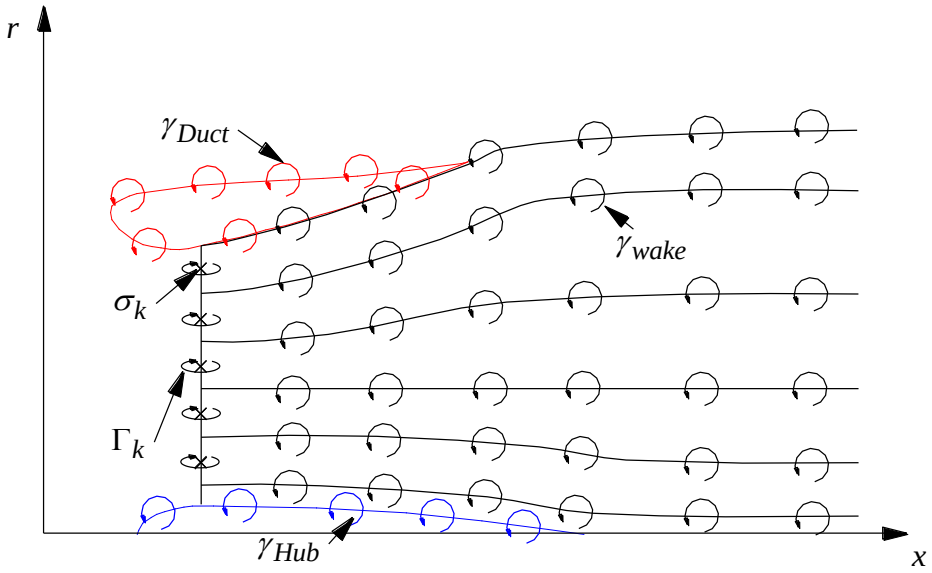

Figure 4.11 Vortex Sheet Discretization Sketch

5 Validation

When applying a mathematical model to a problem, it is important to test the model to make sure the calculations are correct and validate the model with experimental data. This section deals with the application of the proposed theory outlined above. The theory is compared to a known BEM computer code distributed by NREL called WT_perf [40]. The power output as well as the thrust is compared with the current method under various operating conditions. The proposed method is also compared with experimental data given in Reference 31.

5.1 Duct and Hub Without Rotor

Reference 37 gives a validation study of the axisymmetric surface vorticity method using a NACA 66_2-015 annular airfoil. Figure 5.1 shows the cross section of the annular airfoil. Normalized points are given in Appendix A.

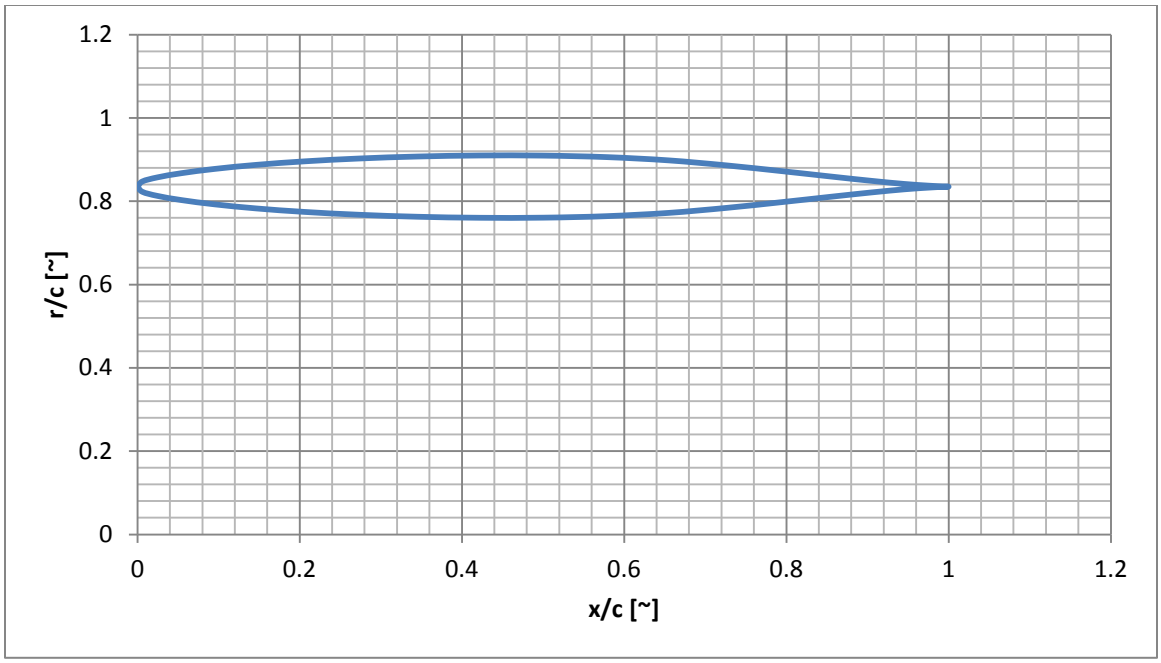

Figure 5.1 NACA 66_2-015 Annular Airfoil

Figure 5.2 shows the results of the surface vorticity method in comparison to the experimental data given in Reference 37.

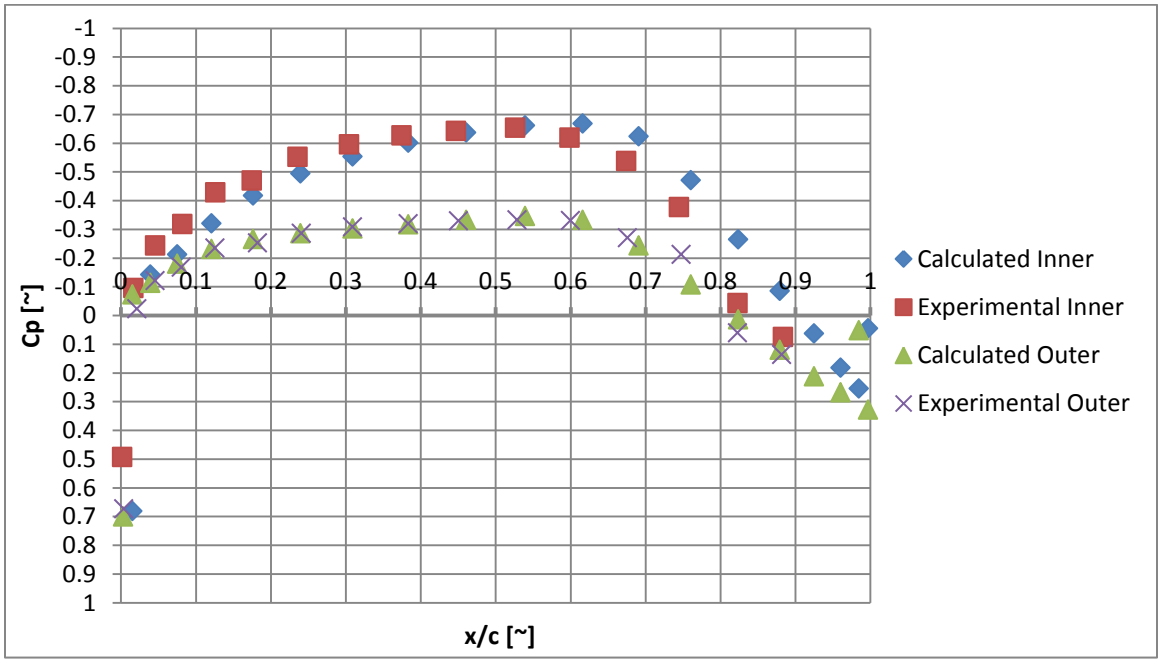

Figure 5.2 Surface Vorticity Method Compared to Experimental for Annular Airfoil

As Figure 5.2 shows, the surface vorticity method predicts the surface pressure coefficient distribution well for inner and outer annular airfoil surfaces.

5.2 Bare Rotor Comparison with BEM

The current methodology should be applicable to bare rotors, provided the influence of the duct is minimized to a point where it can be neglected. Therefore, the current method is used to calculate the performance of a bare rotor. The calculated performance is then compared to the calculated performance using a well-known bare turbine analysis code (WT_perf[40]).

5.2.1 Blade Geometry

The blade twist and chord distributions are taken from the WindPACT 1.5 MW turbine example provided with the WT_perf software. The rotor parameters are given in Table 5.1.

Table 5.1 Blade Geometric Parameters

Parameter	Value
N_{blade} [~]	3
R_{tip} [m]	35
R_{hub} [m]	1.75
Airfoil	S825

For simplicity, the blade cross section is assumed to be a constant S825 airfoil. The WT_perf inputs are given in Appendix B. The blade pitch and chord distributions are shown in Figure 5.3.

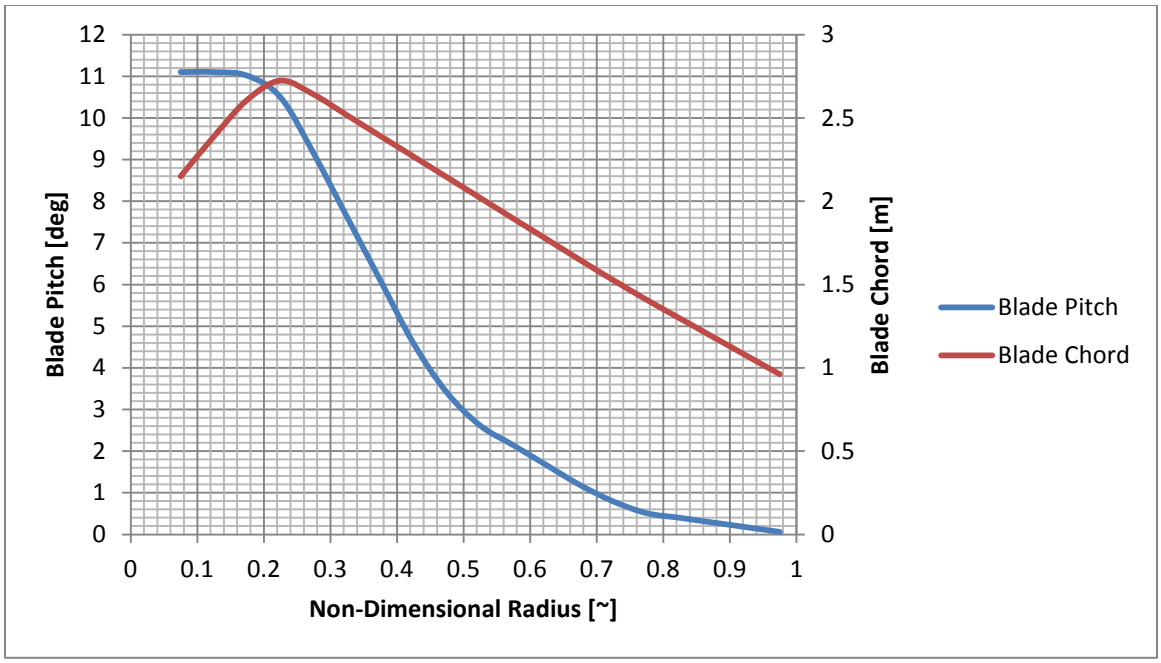

Figure 5.3 Bare Turbine Blade Pitch and Chord Distributions

The airfoil section for the S825 airfoil is shown in Figure 5.4.

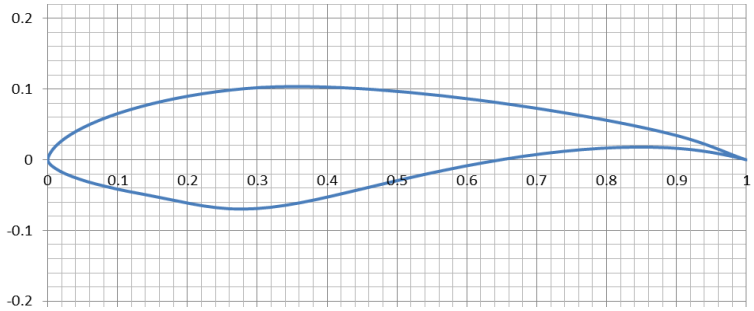

Figure 5.4 Bare Turbine Blade S825 Airfoil Section

The lift and drag characteristics of the S825 airfoil are shown in Figure 5.5. The sectional lift and drag coefficients are obtained using the XFOIL program[41] and then the angle of attack range is extended to +/- 180 degrees.

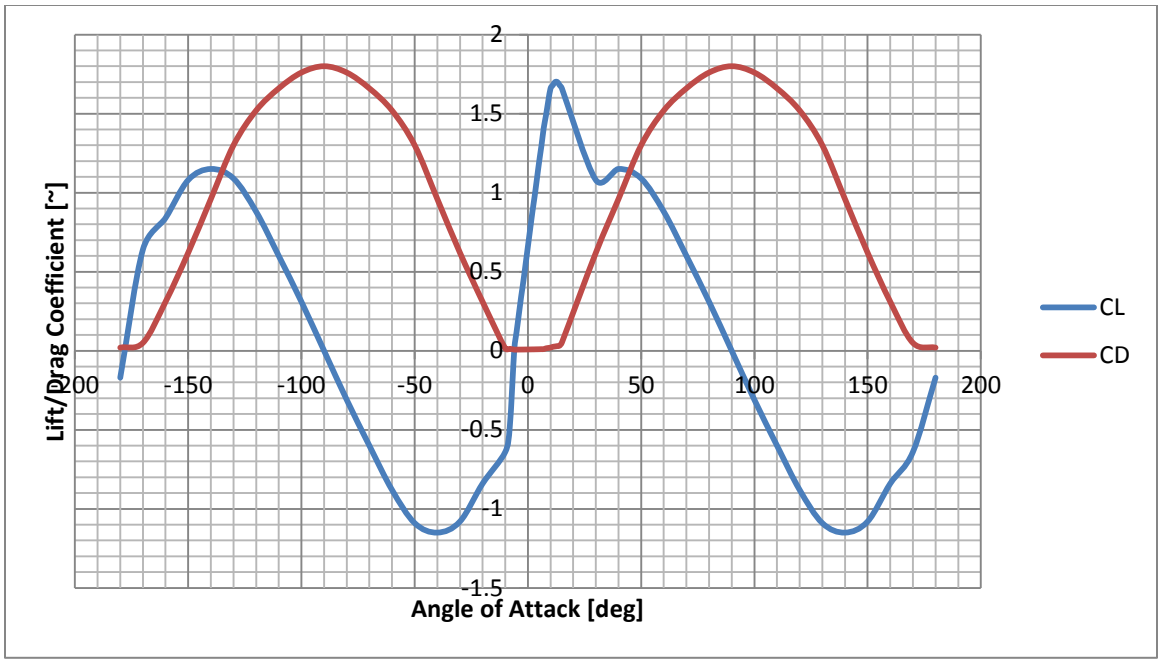

Figure 5.5 S825 Sectional Aerodynamic Characteristics

5.2.2 Duct and Hub Geometry

To simulate a bare turbine with the current methodology, the duct radius is assumed to be very large (200 meters) such that there is minimal effect at the rotor plane. The hub will be modeled to approximate the correct hub geometry. The duct radius is assumed to be twice the rotor diameter. The duct is assumed to be an annular wing with a NACA 0012 airfoil cross section to reduce any duct influences on the rotor plane. This is shown in Figure 5.6.

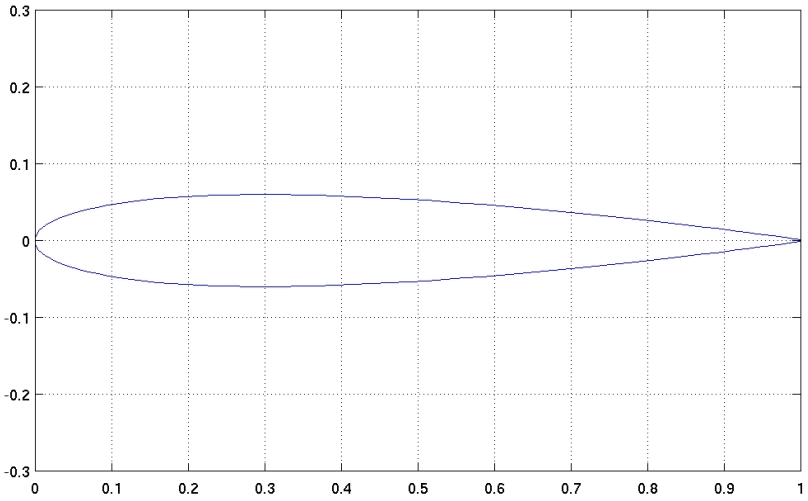

Figure 5.6 NACA 0012 Airfoil

The approximated hub geometry is shown in Figure 5.7.

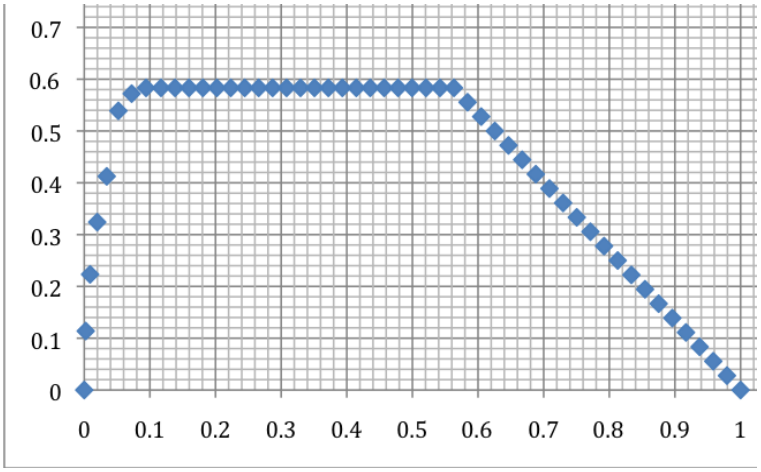

Figure 5.7 Bare Turbine Hub Approximation

5.2.3 Results

Figure 5.8 and Figure 5.9 show the comparison between a BEM code and the current method for a constant rotor rotational speed of 20 RPM and a pitch angle of 4 degrees and 8 degrees, respectively.

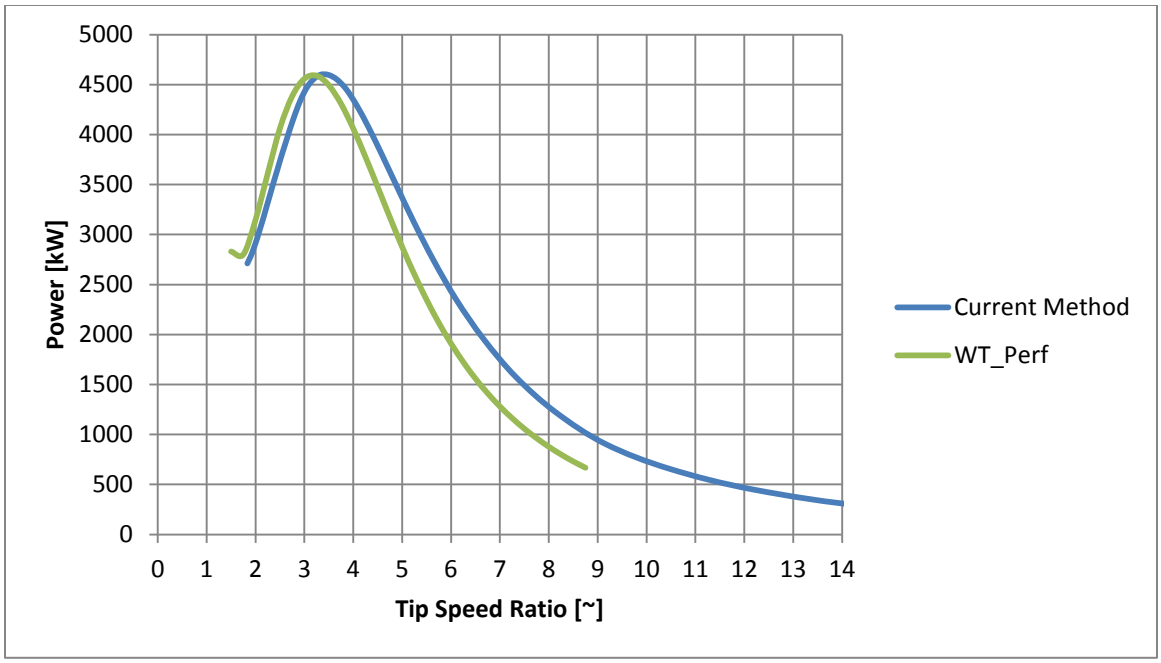

Figure 5.8 Power Curve for a Constant Pitch Angle of 4 Degrees at 20 RPM

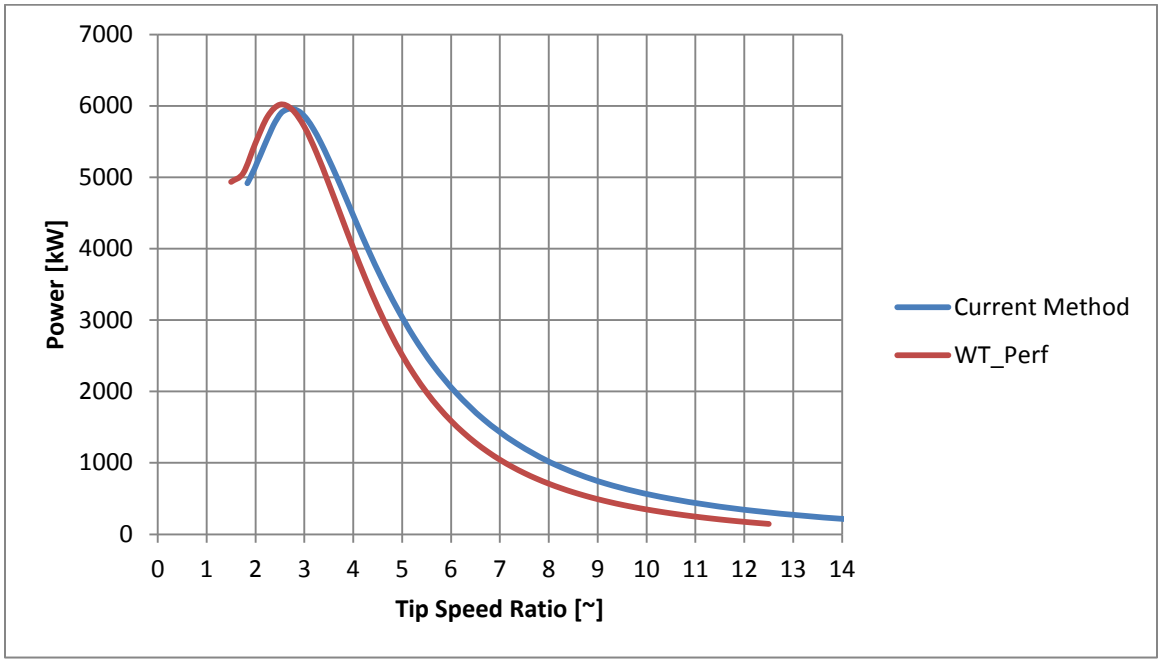

Figure 5.9 Power Curve for a Constant Pitch Angle of 8 Degrees at 20 RPM

As Figure 5.8 and Figure 5.9 show, there is good agreement in the power predicted for the rotors and the maximum power produced as well as the tip speed ratio at maximum power production at lower tip speed ratios. After the peak in the power generated, the power predicted by the current method deviates from the power predicted by WT_Perf. The peak power prediction deviates less than 1% between the two methods. The current method under predicts the power about 5-10% for lower tip speed ratios and over estimates the power production by about 40% at a tip speed ratio of 8 when compared to the data calculated by WT_Perf.

The discrepancy could be caused by the breakdown of the momentum theory at higher values of axial induction factors. As the rotor operates at higher tip speed ratios, the incoming air flow sees a decreasingly permeable disk. As the flow sees an increasingly solid disk, it starts to spread out radially to go around the disc and separates at the blade tips causing a region of low static pressure to develop behind the rotor. The area that does make it through the rotor enters into the low static pressure region where the air speed is low. The only way for the rotor wake to increase the static pressure to that of the far-field atmospheric pressure is to mix with the separated flow off of the rotor edge. The separated flow from the rotor edge develops into an unstable shear layer between the free-stream and wake that breaks down to the turbulence that causes that mixing in the wake. This is known as the turbulent wake state of the rotor operation. Figure 5.10 shows how the rotor states are related to the axial induction factor, a.

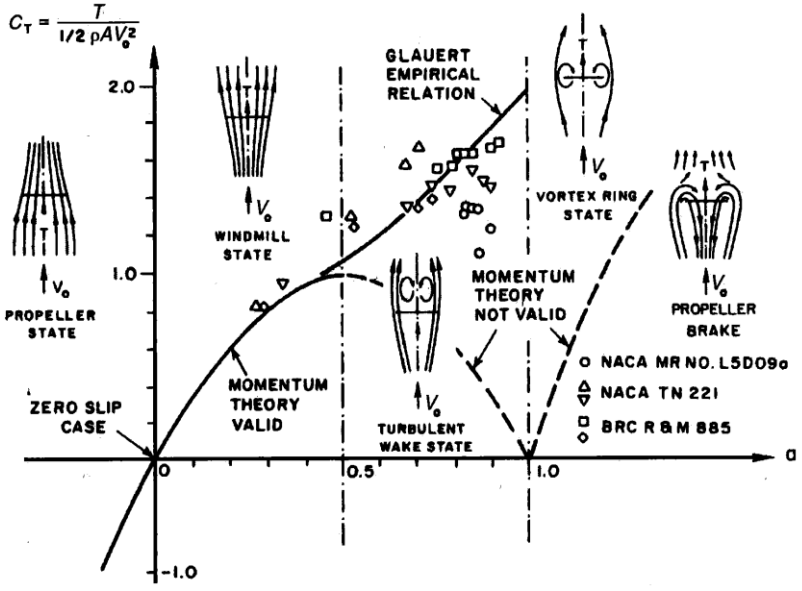

Figure 5.10 Thrust Coefficient and Rotor States in Relation to the Axial Induction Factor[42]

As Figure 5.10 shows, there is an empirical relation developed by Glauert, used in most BEM codes including WT_Perf. The current methodology does not use this correction and this could be the cause for the discrepancy at higher tip speed ratios. The current method may not capture the turbulent wake operation of the rotor as Glauert's empirical relationship, but the current method may not be as far off from the actual operation as the comparison would seem. Figure 5.11 shows a comparison of measured wind turbine data compared with BEM predictions.

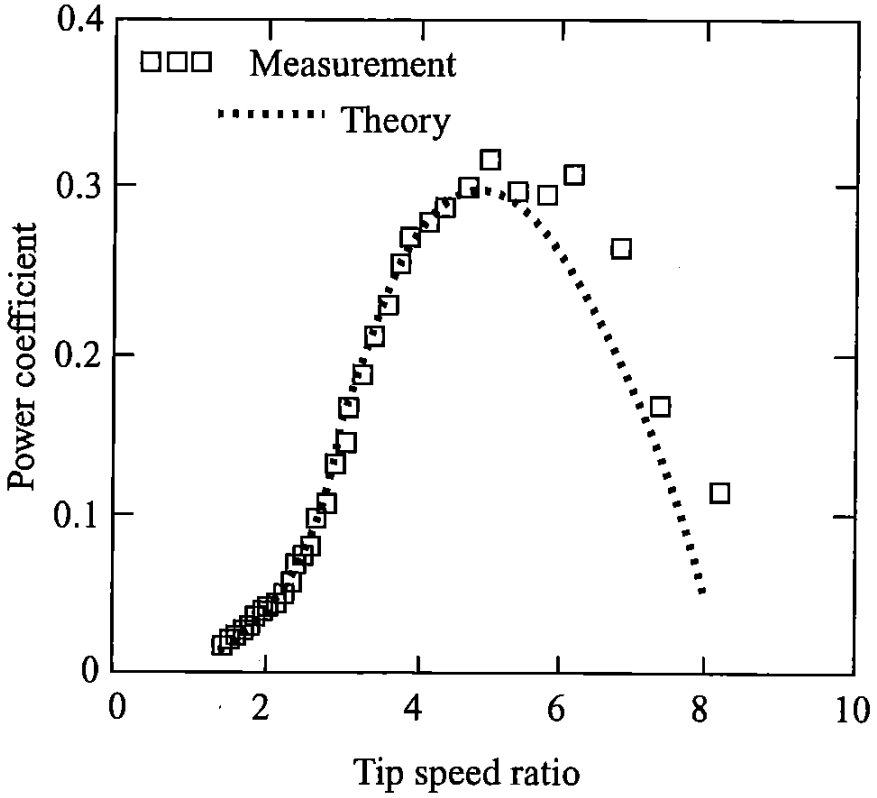

Figure 5.11 Comparison of Measured and Theoretical Performance Curve[2]

As Figure 5.11 shows, the BEM prediction seems to lie on top of the measured data points until the rotor reaches maximum power, then the BEM theory under predicts the measured power at higher tip speed ratios. This does not suggest that the current method is better than BEM at predicting the power in the turbulent wake state of the rotor operation, but it does suggest that there should be further investigation into the validity of the current method at higher tip speed ratios into the turbulent wake state operation of rotors.

5.3 Experimental Results

A comparison between the current analysis method and experimental data is presented in this section. The geometry and experimental results are given in Reference 31.

5.3.1 Duct and Hub Geometry

The duct and hub geometry is given in Reference 31 and is shown in Figure 5.12.

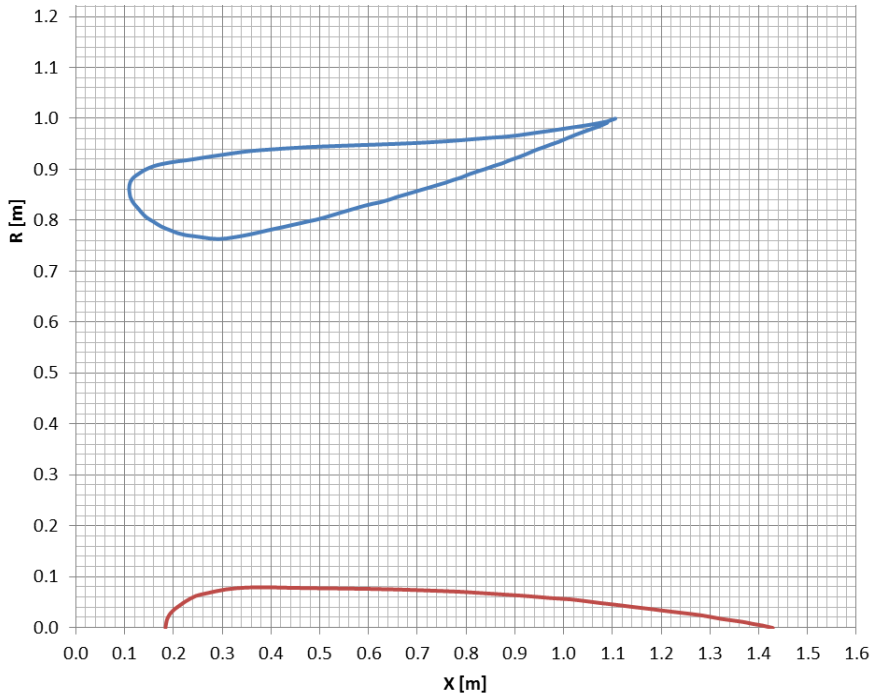

Figure 5.12: Duct and Hub Geometry

5.3.2 Blade Geometry

Figure 5.13 shows the blade pitch and chord distributions for the linearized blade.

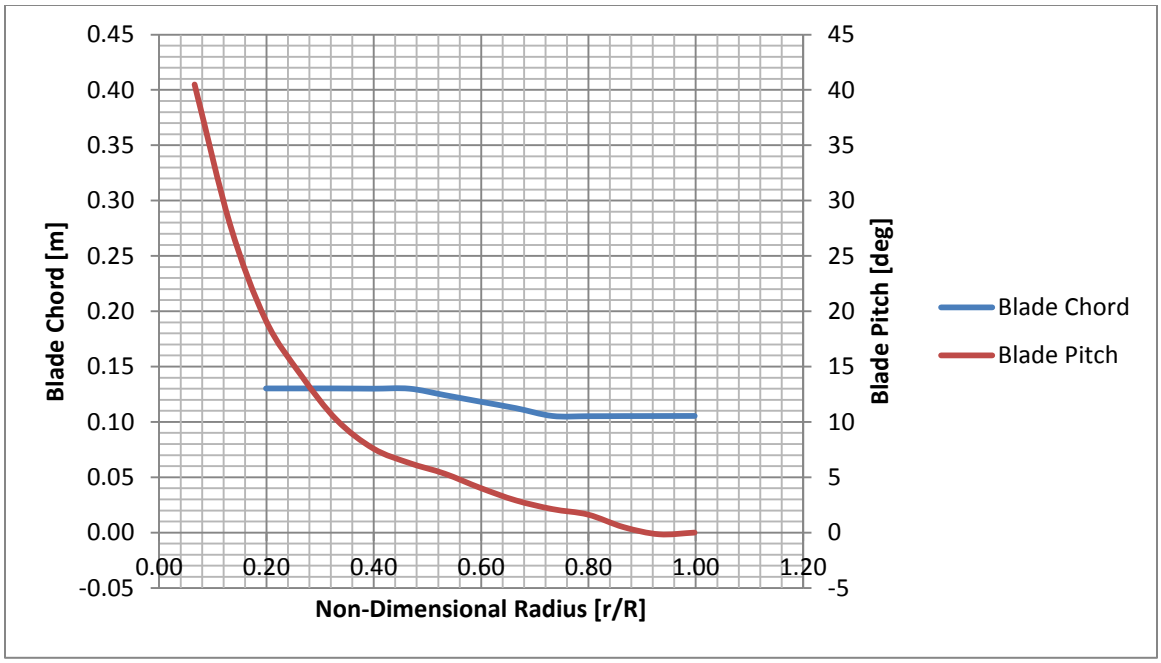

Figure 5.13 Linearized Blade Pitch and Chord Distribution

Figure 5.14 shows the pitch and chord distributions for the optimal blade geometry given in Reference 31.

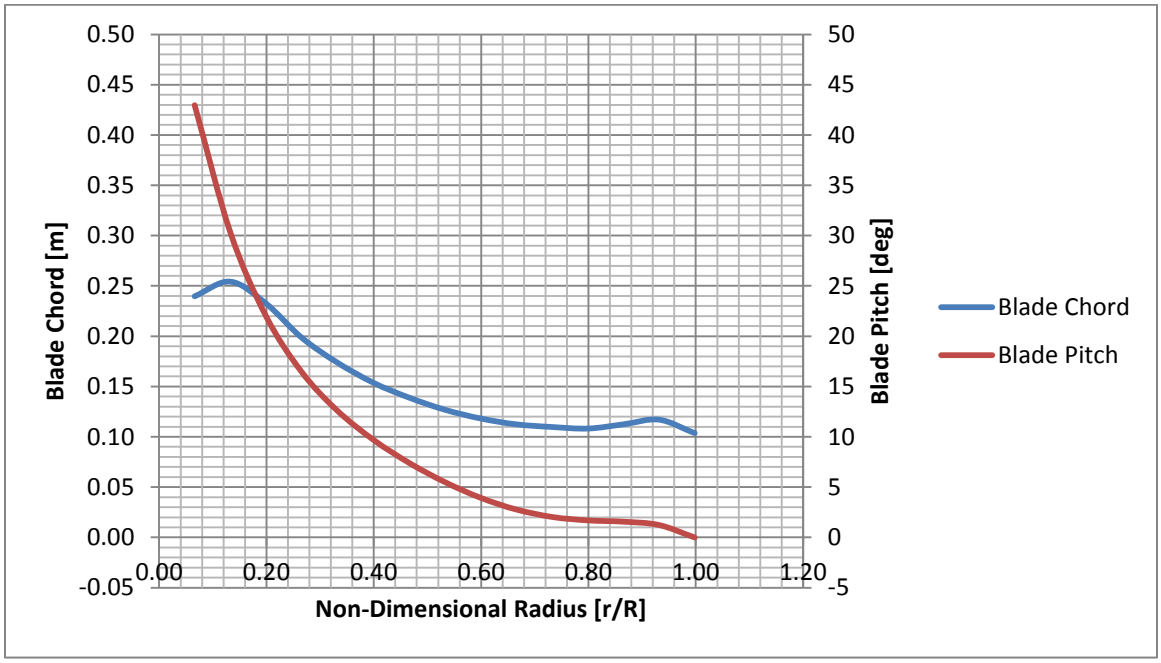

Figure 5.14 Optimal Blade Pitch and Chord Distribution

Table 5.2 shows the rotor parameters for both the linearized blade and optimal blade geometries.

Table 5.2 DAWT Blade Geometric Parameters

Parameter	Value
N_{blades} [~]	3
R_{tip} [m]	0.75
R_{hub} [m]	0.20
Airfoil	NACA 2207

The airfoil used in the by these blades is given in Reference 31 as the NACA 2207 airfoil. This cross section is shown in Figure 5.15.

Figure 5.15 NACA 2207 Airfoil Section

The NACA 2207 airfoil section is analyzed using XFOIL. The 2-D analysis is extended from -180 degrees to 180 degrees and used in the current method rotor simulation code. This is given in Figure 5.16.

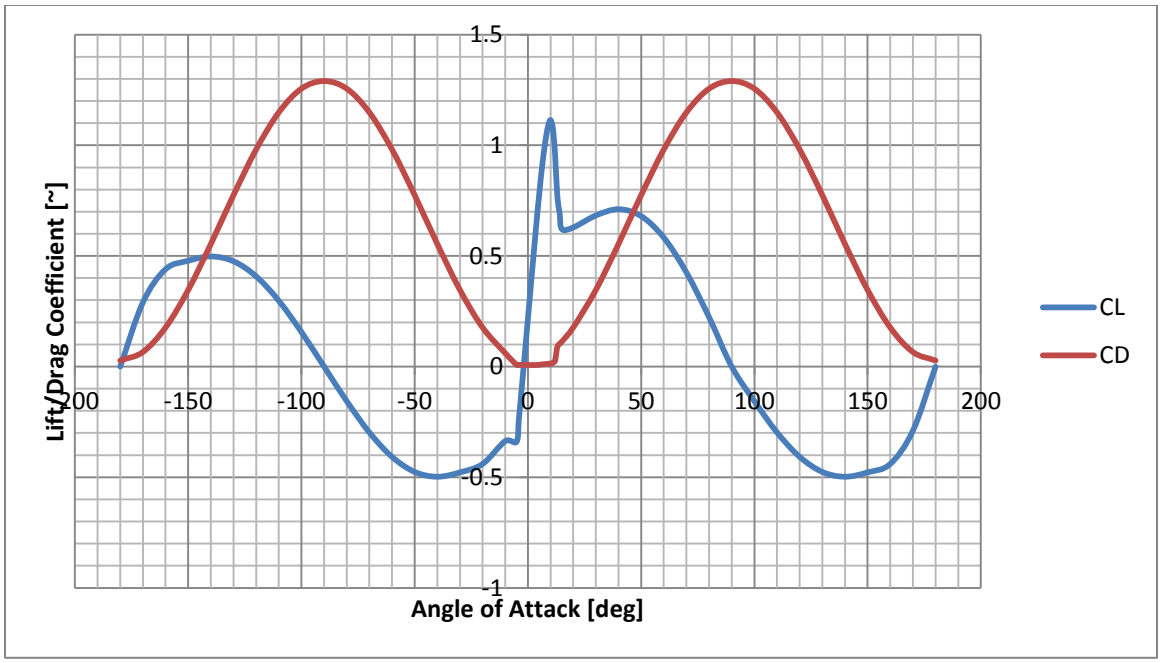

Figure 5.16 NACA 2207 Sectional Aerodynamic Characteristics

5.3.3 Results

A wake with 250 discrete segments is used in the subsequent analysis. Figure 5.17 shows the experimental results given in Reference 31 for the linearized blade geometry.

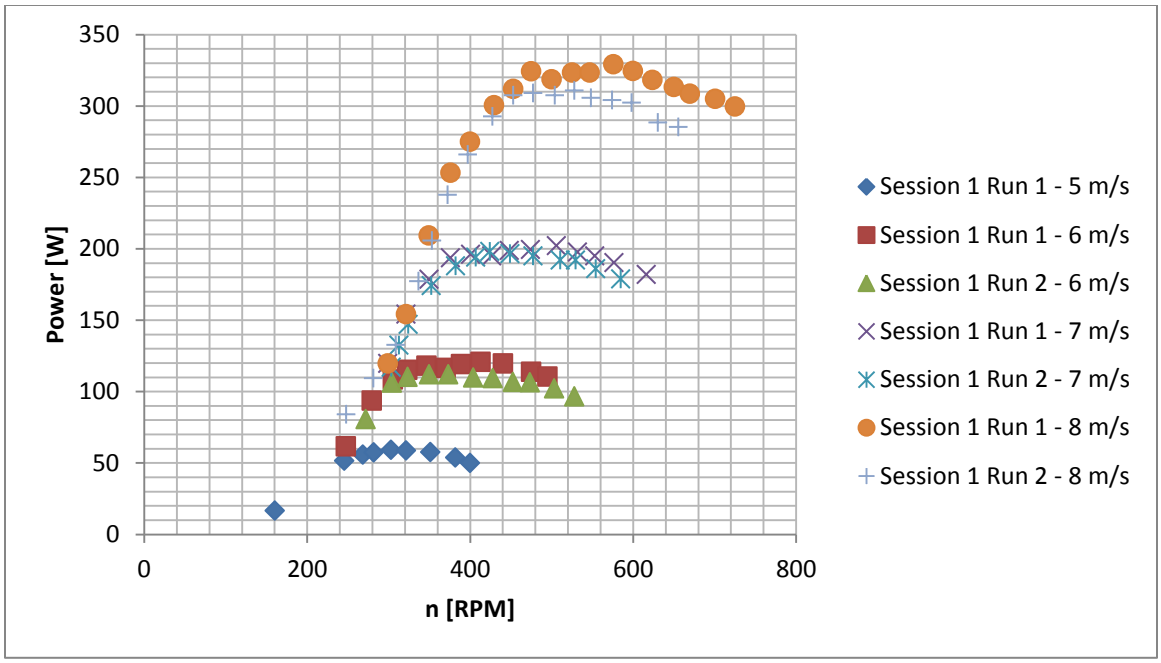

Figure 5.17 Experimental Data for the Linearized Blade Geometry[31]

Figure 5.18, Figure 5.19, Figure 5.20 and Figure 5.21 show the comparison between the current method and the experimental data for wind speeds of 5, 6, 7 and 8 m/s at various RPM.

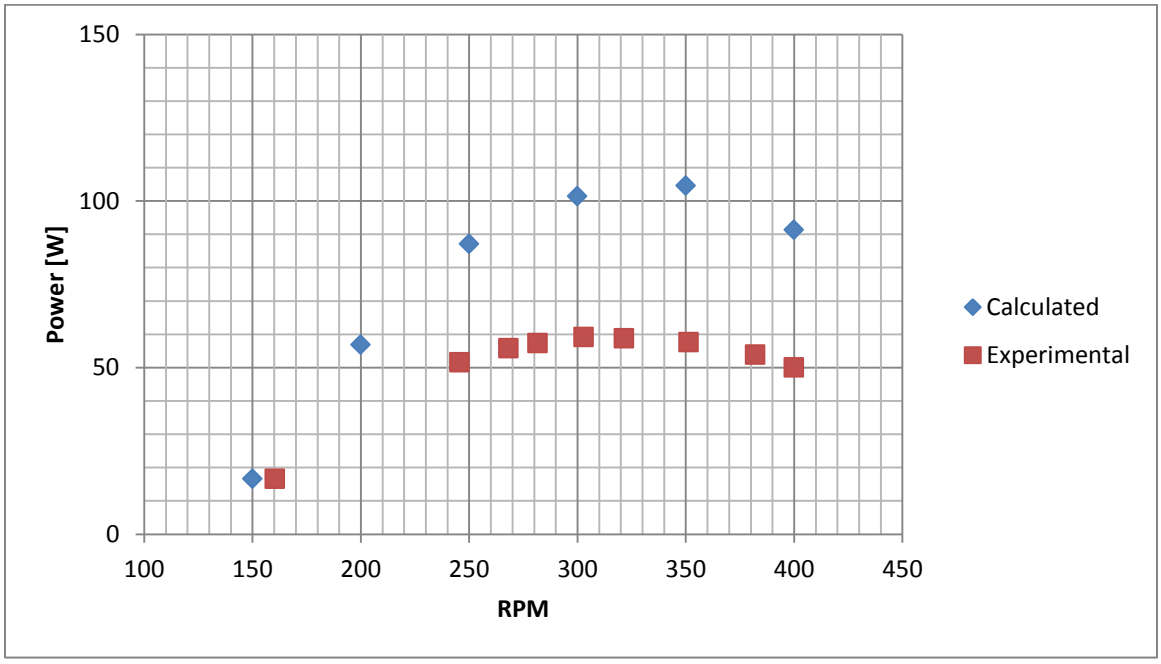

Figure 5.18 Experimental and Calculated Power at 5 m/s for Linearized Blade

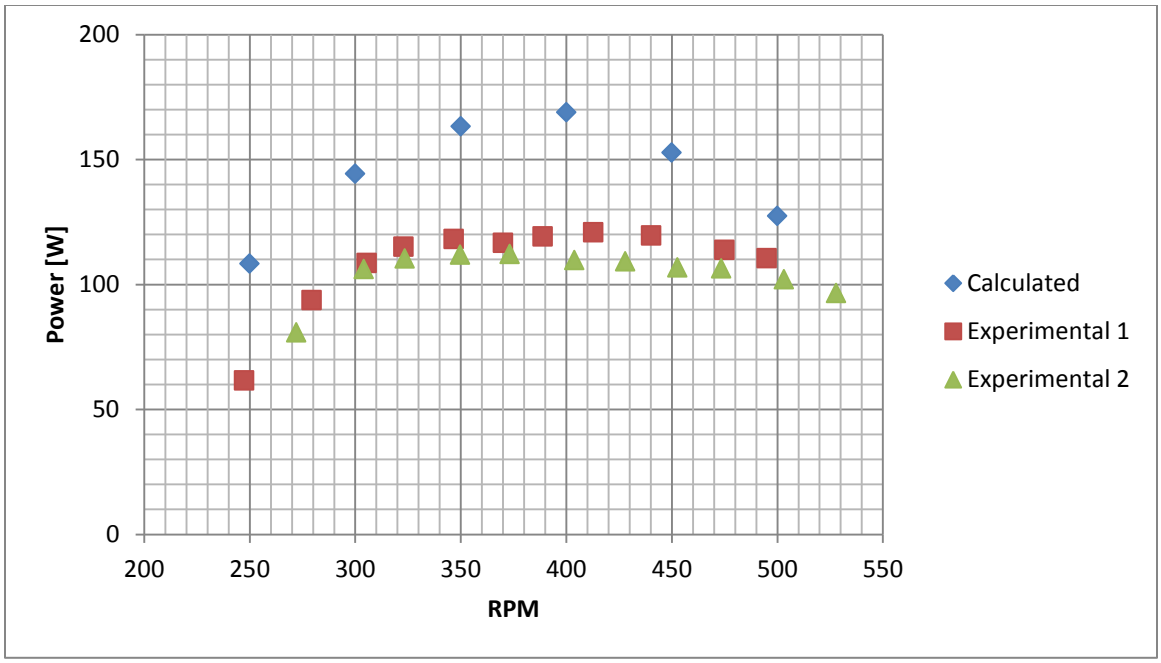

Figure 5.19 Experimental and Calculated Power at 6 m/s for Linearized Blade

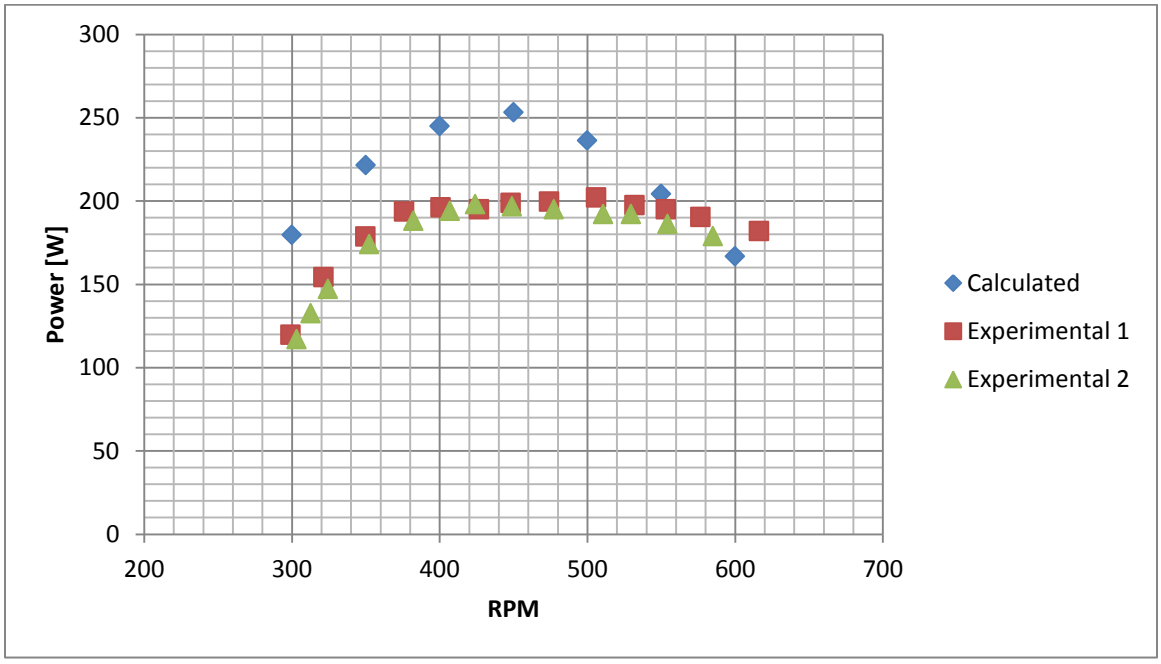

Figure 5.20 Experimental and Calculated Power at 7 m/s for Linearized Blade

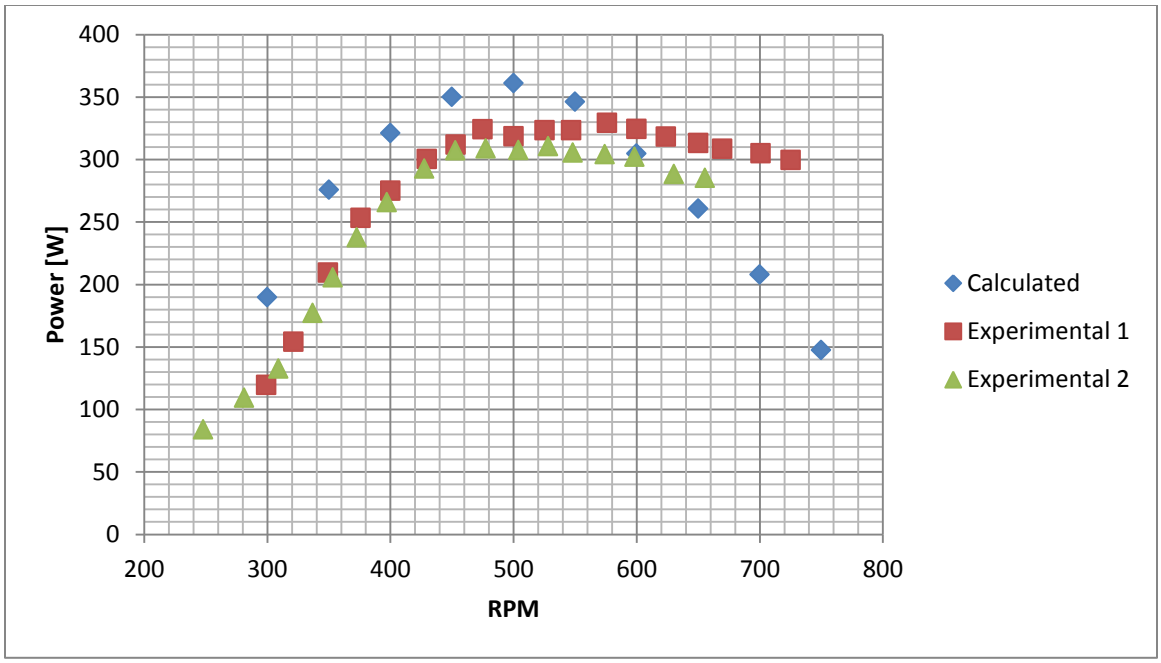

Figure 5.21 Experimental and Calculated Power at 8 m/s for Linearized Blade

The experimental results for the optimal blade geometry given in Reference 31 are shown in Figure 5.22.

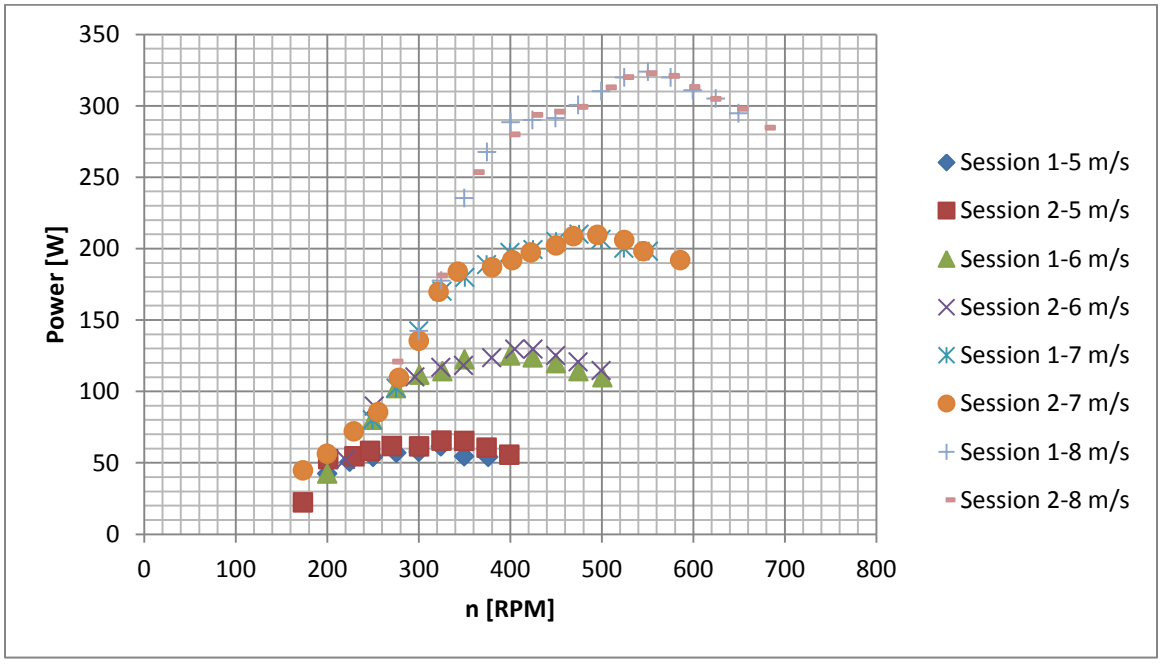

Figure 5.22 Optimal Blade Geometry Experimental Data

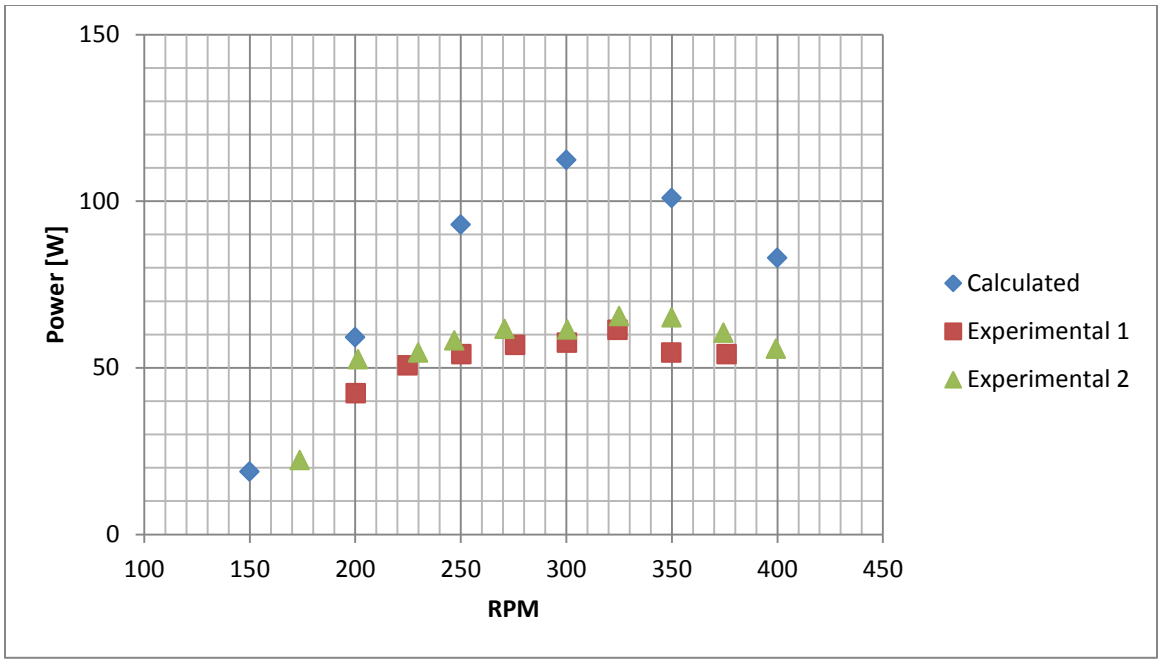

Figure 5.23 Experimental and Calculated Power at 5 m/s for Optimal Blade Geometry

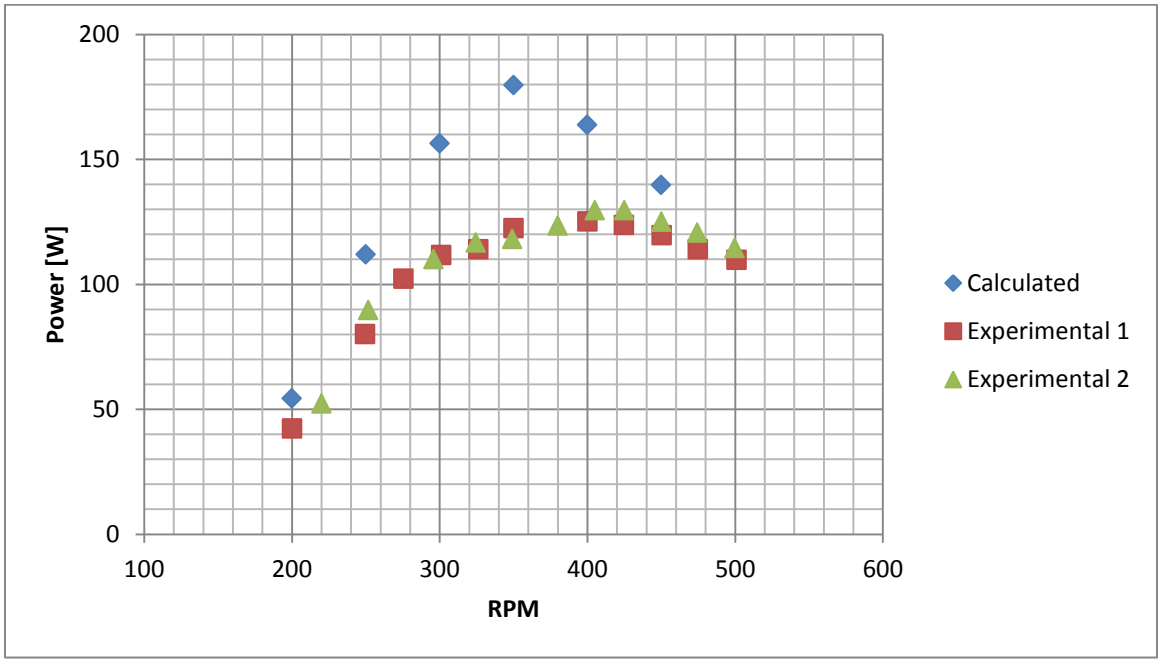

Figure 5.24 Experimental and Calculated Power at 6 m/s for Optimal Blade Geometry

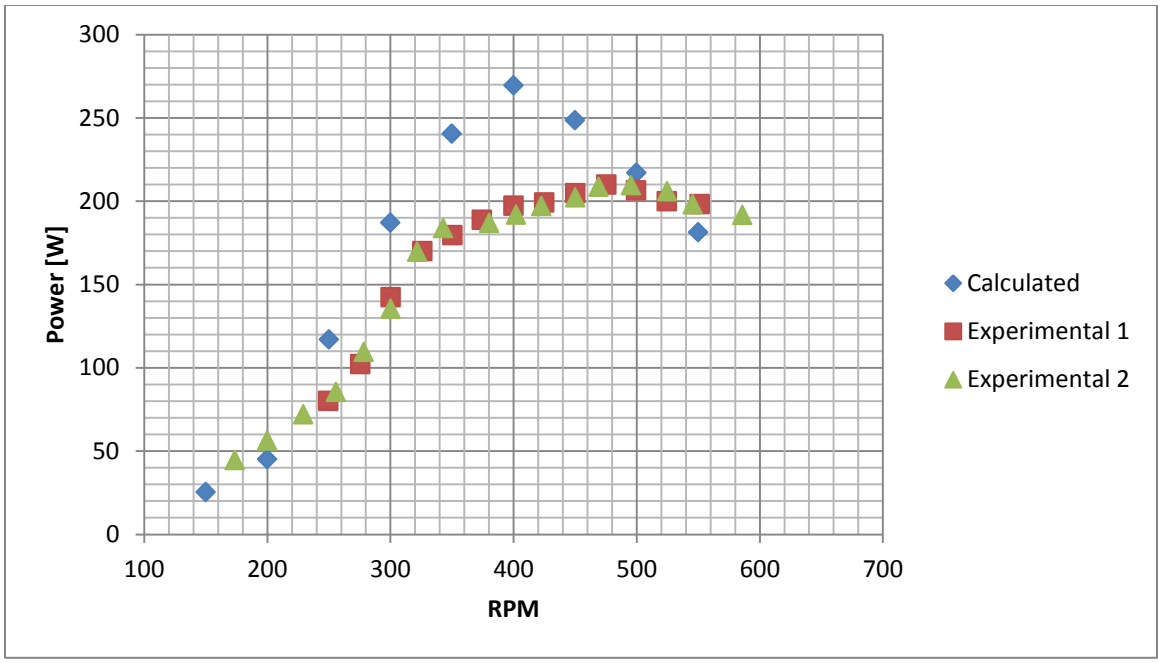

Figure 5.25 Experimental and Calculated Power at 7 m/s for Optimal Blade Geometry

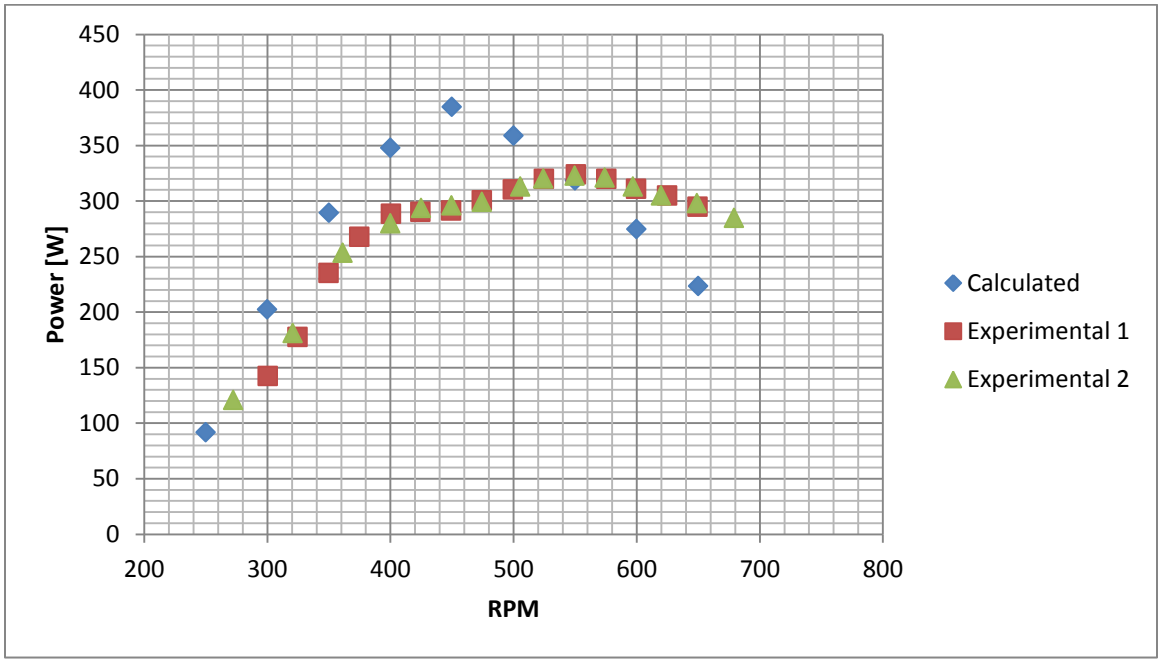

Figure 5.26 Experimental and Calculated Power at 8 m/s for Optimal Blade Geometry

For variable speed wind turbines, the RPM is varied to obtain maximum power output. Using the data in Figure 5.18 through Figure 5.26, a power curve can be constructed by taking the maximum power output at each velocity for both the linearized blade and the optimal blade. Figure 5.27 shows the comparison of the calculated power curve and the experimental power curve for the linearized blade.

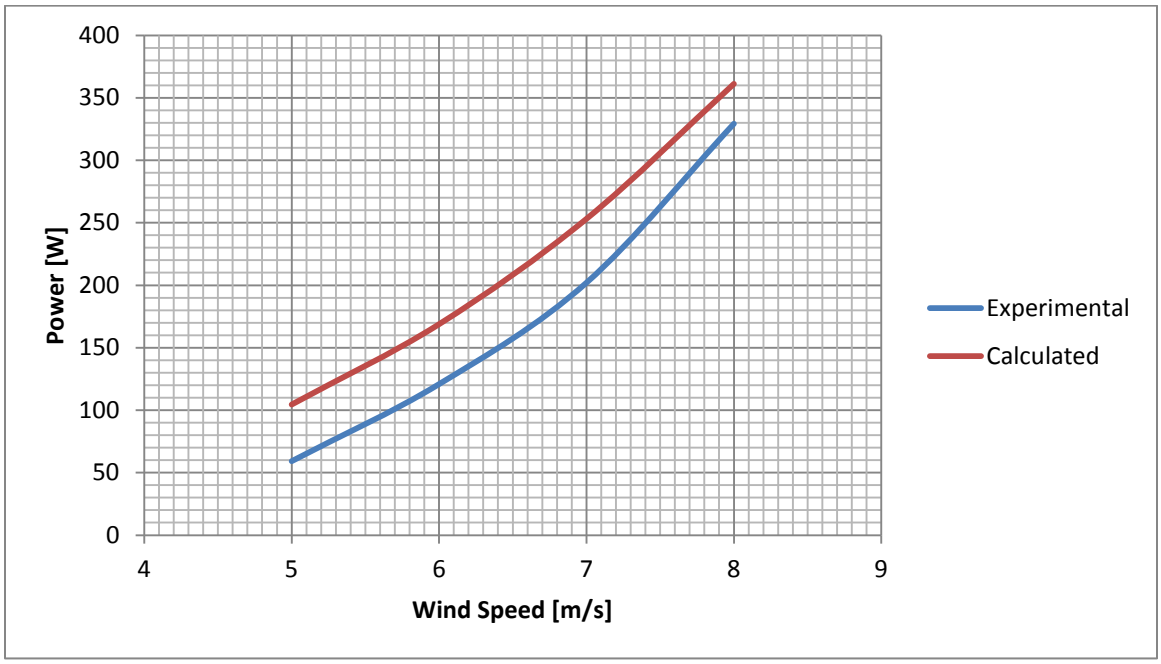

Figure 5.27 Linearized Blade Power Curve

Figure 5.28 shows the calculated and experimental power curve for the optimal blade geometry.

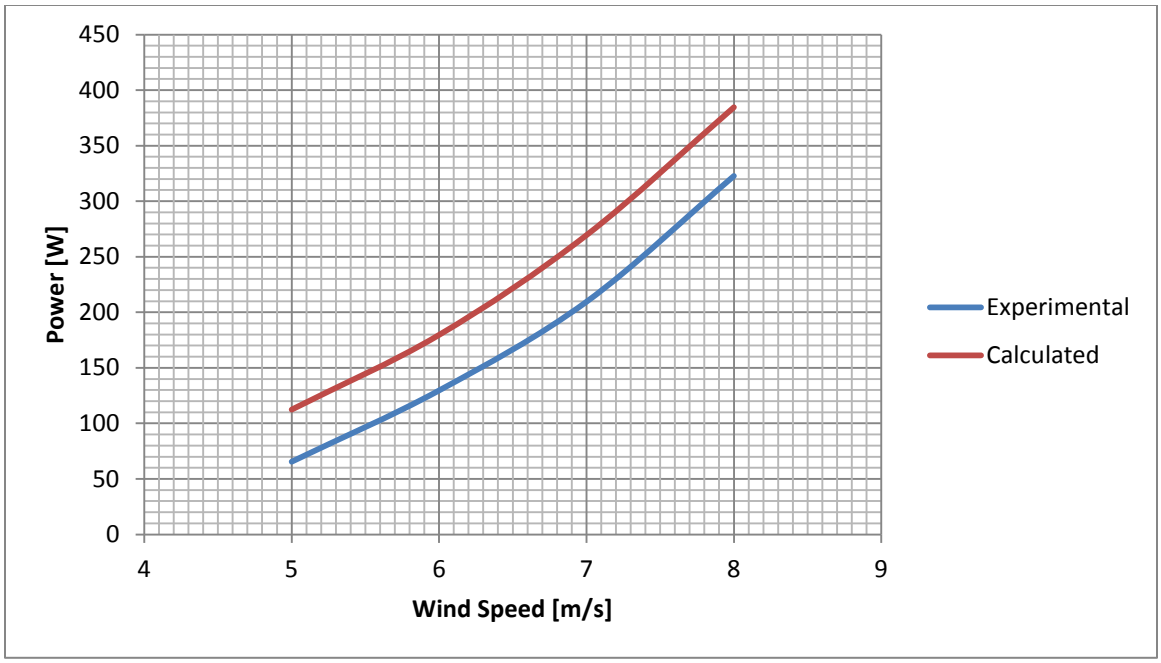

Figure 5.28 Optimal Blade Power Curve

The over prediction of the power output is expected for the calculated power curve. The calculated power shown in Figure 5.27 and Figure 5.28 is the aerodynamic power of the rotor and not the electrical power output by the generator. The electrical output of the generator includes the mechanical and electrical efficiencies of the gearbox and generator of the wind turbine, therefore it is always lower than the aerodynamic power.

Comparing the power curve of the current method and the power curve generated from the method outlined in Reference 31, the current method over predicts the performance whereas van Dorst's method underestimates the power of the turbine. Figure 5.29 shows the comparison of the van Dorst's numerical method and the experimental data for the linear blade.

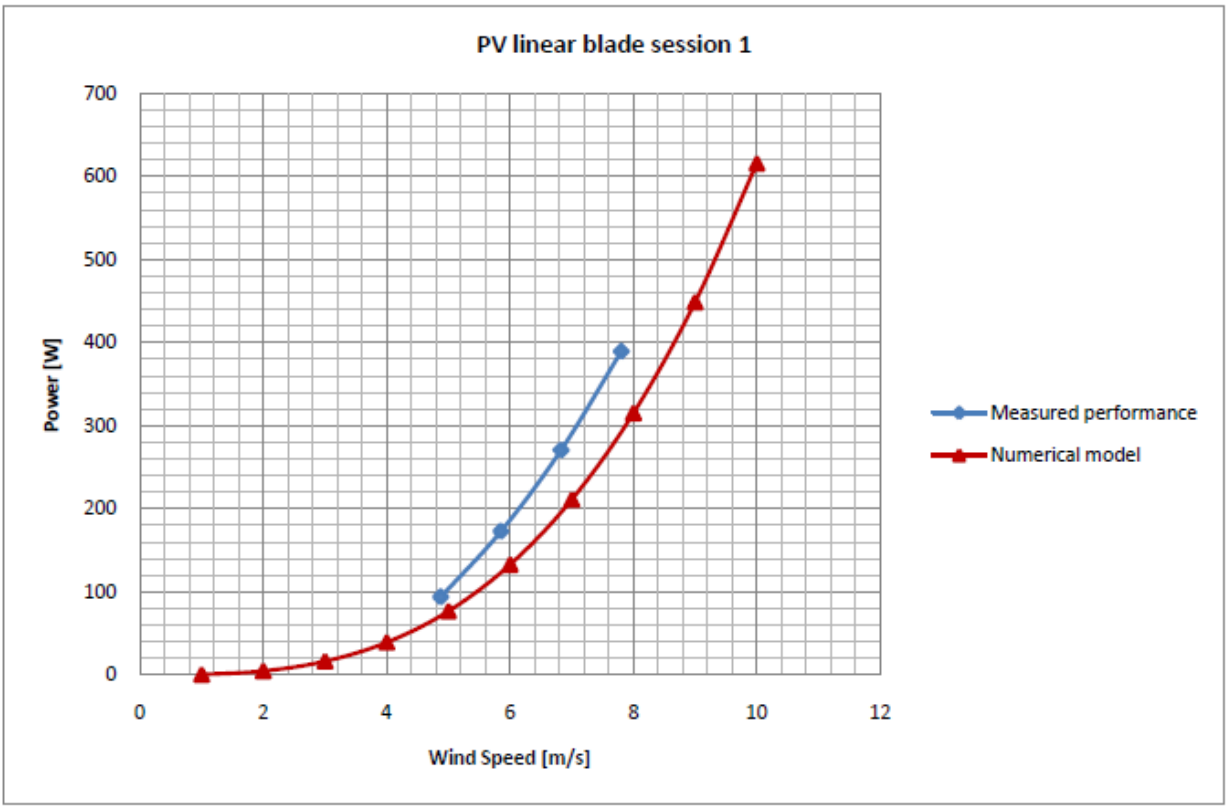

Figure 5.29 Power Curve Comparison from Experimental and Numerical Model[31]

While the prediction of the numerical model seems closer to the experimental data, it should be noted that the mechanical and electrical efficiencies were not taken into account in this model. Therefore, if the system efficiencies are taken into account, the prediction will reduce even further.

5.4 Wake Discretization Dependency Tests

From the previous analyses, it is determined that the wake discretization significantly affects the calculated power of the rotor as well as the convergence of the solution. Due to computation time, it is not recommended to discretize the wake into very small increments. This increases the computational time significantly and could defeat the purpose of a quick design tool. A simple "mesh" dependency test is performed to determine the number of wake points that will produce a relatively accurate solution. The mesh dependency is performed with the DonQi duct and hub

geometry and the linearized blade geometry at a wind speed of 5 m/s and at 300 RPM. Figure 5.30 shows the convergence as the number of wake panels is increased.

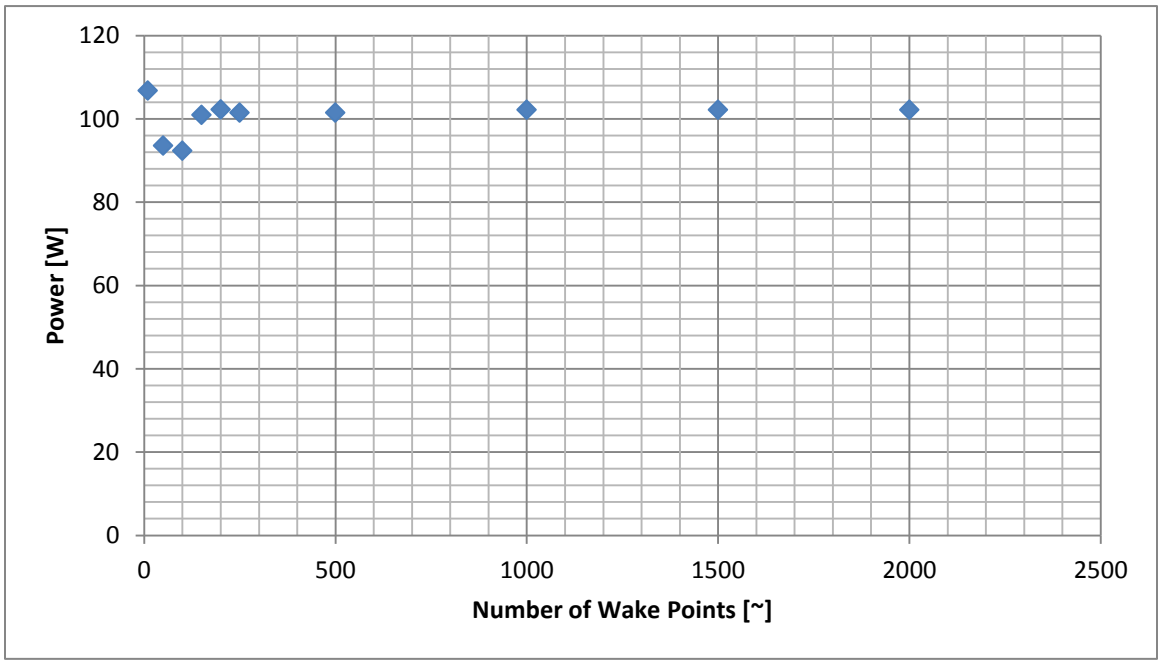

Figure 5.30 Wake Mesh Dependency Test

As Figure 5.30 shows, the solution converges when the wake panels are greater than 200. Increasing the wake panels past this point only increases the calculation time without any more gain in accuracy. Therefore, it is not recommended to exceed 200 wake panel points in the analysis.

5.5 Angular Error Analysis

Van Dorst[31] mentions the fact that blade pitch was manually set using a flat bar. This could lead to errors in the analysis as the global blade pitch used in the experiments may not be exactly 10 degrees as was stated in Reference 31. Also, van Dorst points out that the global blade pitch may have changed during the test due to the way the blade pitch was set (using a set screw). This can be seen in the repeatability issues of the experimental data. To see the effect of these sources of error the global blade pitch is varied by +/- 1 degree in the analysis to determine the effect on the power output of the wind turbine. Figure 5.31 through Figure 5.34 show the effect of global

pitch angle on the power output for the linearized blade geometry. The optimal blade geometry has a similar trend.

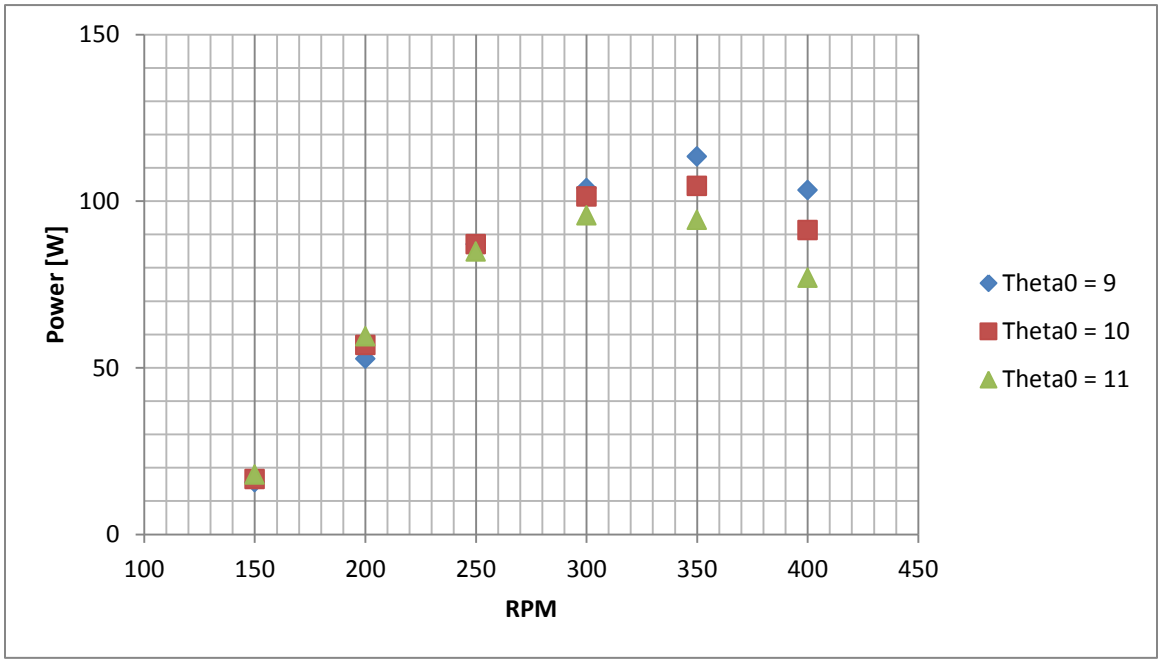

Figure 5.31 Calculated Power at 5 m/s for Linearized Blade

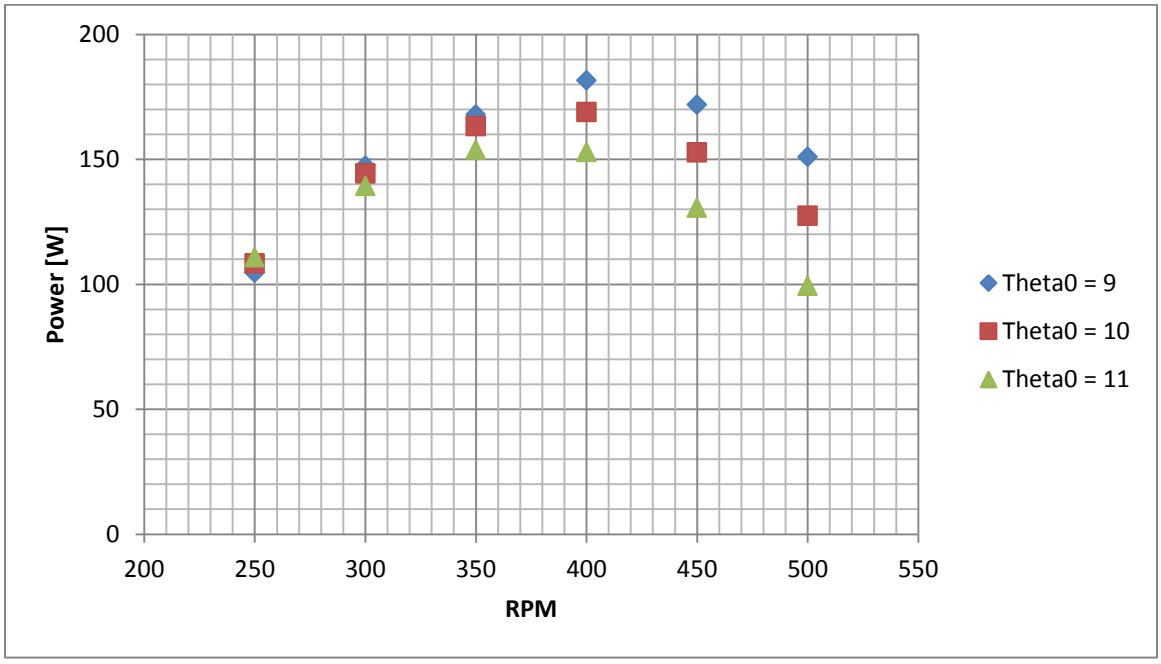

Figure 5.32 Calculated Power at 6 m/s for Linearized Blade

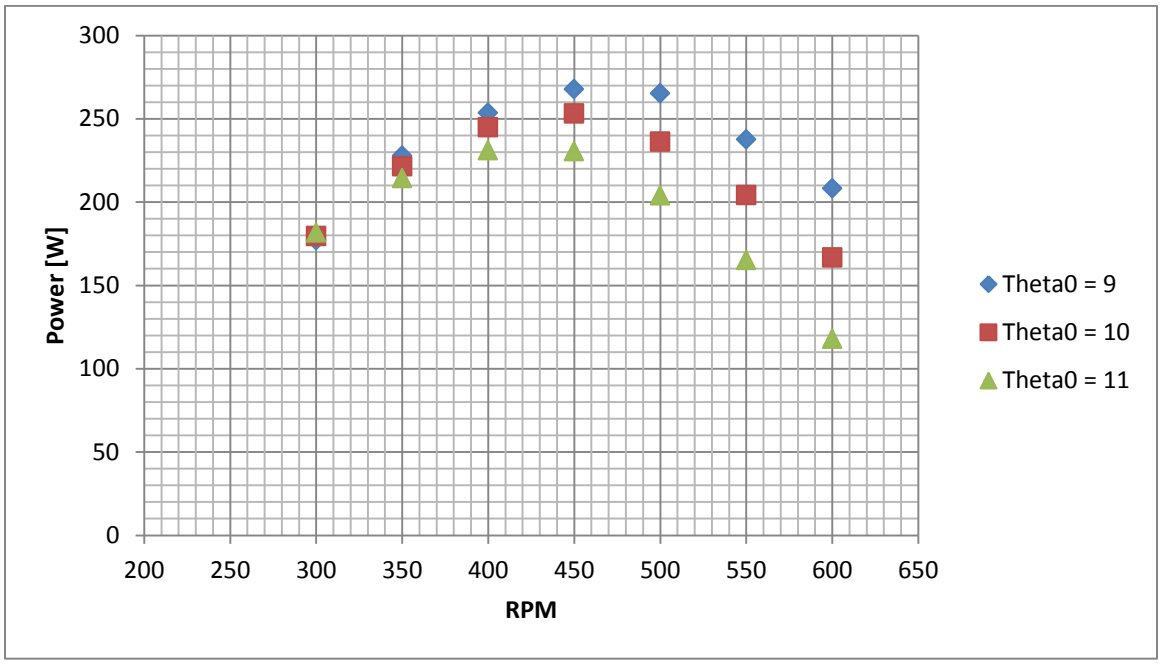

Figure 5.33 Calculated Power at 7 m/s for Linearized Blade

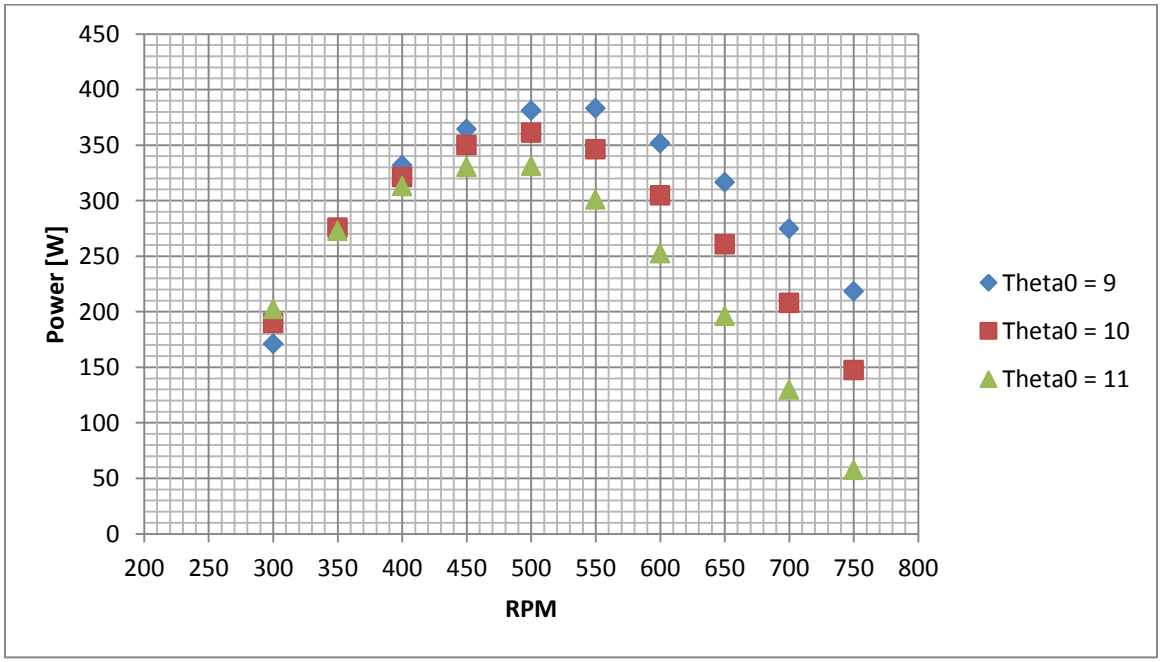

Figure 5.34 Calculated Power at 8 m/s for Linearized Blade

Figure 5.31 through Figure 5.34 show an increase in the blade angle will reduce the power output at higher RPM and a reduction in the global blade angle will increase the power output at higher RPM. It is difficult to determine if this is the exact cause of the discrepancies between the calculated and experimental parameters. More testing is to be done to determine the source of the discrepancies.

5.6 Wind Turbine Efficiency

During the translation of the aerodynamic power generated by the rotor to the electrical power output of the generator there are some other factors that come into play. The power output to the grid is then defined as:

$$P = C_P \eta \frac{1}{2} \rho \pi R_{Tip}^2 U^3 \tag{85}$$

Where η, is the efficiency overall efficiency (mechanical and electrical) of the system after the rotor. Most wind turbines must utilize a gearbox to increase the RPM of the shaft attached to the generator. There are mechanical efficiencies associated with the gearbox that depend on the gearbox design and how much friction is involved. Typical gearbox efficiencies are on the order of 95 to 98 percent[2]. These efficiencies are for large commercial wind turbine gearboxes. Smaller gearboxes have slightly lower efficiencies due to increase friction effects.

There are also efficiencies associated with the generator itself. These will reduce the power output by the generator and are typically on the order of 90 to 95 percent[2]. The mechanical and electrical efficiencies were not explicitly defined for the experimental data provided in Reference 31. Figure 5.35 shows the data presented in Figure 5.19 along with the calculated power assuming an efficiency of 0.72. A mechanical and electrical efficiency of 0.72 is a little low for large wind turbines, which are usually in the range of 0.8 to 0.9. The lower efficiency could be due to the generator not operating at its peak performance and/or a gearbox with a reduced efficiency.

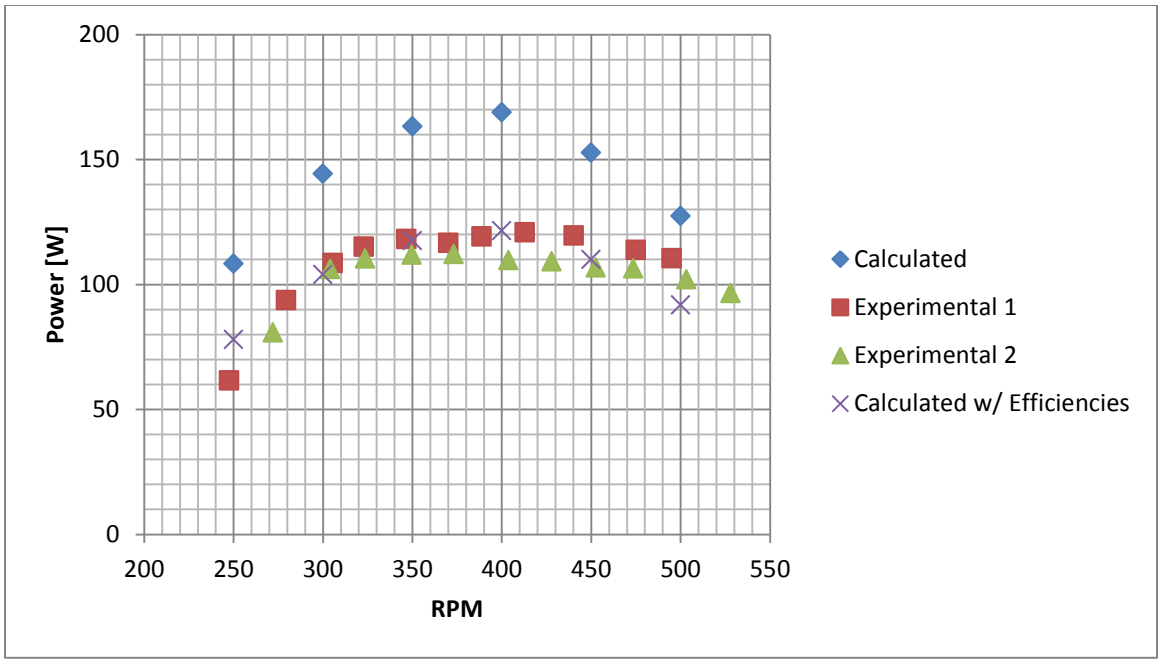

Figure 5.35 Calculated Power with Efficiencies at 6 m/s for Linearized Blade

As the figure shows, the calculated data including mechanical and electrical efficiencies agrees quite well with the experimental data. There is still a slight deviation in the power at higher RPM, but this may be caused by the efficiency changing at higher RPM due to more efficient operation of the generator or it may be due to dynamic effects. The mechanical efficiencies may change depending on the wind speed and rotor RPM, which could explain why the calculated aerodynamic power is closer to the experimental power at higher wind speeds.

5.7 Conclusions

From Figure 5.18 through Figure 5.26 it is seen that the current method over predicts the power when compared to the experimental data, but it follows a similar trend. Most of this over-prediction may be caused by the calculated power of the current method is just the aerodynamic power of the blades and the mechanical and electrical efficiencies are not included in the data shown in Figure 5.18 through Figure 5.26.

Including the mechanical and electrical efficiencies may produce a power output closer to the experimental values, but the actual mechanical and electrical efficiencies of the system must be

known to determine how close the calculated power is to the experimental. The electrical and mechanical efficiency curves for the different rotor RPM must be determined so the correct efficiency is applied to the correct conditions.

Errors in the angular orientation of the blades could be the source of the discrepancies in the power output at higher velocities, but more testing is necessary to determine the actual source of offset. Dynamic effects could also cause the deviation in the calculated power with respect to the experimental data. These are not taken into account in the current method.

6 Conclusions and Recommendations

It is concluded that the current method could be used as a design tool for the initial design of diffuser augmented wind turbines. Although, more validation is needed, the calculation speed and relative accuracy of the results could yield a valuable design tool for DAWTs. This will allow DAWT designers to analyze multiple blade and duct designs before taking the design to more detailed and time consuming calculations such as CFD. It is recommended that some wind tunnel testing be performed on fixed pitched blades as well as different duct geometries for validation purposes to assist in the determination of the accuracy of the surface vorticity method.

Viscous effects are not taken into account with the current method and are a significant factor in the performance of a DAWT. Viscous effects could cause separation from the duct effectively squeezing the flow and minimizing the low pressure region behind the rotor. This would reduce the power the rotor produces. This could be countered by adding slots that would direct outside air into the duct and re-energizing the boundary layer of the duct. Viscous effects could be taken into account using various boundary layer methods to calculate the displacement thickness and use that as the stream tube boundary in the design iteration. It is recommended that further research is done into viscous methods to be implemented with the current methodology.

Dynamic stall effects and Reynolds effects should also be taken into account in future design work using the current method of analysis. Reynolds number effects can easily be included in the model by using a larger data set for the 2D airfoil data, which would include the data for different Reynolds numbers. There are also various dynamic stall models that can be included in

the aerodynamic analysis of the blades. More research is to be done in this area to determine the best dynamic stall model to be used in the analysis.

The 2D approach to the rotor wake can be replaced with a time dependent simulation of the rotor wake using vortex filaments. This may allow for a better prediction of the wake influence on the duct, hub and rotor resulting in better power prediction. This increase in detail will come at the cost of longer computational times due to the time dependent simulation. It is unclear how much longer this would take when compared to the current method.

7 References

1. Manwell, J.F., McGowan, J.G., Rogers, A.L., Wind Energy Explaned: Theory, Design and Application, 2nd Ed., John Wiley and Sons,Chichester, West Sussex. 2009
2. Burton, Tony, Wind Energy Handbook, John Wiley & Sons, LTD, New York, NY. 2001.
3. Anon., http://landartgenerator.org/readsustainablehistory.html, Last Accessed 11/19/2013
4. Philips, D.G. Doctoral Dissertation: An Investigation on Diffuser Augmented Wind Turbine Design. Auckland, Australia: University of Auckland, 2003.
5. Ten Hoopen, P.D.C, Master of Science Thesis: An Experimental and Computational Investigation of a Diffuser Augmented Wind Turbine, Delft University of Technology, October 30, 2009.
6. Sanuki, M. Studies on Biplane Windvanes Ventilator-tubes and Cup Anemometers, Papers in Meteorology and Geophysics, Volume 1, No. 2: pg. 279-290, December 1950
7. Iwasaki, M. The Experimental and Theoretical Investigation of Windmills, Reports of Research Institute for Applied Mechanics, Volume2, No.8: pg. 181-229, December 1953
8. Lilley, G. M. and Rainbird, W.J., Report No. 102 A Preliminary Report on the Design and Performance of Ducted Windmills, The College of Aeronautics Cranfield, April, 1956
9. Kogan, A., Seginer, A., T.A.E Report 32A: Final Report on Shroud Design, 1963
10. Kogan, A., Seginer, A., T.A.E. Report 32: Shrouded Aerogenerator Design Study II, Axisymmetric Shroud Performance, 1963
11. Igra, Ozer, The Shrouded Aerogenerator, Energy, Volume 2: pp. 429-437, 1977.
12. Igra, Ozer, Research and Development for Shrouded Wind Turbines, Energy Conv. & Mgmt. Volume 21, pp. 13-48, 1981.
13. Foreman, K.M., Gilbert, B.L., Fluid Dynamics of Diffuser-Augmented Wind Turbines, Journal of Energy, Volume 2, No. 6: pg. 368-374, November-December, 1978
14. Foreman, K.M., Gilbert, B.L., Experimental Demonstration of the Diffuser-Augmented Wind Turbine Concept, Journal of Energy, Volume 3, No. 4: pg. 235-240, July-August, 1979
15. Foreman, K.M., Gilbert, B.L., Grumman Aerospace 105: Experiments with a Diffuser Augmented Wind Turbine, Journal of Energy Resources, pg. 46-54, 1983

16. Foreman, K.M., Preliminary Design and Economic Investigations of Diffuser Augmented Wind Turbines (DAWT), Research Department Grumman Aerospace. New York, 1981
17. Lewis, R.I., Williams, J.E., Abdelghaffar, M.A., A Theory and Experimental Investigation of Ducted Wind Turbines, Wind Engineering, Volume 1, No. 2: Pg. 104-125, 1977
18. Phillips, D. G., Flay, R. G. J., Nash, T.A., Aerodynamic Analysis and Monitoring of the Vortec 7 Diffuser-Augmented Wind Turbine, IPENZ Conference April 14, 1999.
19. van Bussel, Gerard, The Science of Making More Torque from Wind: Diffuser Experiments and Theory Revisited, Journal of Physics: Conference Series 75, 2007
20. Jamieson, P. Beating Betz – Energy Extraction Limits in a Uniform Flow Field. Garrad Hassan and Partners, 2008, http://www.gl-garradhassan.com/assets/downloads/Beating_Betz_-_Energy_Extraction_Limits_in_a_Uniform_Flow_Field.pdf, Last Accessed 3/14/2014
21. Werle, M.J., Pretz, W.M., Ducted Wind/Water Turbines and Propellers Revisited, Journal of Propulsion and Power, Vol. 24, No. 5, September-October 2008
22. Hansen, M.O, Sorensen, N. and Flay, R.G. Effect of Placing a Diffuser Around a Wind Turbine, Wind Energy, 2000; Volume 3: pg. 207-213.
23. Wang, Sheng-Huan and Chen, Shih-Hsiung, Blade Design for a Ducted Wind Turbine, 9th Asian International Conference on Fluid Machinery, October 16-19, 2007, Jeju, Korea
24. Hu, Ssu-Yuan and Cheng, Jung-Ho, Innovatory Designs for Ducted Wind Turbines, Renewable Energy 2007 Volume 33: pg. 1491-1498
25. Gaden, David and Bibeau, Eric, A Numerical Investigation into the Effect of Diffusers on the Performance of Hydro Kinetic Turbines using a Validated Momentum Source Turbine Model, Renewable Energy 2010, Volume 35: pg. 1152-1158.
26. Foote, Tudor, Master of Science Thesis: Numerical Modeling and Optimization of Power Generation from Shrouded Wind Turbines, Washington University St. Louis, 2011
27. Aranake, Aniket et al., Computational Analysis of Shrouded Wind Turbine Configurations, 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, 07-10 January 2013, Grapevine (Dallas/Ft. Worth Region), Texas

28. Ohya, Yuji, et al. Development of a Shrouded Wind Turbine with a Flanged Diffuser, Journal of Wind Engineering and Industrial Aerodynamics, 2008, Volume 96, pg. 524-539
29. Ohya, Yuji and Karasudani, Takashi, A Shrouded Wind Turbine Generatine High Output Power with Wind-lens Technology, Energies 2010, Volume 3: pg. 634-649.
30. FloDesign, http://flodesign.org/
31. Van Dorst, F.A., An Improved Rotor Design for a Diffuser Augmented Wind Turbine, Master's Thesis, TU Delft, January 10, 2011.
32. Widnall, S., Potential Flow Calculations of Axisymmetric Ducted Wind Turbines, Massachusetts Institute of Technology, July 2009. http://dspace.mit.edu/bitstream/handle/172.1/46707/turbine3.pdf?sequence=1, Last accessed 3/14/2014
33. Anon. http://www.windpoweringamerica.gov/wind_maps_none.asp, Last accessed 2/15/2014.
34. Mertens, Sander. Wind Energy in the Built Environment: Concentrator Effects of Buildings, Multi-Science. Essex, UK, 2006
35. Grant, Andrew, Johnstone, Cameron, Kelly, Nick, Urban Wind Energy Conversion: The Potential of Ducted Turbines, Renewable Energy 2008, Volume 33: pg. 1157-1163.
36. Glauert, H. The Elements of Aerofoil and Airscrew Theory, Second Edition, Cambridge University Press, London, UK, 1948
37. Lewis, R.I., Vortex Element Methods for Fluid Dynamic Analysis of Engineering Systems, Cambridge University Press, Cambridge United Kingdom, 1991.
38. Drela, Mark, Axisymmetric Analysis and Design of Ducted Rotors, 30 January 2005.
39. Anon. MATLAB 2012b, Mathworks 2012.
40. Platt, A. D., Buhl, M.L. WT_perf Version 3.05.00, National Renewable Energy Laboratory, 2012
41. Drela, Mark, XFOIL Version 6.97, 2008
42. Hansen, Martin O., Aerodynamics of Wind Turbines, Second Edition, Earthscan, London, UK, 2008.

Appendix A

Normalized Duct Coordinates:

X [~]	R [~]
0	0
0.006156	0.012523
0.024472	0.022191
0.054497	0.032361
0.095492	0.042616
0.146447	0.0523
0.206107	0.060706
0.273005	0.067498
0.345492	0.072284
0.421783	0.074725
0.5	0.0745
0.578217	0.07088
0.654509	0.063088
0.726995	0.050847
0.793893	0.037291
0.853553	0.024551
0.904509	0.014009
0.945503	0.006442
0.975528	0.002324
0.993844	0.000504
1	0

Appendix B

----- WT_Perf Input File ----------

WT_Perf Test04 input file. WindPACT 1.5 MW turbine (Non-dimen, Metric, Tab, PROP-PC).xxx

Compatible with WT_Perf v3.00f

----- Input Configuration ----------

False Echo: Echo input parameters to "echo.out"?

False DimenInp: Turbine parameters are dimensional?

True Metric: Turbine parameters are Metric (MKS vs FPS)?

----- Model Configuration ----------

1 NumSect: Number of circumferential sectors.

20000 MaxIter: Max number of iterations for induction factor.

1.0e-6 ATol: Error tolerance for induction iteration.

1.0e-6 SWTol: Error tolerance for skewed-wake iteration.

----- Algorithm Configuration ----------

True TipLoss: Use the Prandtl tip-loss model?

True HubLoss: Use the Prandtl hub-loss model?

True Swirl: Include Swirl effects?

True SkewWake: Apply skewed-wake correction?

True AdvBrake: Use the advanced brake-state model?

True IndProp: Use PROP-PC instead of PROPX induction algorithm?

True AIDrag: Use the drag term in the axial induction calculation.

True TIDrag: Use the drag term in the tangential induction calculation.

----- Turbine Data ----------

3 NumBlade: Number of blades.

35.0 RotorRad: Rotor radius [length].

0.05 HubRad: Hub radius [length or div by radius].

0.0 PreCone: Precone angle, positive downwind [deg].

5.0 Tilt: Shaft tilt [deg].

0.0 Yaw: Yaw error [deg].

2.437 HubHt: Hub height [length or div by radius].

19 NumSeg: Number of blade segments (entire rotor radius).

RElm	Twist	Chord	AFfile	PrntElem
0.075	11.100	0.06140	1	True
0.125	11.100	0.06826	1	True
0.175	11.014	0.07452	1	True
0.225	10.496	0.07782	1	True
0.275	9.141	0.07543	1	True
0.325	7.619	0.07188	1	True
0.375	6.098	0.06832	1	True
0.425	4.583	0.06479	1	True
0.475	3.405	0.06126	1	True
0.525	2.614	0.05771	1	True
0.575	2.140	0.05415	1	True
0.625	1.660	0.05062	1	True
0.675	1.186	0.04707	1	True
0.725	0.793	0.04360	1	True
0.775	0.511	0.04024	1	True
0.825	0.399	0.03704	1	True
0.875	0.286	0.03385	1	True
0.925	0.174	0.03066	1	True
0.975	0.061	0.02747	1	True

----- Aerodynamic Data ----------

1.225 Rho: Air density [mass/volume].

1.4639e-5 KinVisc: Kinematic air viscosity

0.0 ShearExp: Wind shear exponent (1/7 law = 0.143).

False UseCm Are Cm data included in the airfoil tables?

1 NumAF: Number of airfoil files.

"s825_2103.dat" AF_File: List of NumAF airfoil files.

----- Output Configuration ----------

True TabDel: Make output tab-delimited (fixed-width otherwise).

True KFact: Output dimensional parameters in K (e.g., kN instead on N)

True WriteBED: Write out blade element data to "<rootname>.bed"?

False InputTSR: Input speeds as TSRs?

"mps" SpdUnits: Wind-speed units (mps, fps, mph).

----- Combined-Case Analysis ----------

0 NumCases: Number of cases to run. Enter zero for parametric analysis.

WS or TSR RotSpd Pitch Remove following block of lines if NumCases is zero.

----- Parametric Analysis (Ignored if NumCases > 0) ----------

3 ParRow: Row parameter (1-rpm, 2-pitch, 3-tsr/speed).

2 ParCol: Column parameter (1-rpm, 2-pitch, 3-tsr/speed).

1 ParTab: Table parameter (1-rpm, 2-pitch, 3-tsr/speed).

True OutPwr: Request output of rotor power?

True OutCp: Request output of Cp?

True OutTrq: Request output of shaft torque?

True OutFlp: Request output of flap bending moment?

True OutThr: Request output of rotor thrust?

4, 8, 4 PitSt, PitEnd, PitDel: First, last, delta blade pitch (deg).

20, 20, 0 OmgSt, OmgEnd, OmgDel: First, last, delta rotor speed (rpm).

3, 12.5, 0.25 SpdSt, SpdEnd, SpdDel: First, last, delta speeds.

Appendix C

Blade Input:

Support Geometry

X-Location Support [%CHub]	16
Number of Supports	3
Support Drag Coefficient	0.00

Blade Geometry

Rtip [m]	0.75
Rhub [m]	0.199
Number of Blades	3
X-Location Rotor [%CHub]	14.08

r/R	Chord [m]	Pitch [deg]	Airfoil
0.199	0.1303	19.34	NACA2207
0.264	0.1303	13.86	NACA2207
0.331	0.1303	10.22	NACA2207
0.400	0.1300	7.81	NACA2207
0.466	0.1300	6.26	NACA2207
0.532	0.1243	5.11	NACA2207
0.598	0.1183	4.10	NACA2207

0.664	0.1125	3.13	NACA2207
0.733	0.1054	2.16	NACA2207
0.799	0.1054	1.28	NACA2207
0.866	0.1054	0.54	NACA2207
0.932	0.1054	0.06	NACA2207
1.000	0.1054	0.00	NACA2207

Airfoil Input:

NACA2207

Alpha	CL	CD
-180	0.0000	0.0273
-170	0.2898	0.0658
-160	0.4399	0.1765
-150	0.4780	0.3461
-140	0.4976	0.5539
-130	0.4759	0.7745
-120	0.4077	0.9811
-110	0.2974	1.1484
-100	0.1562	1.2558
-90	0.0000	1.2900
-80	-0.1562	1.2558

-70	-0.2974	1.1484
-60	-0.4077	0.9811
-50	-0.4759	0.7745
-40	-0.4976	0.5539
-30	-0.4780	0.3461
-20	-0.4399	0.1765
-10	-0.3364	0.0605
-5	-0.3400	0.0082
-4	-0.2380	0.0080
-3	-0.1280	0.0079
-2	-0.0150	0.0079
-1	0.1000	0.0080
0	0.2160	0.0072
1	0.3310	0.0071
2	0.4470	0.0071
3	0.5610	0.0071
4	0.6740	0.0073
5	0.7800	0.0078
6	0.8790	0.0100
7	0.9650	0.0107
8	1.0370	0.0117

9	1.0900	0.0129
10	1.1140	0.0143
11	1.0790	0.0161
12	0.9580	0.0270
13	0.7680	0.0900
14	0.6950	0.1028
15	0.6210	0.1128
20	0.6284	0.1765
30	0.6828	0.3461
40	0.7108	0.5539
50	0.6799	0.7745
60	0.5825	0.9811
70	0.4249	1.1484
80	0.2231	1.2558
90	0.0000	1.2900
100	-0.1562	1.2558
110	-0.2974	1.1484
120	-0.4077	0.9811
130	-0.4759	0.7745
140	-0.4976	0.5539
150	-0.4780	0.3461

160	-0.4399	0.1765
170	-0.2898	0.0658
180	0.0000	0.0273

Duct Input

Duct Geometry

Coordinates	Position
RDuct	0.864
XDuct	0.10899045
ChordDuct	1.0

X	R
0.10899045	0.863208132
0.110652231	0.872627122
0.112313766	0.876573465
0.115637329	0.881348983
0.118960645	0.8845393
0.122283962	0.887129113
0.125607525	0.889449172
0.142224354	0.899388179
0.158841182	0.906082163
0.175458258	0.910346049
0.192075086	0.913305596

0.208692162 0.915732551

0.225308991 0.917658772

0.241926066 0.919978621

0.258542895 0.922524628

0.27515997 0.924990778

0.291776799 0.927368267

0.308393628 0.9296965

0.325010703 0.93203689

0.341627532 0.934256131

0.358244607 0.936039195

0.374861436 0.93737707

0.391478511 0.938538252

0.40809534 0.939658772

0.424712416 0.940745965

0.441329244 0.941774471

0.45794632 0.9426965

0.474563148 0.943504297

0.491179977 0.944224481

0.507797053 0.944884091

0.524413881 0.945509956

0.541030957 0.946113603

0.557647785 0.946695242

0.574264861 0.947254244

0.590881689 0.947791239

0.607498765 0.948318172

0.624115594 0.948856005

0.640732669 0.949426535

0.657349498 0.950049885

0.673966326 0.950732341

0.690583402 0.951469922

0.707200231 0.952258227

0.723817306 0.95309411

0.740434135 0.953992245

0.75705121 0.954979878

0.773668039 0.956085097

0.790285114 0.957330119

0.806901943 0.958670719

0.823519019 0.960016558

0.840135847 0.96127772

0.856752676 0.962385454

0.873369751 0.963508279

0.88998658 0.964942989

0.906603656 0.966798365

0.923220484 0.969046322

0.93983756 0.97144498

0.956454388 0.973677845

0.973071464 0.975833159

0.989688292 0.978166632

1.006305368 0.980613289

1.022922197 0.982987843

1.039539025 0.985334731

1.056156101 0.987851394

1.072772929 0.990859568

1.089390005 0.994119053

1.106006834 0.999671348

1.089390005 0.991536366

1.072772929 0.984508279

1.056156101 0.979037728

1.039539025 0.973314399

1.022922197 0.967286313

1.006305368 0.960791658

0.989688292 0.954054496

0.973071464 0.948503249

0.956454388 0.942611821

0.93983756 0.936876546

0.923220484 0.929867743

0.906603656 0.923650178

0.88998658 0.918153637

0.873369751 0.911606372

0.856752676 0.906696919

0.840135847 0.900898344

0.823519019 0.896082582

0.806901943 0.890702788

0.790285114 0.884057011

0.773668039 0.879246699

0.75705121 0.873578705

0.740434135 0.868888493

0.723817306 0.864121777

0.707200231 0.859573884

0.690583402 0.854765458

0.673966326 0.850038147

0.657349498 0.845436177

0.640732669 0.839891008

0.624115594 0.835532174

0.607498765 0.832258856

0.590881689 0.828366799

0.574264861 0.823835464

0.557647785 0.81916244

0.541030957 0.81443471

0.524413881 0.809622092

0.507797053 0.804885349

0.491179977 0.800716412

0.474563148 0.797500314

0.45794632 0.793710333

0.441329244 0.790247118

0.424712416 0.786656885

0.40809534 0.783330958

0.391478511 0.779835464

0.374861436 0.776094949

0.358244607 0.772584993

0.341627532 0.769485223

0.325010703 0.766791658

0.308393628 0.7643559

0.291776799 0.76317271

0.27515997 0.764174806

0.258542895 0.766475582

0.241926066 0.768712848

0.225308991 0.770753301

0.208692162 0.774752044

0.192075086 0.780929994

0.175458258 0.787432614

0.158841182 0.797245232

0.142224354 0.807509746

0.125607525 0.825162859

0.122283962 0.828715364

0.118960645 0.832627541

0.115637329 0.837278348

0.112313766 0.843740935

0.110652231 0.848655628

0.10899045 0.863208132

Hub Input:

Hub Geometry

Coordinates	Position
X_nose Hub	0.183396921
Hub Chord	1.25
X	R

0.183396921 0.00000000

0.185472666 0.011385663

0.187548411 0.017968979

0.191699901 0.02543974

0.195851638 0.03035611

0.200003129 0.03429323

0.204154866 0.037673234

0.224912564 0.051598826

0.245670509 0.062256969

0.266428454 0.067166632

0.287186399 0.071566757

0.307944097 0.075170614

0.328702042 0.077469294

0.349459987 0.078737581

0.370217932 0.079230979

0.39097563 0.079189478

0.411733575 0.078842171

0.43249152 0.078412911

0.453249465 0.078039824

0.474007163 0.077721023

0.494765108 0.077441836

0.515523053 0.077187173

0.536280998 0.07694215

0.557038943 0.07669126

0.577796641 0.076419199

0.598554586 0.076110669

0.619312531 0.075750367

0.640070476 0.075323412

0.660828174 0.074820583

0.681586119 0.074248795

0.702344064 0.073616433

0.723102009 0.07293209

0.743859707 0.07220394

0.764617652 0.071417103

0.785375597 0.070519388

0.806133542 0.069455879

0.826891487 0.068218612

0.847649185 0.066906728

0.86840713 0.065622092

0.889165075 0.064357158

0.90992302 0.06304171

0.930680718 0.061581639

0.951438663 0.059861035

0.972196608 0.058124293

0.992954553 0.056848669

1.013712251 0.055548942

1.034470196 0.053400964

1.055228141 0.050752253

1.075986086 0.048258646

1.096743784 0.045992664

1.117501729 0.04363865

1.138259674 0.041041082

1.159017619 0.038731712

1.179775564 0.03640327

1.200533262 0.033820583

1.221291207 0.031426326

1.242049152 0.029105219

1.262807097 0.026694823

1.283564795 0.024014462

1.30432274 0.020550409

1.325080685 0.016857682

1.34583863 0.014154894

1.366596328 0.011454831

1.387354273 0.007437225

1.408112218 0.004029973

1.428870163 0.000000000

Operating Condition Input:

Operating Condition			
Altitude[m]	1		
Rho [kg/m3]	1.225		
Viscosity	0.00001789		
Eta_mech	1.0		
	Begin	End	Step
Velocity[m/s]	5	5	1
RotorRPM[RPM]	300	300	1
Theta0[deg]	10	10	1

Matlab Code:

```
%% Introduction
%D.A.W.T.A.C.
%Created by: Jonathan Carroll
%
%Submitted to the Department of Aerospace Engineering and the Faculty of
%the Graduate School of Engineering in Partial Fulfillment of the
%Requirements for the Degree of Master of Science in Aerospace Engineering.
%March 7, 2014
%
%Inputs: All inputs are made through corresponding text files.

clc
clear all

%% Import geometry data

Input.DuctGeoFile  = 'DuctGeoDonQi.txt';
```

```
Input.HubGeoFile   = 'HubGeoDonQi.txt';
Input.BladeGeoFile = 'BladeGeoDonQi_Linearized.txt';
Input.OpCondFile   = 'OperatingCondition.txt';

disp 'Importing Geometry'

[ DuctGeo, CBGeo, OpCond, BladeGeo ] = FunImport( Input );

disp 'Import Complete'

%% Hub, Duct and Blade Geometry Data

disp 'Generating Geometry'

[ GeoData ] = FunGeoPrep( CBGeo, DuctGeo, BladeGeo );

%% Hub and Duct Coupling Coefficients

disp 'Calculating Duct and Center Body Influence Coefficients'

[ CoupData ] = FunCouple( GeoData );

%% Calculate the influence coefficients of the wake

disp 'Calculating Wake Influence Coefficients'

[ GeoData, CoupData ] = FunWake( GeoData,CoupData );

disp 'Calculating Rotor Influence Coefficients'

[ GeoData, CoupData ] = FunCoupSource( GeoData,CoupData );

%disp 'Calculating Support Structure Influence Coefficients'

disp 'Influence Coefficients - Complete'

%% Solve for Initial Duct and Hub Vorticities

Output.WakeGammas = zeros(numel(GeoData.RBldSec),1);
Output.BldSource  = zeros(numel(GeoData.RBldSec)-1,1);
Output.SupSource  = zeros(numel(GeoData.RBldSec)-1,1);

[ Output ] = FunSolve( CoupData,Output,GeoData, 1 );

Output.NoRotorGamma = Output.Gamma;
Output.NoRotorCp    = Output.Cp;

%% Iteration Step

[ Power, GeoData, CoupData, Output ] = FunIterate2( GeoData, CoupData,
OpCond, Output );
```

```
disp 'Analysis Complete'

disp 'Done'


function [ DuctGeo HubGeo OpCond BladeGeo ] = FunImport( InputFile )
%Imports the input data from the input files

DuctGeoFile  = InputFile.DuctGeoFile;
HubGeoFile   = InputFile.HubGeoFile;
BladeGeoFile = InputFile.BladeGeoFile;
OpCondFile   = InputFile.OpCondFile;

%Import Duct Geometry

FDuctGeo = fopen(DuctGeoFile);

Duct0 = textscan(FDuctGeo, '%s %s',1,'headerlines',1,'delimiter','\t',
'MultipleDelimsAsOne',1);
Duct1 = textscan(FDuctGeo, '%s %f',3,'delimiter','\t',
'MultipleDelimsAsOne',1);
Duct2 = textscan(FDuctGeo, '%f %f','headerlines',2,'delimiter','\t',
'MultipleDelimsAsOne',1);

fclose(FDuctGeo);

DuctGeo.Coords = Duct0{1,2}(1,1);
DuctGeo.Radius = Duct1{1,2}(1,1);
DuctGeo.XNose  = Duct1{1,2}(2,1);
DuctGeo.Chord  = Duct1{1,2}(3,1);

if strcmp(DuctGeo.Coords,'Position')

  disp 'Duct coordinates are POSITION'

  DuctGeo.XCoords = Duct2{1,1}(1:end);
  DuctGeo.YCoords = Duct2{1,2}(1:end);

elseif strcmp(DuctGeo.Coords,'Normalized')

  disp 'Duct coordinates are NORMALIZED'

  DuctGeo.XCoords = Duct2{1,1}(1:end).*DuctGeo.Chord+DuctGeo.XNose;
  DuctGeo.YCoords = Duct2{1,2}(1:end).*DuctGeo.Chord+DuctGeo.Radius;

end

%Import CenterBody Geometry

FCBGeo = fopen(HubGeoFile);
```

```
CB0 = textscan(FCBGeo, '%s
%s',1,'headerlines',1,'delimiter','\t','MultipleDelimsAsOne',1);
CB1 = textscan(FCBGeo, '%s %f',2,'delimiter','\t','MultipleDelimsAsOne',1);
CB2 = textscan(FCBGeo, '%n
%n','headerlines',2,'delimiter','\t','MultipleDelimsAsOne',1);

HubGeo.Coords= CB0{1,2}(1,1);
HubGeo.XNose = CB1{1,2}(1,1);
HubGeo.Chord = CB1{1,2}(2,1);

if strcmp(DuctGeo.Coords,'Position')

  HubGeo.XCoords = CB2{1,1}(1:end);
  HubGeo.YCoords = CB2{1,2}(1:end);

elseif strcmp(DuctGeo.Coords,'Normalized')

  HubGeo.XCoords = CB2{1,1}(1:end).*HubGeo.Chord+HubGeo.XNose;
  HubGeo.YCoords = CB2{1,2}(1:end).*HubGeo.Chord;

end

fclose(FCBGeo);

%Import Blade Geometry

FBladeGeo = fopen(BladeGeoFile);

Support = textscan(FBladeGeo, '%s
%f',3,'headerlines',1,'delimiter','\t','MultipleDelimsAsOne',1);
Blade1 = textscan(FBladeGeo, '%s
%f',4,'headerlines',3,'delimiter','\t','MultipleDelimsAsOne',1);
Blade2 = textscan(FBladeGeo, '%f %f %f
%s','headerlines',3,'delimiter','\t','MultipleDelimsAsOne',1);

BladeGeo.SupportX   = Support{1,2}(1,1)/100;
BladeGeo.NumSupport = Support{1,2}(2,1);
BladeGeo.SupportCD  = Support{1,2}(3,1);

BladeGeo.RTip      = Blade1{1,2}(1,1);
BladeGeo.RHub      = Blade1{1,2}(2,1);
BladeGeo.NumBlades = Blade1{1,2}(3,1);
BladeGeo.RotorXLoc = Blade1{1,2}(4,1)/100;

BladeGeo.r       = Blade2{1,1}(1:end);
BladeGeo.Chord   = Blade2{1,2}(1:end);
BladeGeo.Pitch   = Blade2{1,3}(1:end);
BladeGeo.Airfoil = Blade2{1,4}(1:end);

% Add a section of code resetting the r such that all points r are greater
% than RHub.
```

```
fclose(FBladeGeo);

%Import Airfoil Data

SectionalAirfoil=BladeGeo.Airfoil;

k=1;

for i=1:numel(SectionalAirfoil)
  Compare=strcmp(SectionalAirfoil(i),'--');

  if Compare==0
    AirfoilFile(i)=strcat(SectionalAirfoil(i),'.txt');
    FAirfoil = fopen(char(AirfoilFile(i)));

    AeroData = textscan(FAirfoil, '%f %f
%f','headerlines',2,'delimiter','\t','MultipleDelimsAsOne',1);

    BladeGeo.AeroData{k,1}=[AeroData{1,1}(1:end) AeroData{1,2}(1:end)
AeroData{1,3}(1:end)];

    fclose(FAirfoil);

    k=k+1;
  end

end


%Import Operating Condition

FOpCond = fopen(OpCondFile);

OpC = textscan(FOpCond, '%s %f',
4,'headerlines',1,'delimiter','\t','MultipleDelimsAsOne',1);

OpCond.Altitude = OpC{1,2}(1);
OpCond.Rho      = OpC{1,2}(2);
OpCond.Visc     = OpC{1,2}(3);
OpCond.EtaMech  = OpC{1,2}(4);

OpCEnv = textscan(FOpCond, '%s %f %f %f',
3,'headerlines',2,'delimiter','\t','MultipleDelimsAsOne',1);

OpCond.UInfBegin   = OpCEnv{1,2}(1);
OpCond.UInfEnd     = OpCEnv{1,3}(1);
OpCond.UInfStep    = OpCEnv{1,4}(1);
OpCond.RPMBegin    = OpCEnv{1,2}(2);
OpCond.RPMEnd      = OpCEnv{1,3}(2);
OpCond.RPMStep     = OpCEnv{1,4}(2);
OpCond.Theta0Begin = OpCEnv{1,2}(3);
```

```
OpCond.Theta0End   = OpCEnv{1,3}(3);
OpCond.Theta0Step  = OpCEnv{1,4}(3);

fclose(FOpCond);

end


function [ GeoData ] = FunGeoPrep( CBGeo, DuctGeo, BladeGeo )
%Prepares geometry data for use in analysis

XCB       = CBGeo.XCoords;
YCB       = CBGeo.YCoords;
ChordCB   = CBGeo.Chord;
XNoseCB   = CBGeo.XNose;
XDuct     = DuctGeo.XCoords;
YDuct     = DuctGeo.YCoords;
XNoseDuct = DuctGeo.XNose;
DuctChord = DuctGeo.Chord;
RTip      = BladeGeo.RTip;
RHub      = BladeGeo.RHub;
NumBlade  = BladeGeo.NumBlades;
RotorLoc  = BladeGeo.RotorXLoc;
r         = BladeGeo.r;
BldChord  = BladeGeo.Chord;
BldPitch  = BladeGeo.Pitch;
SupXLoc   = BladeGeo.SupportX;
NumSup    = BladeGeo.NumSupport;
SupCD     = BladeGeo.SupportCD;

%% Hub and Duct Geometry Parameters

mCB   = numel(XCB);
mDuct = numel(XDuct);

m = mCB+mDuct;

XData = [XCB
         XDuct];
RData = [YCB
         YDuct];

TE = ((mDuct-1)/2)+mCB;

ex = 0.00001;

for imod = 1:m-1
  if imod~=mCB
    if imod<mCB
      i = imod;
    else
      i = imod-1;
    end
```

```
    dels      = sqrt((XData(imod+1)-XData(imod))^2+((RData(imod+1)-
RData(imod))^2));
    ds(i)     = dels;
    Sine(i)   = (RData(imod+1)-RData(imod))/dels;
    Cosine(i) = (XData(imod+1)-XData(imod))/dels;
    XPoint(i) = (XData(imod+1)+XData(imod))/2;
    RPoint(i) = (RData(imod+1)+RData(imod))/2;
    abscos    = abs(Cosine(i));

    if abscos>ex
      t = atan(Sine(i)/Cosine(i));
    end
    if abscos<ex
      Slope(i) = sign(Sine(i))*pi/2;
    end
    if Cosine(i)>ex
      Slope(i) = t;
    end
    if Cosine(i)<-ex && i>TE
      Slope(i) = t-pi;
    end
    if Cosine(i)<-ex && i<TE
      Slope(i) = t+pi;
    end

  end
end

m = m-2;

mCB   = mCB-1;
mDuct = mDuct-1;
TE    = TE-1;

for i = 2:mCB-1
  Curve(i) = (Slope(i+1)-Slope(i-1))/(8*pi);
end

Curve(1)   = Curve(2);
Curve(mCB) = Curve(mCB-1);

for i = mCB+2:m-1
  Curve(i) = (Slope(i+1)-Slope(i-1))/(8*pi);
end

Curve(mCB+1)  = (Slope(mCB+2)-Slope(m)-2*pi)/(8*pi);
Curve(m)      = (Slope(mCB+1)-Slope(m-1)-2*pi)/(8*pi);
Curve(TE)     = 0;
Curve(TE+1)   = 0;

%% Blade Geometry Parameters
```

```
%Rotor plane points for inflow velocity calculations
RotorXLocation = RotorLoc*ChordCB+XNoseCB;

for i=1:numel(r)-1
    rVelRotPln(i,1) = ((r(i)+r(i+1))/2)*RTip;
    drRotPln(i,1)   = (r(i+1)-r(i))*RTip;
    xVelRotPln(i,1) = RotorXLocation;
    Pitch(i,1)      = (BldPitch(i)+BldPitch(i+1))/2;
    Chord(i,1)      = (BldChord(i)+BldChord(i+1))/2;
end

RotorPln = [ xVelRotPln rVelRotPln];

%% Support Geometry Parameters

XLoc      = SupXLoc*ChordCB+XNoseCB;
XTED      = XData(TE+2,1);
XTECB     = XData(mCB+1,1);
BladeDist = RTip-RHub;


for j = 1:mCB+1
  for i = TE+2:numel(XData)
    if XLoc > XTED && XLoc > XTECB
      disp 'Support Structure is outside of duct. Check Inputs.'
      break
    elseif XData(j)>=XLoc && XData(j-1)<XLoc && XData(i)<=XLoc && XData(i-
1)>XLoc && XLoc<=XTED && XLoc<=XTECB
      %Interpolate between the defined geometry to get the
      %distance between the duct and the center body at the rotor
      %location
      RCBody    = RData(j-1)+(RData(j)-RData(j-1))*((XLoc-XData(j-
1))/(XData(j)-XData(j-1)));
      RDuctBody = RData(i)+(RData(i-1)-RData(i))*((XLoc-XData(i))/(XData(i-
1)-XData(i)));
      Dist      = RDuctBody-RCBody;
    elseif XLoc>XTED && XData(j)>=XLoc && XData(j-1)<XLoc
      disp 'Support structure is not attached to duct. Check inputs.'
      break
    elseif XLoc>XTECB && XData(i)<=XLoc && XData(i-1)>XLoc
      disp 'Support structure is not attached to hub. Check inputs.'
      break
    end
  end
end

PercentSupSection=r.*RTip./BladeDist;
PercentRBld=rVelRotPln./BladeDist;
rSupport=Dist.*PercentRBld;

for i=1:numel(r)-1
    drSup(i,1)=(PercentSupSection(i+1)-PercentSupSection(i))*Dist;
end
```

```
%% Outputs

GeoData.XData     = XData;
GeoData.RData     = RData;
GeoData.ds        = ds';
GeoData.Sine      = Sine';
GeoData.Cosine    = Cosine';
GeoData.XPoint    = XPoint';
GeoData.RPoint    = RPoint';
GeoData.Slope     = Slope';
GeoData.Curve     = Curve';
GeoData.TE        = TE;
GeoData.m         = m;
GeoData.mCB       = mCB;
GeoData.mDuct     = mDuct;
GeoData.CBChord   = ChordCB;
GeoData.CBNose    = XNoseCB;
GeoData.DuctNose  = XNoseDuct;
GeoData.DuctChord = DuctChord;
GeoData.RTip      = RTip;
GeoData.RHub      = RHub;
GeoData.NumBld    = NumBlade;
GeoData.RBldSec   = r;
GeoData.RotorPln  = RotorPln;
GeoData.drRotPln  = drRotPln;
GeoData.BldPitch  = Pitch;
GeoData.BldChord  = Chord;
GeoData.AeroData  = BladeGeo.AeroData;
GeoData.SupportX  = XLoc;
GeoData.RSupPln   = rSupport;
GeoData.drSupPln  = drSup;
GeoData.NumSup    = NumSup;
GeoData.SupCD     = SupCD;


end

function [ CoupData ] = FunCouple( GeoData )
%Calculates the influence coefficients

m      = GeoData.m;
mCB    = GeoData.mCB;
mDuct  = GeoData.mDuct;
XPoint = GeoData.XPoint;
RPoint = GeoData.RPoint;
ds     = GeoData.ds;
Cosine = GeoData.Cosine;
Sine   = GeoData.Sine;
Slope  = GeoData.Slope;
Curve  = GeoData.Curve;
TE     = GeoData.TE;

%Coupling Coefficient Matrix Formulation
```

```
for i = 1:m
  for j = 1:m
    if j ~= i
      [u v]        = FunVelocity(XPoint(j),RPoint(j),XPoint(i), RPoint(i));
      CoupIni(j,i) = (u*Cosine(j)+v*Sine(j))*ds(i);
    end
  end

  cons         = 4*pi*RPoint(i)/ds(i);
  CoupIni(i,i) = -0.5-(log(2*cons)-0.25)/cons*Cosine(i)-Curve(i);

  % Right Hand Side Values

  RHS(i) = -Cosine(i);

end

%Back Diagonal Correction

for i = mCB+1:m
  sum = 0;
  for j = mCB+1:m
    if j ~= (m-i+mCB+1)
      sum = sum-CoupIni(j,i)*ds(j);
    end
    CoupIni((m-i+mCB+1),i) = sum/ds(m-i+mCB+1);
    %CoupIni((m-i+mHub+1),i)=1;
  end
end

CoupCheck=CoupIni;

% Kutta Condition

for j = TE:m-1
  for i = 1:m
    if j > TE
      CoupIni(i,j) = CoupIni(i,j+1);
    else
      CoupIni(i,j) = CoupIni(i,j)-CoupIni(i,j+1);
    end
  end
end


for i = TE:m-1
  for j = 1:m
    if i > TE
      CoupIni(i,j) = CoupIni(i+1,j);
    else
      CoupIni(i,j) = CoupIni(i,j)-CoupIni(i+1,j);
    end
```

```
  end
end

for i = TE:m-1
  if i > TE
    RHS(i)    = RHS(i+1);
    Cosine(i) = Cosine(i+1);
    Sine(i)   = Sine(i+1);
  else
    RHS(i) = RHS(i)-RHS(i+1);
  end
end

CoupStore   = CoupIni;
RHSStore    = RHS;
XStore      = XPoint;
RStore      = RPoint;
CosineStore = Cosine;
SineStore   = Sine;

clear RHS
clear RHSG
clear Coup
clear XData
clear RData
clear Cosine
clear Sine

for i = 1:m-1
  RHS(i,1)    = RHSStore(i);
  XData(i,1)  = XStore(i);
  RData(i,1)  = RStore(i);
  Cosine(i,1) = CosineStore(i);
  Sine(i,1)   = SineStore(i);
  for j = 1:m-1
    Coup(i,j) = CoupStore(i,j);
  end
end

CoupData.Coup      = Coup;
CoupData.CoupIni   = CoupIni;
CoupData.RHS       = RHS;
CoupData.CoupCheck = CoupCheck;
CoupData.XPoint    = XData;
CoupData.RPoint    = RData;
CoupData.Cosine    = Cosine;
CoupData.Sine      = Sine;
CoupData.WakeRHS   = 0;

end

function [ GeoData, CoupData ] = FunWake( GeoData,CoupData )
%Calculates the wake geometry and influence coefficients
```

```
XLocationRotor  = GeoData.RotorPln(1,1);
RHub            = GeoData.RHub;
RTip            = GeoData.RTip;
RotorDiameter   = RTip*2;
XData           = GeoData.XData;
RData           = GeoData.RData;
XPoint          = CoupData.XPoint;
RPoint          = CoupData.RPoint;
Cosine          = CoupData.Cosine;
Sine            = CoupData.Sine;
mCB             = GeoData.mCB;
r               = GeoData.RBldSec;
XTED            = XData(GeoData.TE+2,1);
RTED            = RData(GeoData.TE+2,1);
XTECB           = XData(mCB+1,1);
NumWakeDivision = 250;
WakeExtend      = 8;

XWakeData =
(XLocationRotor:WakeExtend*RotorDiameter/NumWakeDivision:XLocationRotor+WakeE
xtend*RotorDiameter)';

% Calculate the distance bewteen the hub and duct at the rotor. This will
% be used to determing the expansion of the wake by keeping the vortex
% sheets at the same percentage of the distance between the duct and hub.

for k = 1:numel(XWakeData)
  for j = 1:mCB+1
    for i = GeoData.TE+2:numel(XData)
      XLoc = XWakeData(k);
      if XLoc > XTED && XLoc > XTECB
        break
      elseif XData(j)>=XLoc && XData(j-1)<XLoc && XData(i)<=XLoc && XData(i-
1)>XLoc && XLoc<XTED && XLoc<XTECB
        %Interpolate between the defined geometry to get the
        %distance between the duct and the center body at the rotor
        %location
        RCBody(k,1)    = RData(j-1)+(RData(j)-RData(j-1))*((XLoc-XData(j-
1))/(XData(j)-XData(j-1)));
        RDuctBody(k,1) = RData(i)+(RData(i-1)-RData(i))*((XLoc-
XData(i))/(XData(i-1)-XData(i)));
        Dist(k,1)      = RDuctBody(k,1)-RCBody(k,1);
      elseif XLoc>XTED && XData(j)>=XLoc && XData(j-1)<XLoc
        RCBody(k,1)    = RData(j-1)+(RData(j)-RData(j-1))*((XLoc-XData(j-
1))/(XData(j)-XData(j-1)));
        RDuctBody(k,1) = RTED;
        Dist(k,1)      = RDuctBody(k,1)-RCBody(k,1);
      elseif XLoc>XTECB && XData(i)<=XLoc && XData(i-1)>XLoc
        RCBody(k,1)    = 0;
        RDuctBody(k,1) = RData(i)+(RData(i-1)-RData(i))*((XLoc-
XData(i))/(XData(i-1)-XData(i)));
        Dist(k,1)      = RDuctBody(k,1)-RCBody(k,1);
      end
    end
  end
```

```
    if k == 1
      for i = 1:numel(r)
        PercentRLoc(i,1) = (r(i)*RTip-RCBody(k,1))/(Dist(k,1));
      end
    end
    if XWakeData(k,1) < XTED || XWakeData(k,1) < XTECB
      for j = 1:numel(r)
        RWakeData(j,k) = Dist(k,1)*PercentRLoc(j,1)+RCBody(k,1);
      end
    else
      for j = 1:numel(r)
        RWakeData(j,k) = RTED*PercentRLoc(j,1);
      end
    end
end

GeoData.Test = PercentRLoc;

%% Prepare Geometry for Influence Coefficient Calculation

ex = 0.000001;

for j = 1:numel(XWakeData)-1
  for i = 1:numel(r)
    delsWake             = sqrt((XWakeData(j+1)-
XWakeData(j))^2+((RWakeData(i,j+1)-RWakeData(i,j))^2));
    dsWakeStore(i,j)     = delsWake;
    SineWakeStore(i,j)   = (RWakeData(i,j+1)-RWakeData(i,j))/delsWake;
    CosineWakeStore(i,j) = (XWakeData(j+1)-XWakeData(j))/delsWake;
    XWakePointStore(i,j) = (XWakeData(j+1)+XWakeData(j))/2;
    RWakePointStore(i,j) = (RWakeData(i,j+1)+RWakeData(i,j))/2;
    abscos               = abs(CosineWakeStore(i,j));
    if abscos > ex
      t = atan(SineWakeStore(i,j)/CosineWakeStore(i,j));
    end
    if abscos < ex
      SlopeWake(i,j) = sign(SineWakeStore(i,j))*pi/2;
    end
    if CosineWakeStore(i,j) > ex
      SlopeWake(i,j) = t;
    end
  end
end

for j = 2:numel(XWakeData)-2
  for i = 1:numel(r)
    CurveWakeStore(i,j) = (SlopeWake(i,j+1)-SlopeWake(i,j-1))/(8*pi);
  end
end

for i=1:numel(r)
  CurveWakeStore(i,1)                  = CurveWakeStore(i,2);
  CurveWakeStore(i,numel(XWakeData)-1) = CurveWakeStore(i,numel(XWakeData)-
2);
```

```
end

k = 1;

for j = 1:numel(XWakeData)-1
  for i = 1:numel(r)
    dsWake(k,1)     = dsWakeStore(i,j);
    SineWake(k,1)   = SineWakeStore(i,j);
    CosineWake(k,1) = CosineWakeStore(i,j);
    XWakePoint(k,1) = XWakePointStore(i,j);
    RWakePoint(k,1) = RWakePointStore(i,j);
    CurveWake(k,1)  = CurveWakeStore(i,j);
    k = k+1;
  end
end

%% Output Wake Geo Data

GeoData.dsWake     = dsWake;
GeoData.SineWake   = SineWake;
GeoData.CosineWake = CosineWake;
GeoData.XWakePoint = XWakePoint;
GeoData.RWakePoint = RWakePoint;
GeoData.CurveWake  = CurveWake;

%% Calculate Influence Coefficient Matrix

for i = 1:numel(XWakePoint)
  for j = 1:numel(XPoint)
    [u v]         = FunVelocity(XPoint(j),RPoint(j),XWakePoint(i),
RWakePoint(i));
    CoupWake(j,i) = (u*Cosine(j)+v*Sine(j))*dsWake(i);
  end
end

CoupData.CoupWake = CoupWake;

end

function [ GeoData, CoupData ] = FunCoupSource( GeoData,CoupData )
%Calculates the influence coefficients of point source approximating the
%blade section drag.

XLocationRotor  = GeoData.RotorPln(1,1);
RRotorPln       = GeoData.RotorPln(1:end,2);
drRotorPln      = GeoData.drRotPln;
XPoint          = CoupData.XPoint;
RPoint          = CoupData.RPoint;
Cosine          = CoupData.Cosine;
Sine            = CoupData.Sine;
RSupportPln     = GeoData.RSupPln;
XLocSupport     = GeoData.SupportX;
drSupPln        = GeoData.drSupPln;
```

```
for i = 1:numel(drRotorPln)
  for j = 1:numel(XPoint)
    [u, v, ~, ~]    = FunSourceVelocity(XPoint(j),RPoint(j),XLocationRotor,
RRotorPln(i));
    CoupSource(j,i) = ((u*Cosine(j)+v*Sine(j)))*drRotorPln(i);
  end
end

for i = 1:numel(RSupportPln)
  for j = 1:numel(XPoint)
    [u, v, ~, ~]       = FunSourceVelocity(XPoint(j),RPoint(j),XLocSupport,
RSupportPln(i));
    CoupSourceSup(j,i) = ((u*Cosine(j)+v*Sine(j)))*drSupPln(i);
  end
end

CoupData.CoupSource    = CoupSource;
CoupData.CoupSourceSup = CoupSourceSup;

end

function [ Output ] = FunSolve( CoupData, Output, GeoData, UInf )
%Solves for the surface vorticity distribution

coup = CoupData.Coup;
rhs  = CoupData.RHS;

for j = 1:numel(GeoData.RBldSec):numel(GeoData.XWakePoint)
    for i = 1:numel(GeoData.RBldSec)
        WakeGammas(j+i-1,1) = Output.WakeGammas(i);
    end
end

GammaWake = WakeGammas;
WakeRHS   = CoupData.CoupWake*GammaWake;
RotorRHS  = CoupData.CoupSource*Output.BldSource;
SupRHS    = CoupData.CoupSourceSup*Output.SupSource;
RHS       = rhs-(WakeRHS./UInf)-(RotorRHS./UInf)-(SupRHS./UInf);
gamma     = coup\RHS;
Cp        = 1-abs(gamma).^2;

Output.Gamma     = gamma;
Output.Cp        = Cp;
Output.GammaWake = WakeGammas;

end

function [ Power GeoData CoupData FinalOutput ] = FunIterate2( GeoData,
CoupData, OpCond, InitialOutput )
%Iterates to find a solution for the blade circulation for a DAWT.
```

```
UInf        = OpCond.UInfBegin:OpCond.UInfStep:OpCond.UInfEnd;
Theta0      = OpCond.Theta0Begin:OpCond.Theta0Step:OpCond.Theta0End;
RPM         = OpCond.RPMBegin:OpCond.RPMStep:OpCond.RPMEnd;
Rho         = OpCond.Rho;
FinalOutput = InitialOutput;
RtrR        = GeoData.RotorPln(1:end,2);
EtaMech     = OpCond.EtaMech;
Relax       = 0.03;
NumIter     = 100;
ErrorTol    = 0.001;

disp 'Iterating ...'

for k = 1:numel(RPM)
  for j = 1:numel(Theta0)
    for i = 1:numel(UInf)

      CpOld = InitialOutput.Cp;
      CpNew = zeros(numel(CpOld),1);
      tol   = ErrorTol.*ones(numel(RtrR),1);
      diff  = ones(numel(RtrR),1);
      UOld  = zeros(numel(RtrR),1);
      Iter  = 1;

      while Iter<NumIter+1

%% Calculate the Velocities at the Rotor Plane

        [ FinalOutput ] = FunRtrVel( GeoData, UInf(i), FinalOutput );

        UNew = FinalOutput.U;

%% Calculate the Vorticity based on the Blade Section;

        [ FinalOutput ] = FunBladeGamma( FinalOutput, GeoData, RPM(k),
Theta0(j), Rho );

%% Solve for Duct and Hub Vorticies including wake effects

        [ FinalOutput ] = FunSolve( CoupData,FinalOutput,GeoData, UInf(i) );

        CpNew             = FinalOutput.Cp;
        FinalOutput.CpNew = CpNew;
        FinalOutput.CpOld = CpOld;
        FinalOutput.UNew  = UNew;
        FinalOutput.UOld  = UOld;
        diff              = abs((UNew-UOld)./UOld);

        if diff < tol
          WindSpeed   = UInf(i);
          GlobalPitch = Theta0(j);
```

```
          DispStr     = strcat(['Iteration converged for a wind speed of
',num2str(WindSpeed),' m/s at a pitch angle of ',num2str(GlobalPitch),'
deg.']);
          disp(DispStr)
          break
        else
          Iter  = Iter+1;
          UOld  = UNew;
          CpOld = CpNew.*(Relax)+(1-Relax).*CpOld;

        end
      end

      if Iter>NumIter
        WindSpeed   = UInf(i);
        GlobalPitch = Theta0(j);
        DispStr     = strcat(['Iteration did not converge for a wind speed of
',num2str(WindSpeed),' m/s at a pitch angle of ',num2str(GlobalPitch),'
deg.']);
        disp(DispStr)
      end

      dr         = GeoData.drRotPln;
      r          = GeoData.RotorPln(1:end,2);
      Chord      = GeoData.BldChord;
      NumBlades  = GeoData.NumBld;
      RTip       = GeoData.RTip;
      CL         = FinalOutput.BldSectionCLCD(1:end,1);
      CD         = FinalOutput.BldSectionCLCD(1:end,2);
      Phi        = FinalOutput.BldPhi.*pi/180;
      U          = FinalOutput.U;
      Omega      = FinalOutput.Omega;
      Vtheta     = FinalOutput.Vtheta;

      DT = 0;
      DQ = 0;

      for l = 1:numel(r)
        Urelsqrd(l) = U(l)^2+(Omega*r(l)+Vtheta(l))^2;
        DTi(l,1)    =
NumBlades*0.5*Rho*Urelsqrd(l)*(CL(l)*cos(Phi(l))+CD(l)*sin(Phi(l)))*Chord(l)*
dr(l);
        DQi(l,1)    = NumBlades*0.5*Rho*Urelsqrd(l)*(CL(l)*sin(Phi(l))-
CD(l)*cos(Phi(l)))*Chord(l)*r(l)*dr(l);
        DT          = DT+DTi(l);
        DQ          = DQ+DQi(l);
      end

      PTotal = DQ*Omega;

      Power{k,1}.W{i,j}           = sqrt(Urelsqrd)';
      Power{k,1}.T{i,j}           = DTi;
      Power{k,1}.Q{i,j}           = DQi;
      Power{k,1}.P{i,j}           = DQi.*(Omega);
```

```
      Power{k,1}.Thrust(i,j)       = DT;
      Power{k,1}.PowerFinal(i,j)   = PTotal*EtaMech;
      Power{k,1}.Torque(i,j)       = DQ;
      Power{k,1}.a{i,j}            = (UInf(i)-U)./UInf(i);
      Power{k,1}.TSR{i,j}          = Omega*RTip/UInf(i);
      Power{k,1}.BodyGamma{i,j}    = FinalOutput.Gamma;
      Power{k,1}.WakeGamma{i,j}    = FinalOutput.GammaWake;
      Power{k,1}.BladeSource{i,j} = FinalOutput.BldSource;

    end
  end

  RPMStr = num2str(RPM(k));
  StrOut = strcat(['RPM ',RPMStr,' completed']);

  disp(StrOut)
end
end

function [ Output ] = FunRtrVel( GeoData,UInf, Output )
%Calculates the velocity at the rotor plane

XPoint     = GeoData.XPoint;
RPoint     = GeoData.RPoint;
XWakePoint = GeoData.XWakePoint;
RWakePoint = GeoData.RWakePoint;
ds         = GeoData.ds;
dsWake     = GeoData.dsWake;
drRotor    = GeoData.drRotPln;
GamWake    = Output.GammaWake;
RtrR       = GeoData.RotorPln(1:end,2);
RtrX       = GeoData.RotorPln(1:end,1);
Gamma      = Output.Gamma;
BldSource  = Output.BldSource;
SupSource  = Output.SupSource;
RSupport   = GeoData.RSupPln;
XSupport   = GeoData.SupportX;
drSup      = GeoData.drSupPln;
TE         = GeoData.TE;
U          = UInf;

for i = 1:numel(XPoint)-1
  if i<=TE
    GammaPoint(i,1) = Gamma(i);
  elseif i == TE+1
    GammaPoint(i,1)   = Gamma(TE);
    GammaPoint(i+1,1) = Gamma(TE+1);
  elseif i >= TE+2
    GammaPoint(i+1,1) = Gamma(i);
  end
end

dsGamma = GammaPoint.*ds;
dsGamWake = GamWake.*dsWake;
```

```
dsSigSup = SupSource.*drSup;

for j = 1:numel(XPoint)
  for i = 1:numel(RtrR)
    [ u(i,j),v(i,j), ~, ~ ] = FunVelocity(
XPoint(j),RPoint(j),RtrX(i),RtrR(i) );
  end
end

for j = 1:numel(XWakePoint)
  for i = 1:numel(RtrR)
    [ uWake(i,j),vWake(i,j), ~, ~ ] = FunVelocity(
XWakePoint(j),RWakePoint(j),RtrX(i),RtrR(i) );
  end
end

for i = 1:numel(RtrR)
  uRotorSig(i,1)=-0.5*BldSource(i)*drRotor(i);
end

for j = 1:numel(RSupport)
  for i = 1:numel(RtrR)
    [ uSup(i,j),vSup(i,j), ~, ~ ] = FunSourceVelocity(
XSupport,RSupport(j),RtrX(i),RtrR(i) );
  end
end

DeltaU     = -u*dsGamma;
DeltaV     = v*dsGamma;
DeltaUWake = uWake*dsGamWake;
DeltaVWake = vWake*dsGamWake;
DeltaUSup  = -uSup*dsSigSup.*0;
DeltaVSup  = vSup*dsSigSup.*0;

%% Outputs

Output.DeltaU     = DeltaU;
Output.DeltaV     = DeltaV;
Output.DeltaUWake = DeltaUWake;
Output.DeltaVWake = DeltaVWake;
Output.DeltaURtrS = uRotorSig;
Output.U          = U+DeltaU+DeltaUWake+uRotorSig+DeltaUSup;
Output.V          = DeltaV+DeltaVWake+DeltaVSup;
Output.GammaPoint = GammaPoint;

end

function [ Output ] = FunBladeGamma( Output, GeoData, RPM,Theta0,Rho )
%Calculates the blade section circulation

Pitch    = GeoData.BldPitch*pi/180;
Chord    = GeoData.BldChord;
NumBld   = GeoData.NumBld;
RHub     = GeoData.RHub;
```

```
RTip     = GeoData.RTip;
R        = GeoData.RotorPln(1:end,2);
r        = GeoData.RBldSec.*RTip;
U        = Output.U;
Omega    = RPM*pi/30;
Theta0   = Theta0;
AeroData = GeoData.AeroData;
Rho      = Rho;
NumSup   = GeoData.NumSup;
SupCD    = GeoData.SupCD;
RSup     = GeoData.RSupPln;
Relax    = 0.03;

%% Calculate inflow parameters and blade sectional circulation distribution

tol    = 0.001*ones(numel(R),1);
diff2  = ones(numel(R),1);
Vtheta = zeros(numel(R),1);

  for i = 1:numel(R)
    Phi(i,1)              = atan(U(i)/(Omega*R(i)+Vtheta(i)));
    Alpha(i,1)            = Phi(i)-(Pitch(i)+Theta0*pi/180);
    [ cl(i,1), cd(i,1) ] = FunLookUpCLCD( Alpha(i,1), AeroData{i,1} );
    BldSectionGamma(i,1) =
0.5*Chord(i)*cl(i,1)*sqrt(U(i)^2+(Omega*R(i)+Vtheta(i))^2);
    BldSectionSigma(i,1) =
NumBld/(4*pi*R(i))*cd(i,1)*sqrt(U(i)^2+(Omega*R(i)+Vtheta(i))^2);
    SupSectionSigma(i,1) =
NumSup/(4*pi*RSup(i))*SupCD*sqrt(U(i)^2+(Vtheta(i))^2);
  end

%% Calculate the strength of the wake vortex sheets

for i = 1:numel(R)
  if i == 1
    UmAVG(i,1) = U(i);
  else
    UmAVG(i,1) = 0.5*(U(i-1)+U(i));
  end
    UmAVG(numel(R)+1,1) = U(numel(R));
end

for i = 1:numel(UmAVG)
  if i == 1
    SheetGamma(i,1) = 1/UmAVG(i,1)*(-
0.5*((NumBld*BldSectionGamma(i,1))/(2*pi*RHub))^2+(Omega*NumBld*BldSectionGam
ma(i,1))/(2*pi));
  elseif i == numel(UmAVG)
    SheetGamma(i,1) = 1/UmAVG(i,1)*(-0.5*((NumBld*BldSectionGamma(i-
1,1))/(2*pi*RTip))^2+(Omega*NumBld*BldSectionGamma(i-1,1))/(2*pi));
  else
    SheetGamma(i,1) = 1/UmAVG(i,1)*((-
0.5*(NumBld/(2*pi*r(i)))^2)*(BldSectionGamma(i,1)^2-BldSectionGamma(i-
1,1)^2)+...
```

```
                        (Omega*NumBld)/(2*pi)*(BldSectionGamma(i,1)-
BldSectionGamma(i-1,1)));
  end
end

%% Set Output Parameters

Output.BldPhi          = Phi.*180/pi;
Output.BldSectionAlpha = Alpha.*180/pi;
Output.BldSectionCLCD  = [cl cd];
Output.BldCirc         = BldSectionGamma;
Output.BldSource       = BldSectionSigma;
Output.SupSource       = SupSectionSigma;
Output.WakeGammas      = SheetGamma;
Output.Omega           = Omega;
Output.Rho             = Rho;
Output.Vtheta          = Vtheta;

end

function [ cl, cd ] = FunLookUpCLCD( Alpha, AeroData )
%Looks up the CL and CD values based on the AoA

Alpha=Alpha*180/pi;

AlphaArray=AeroData(1:end,1);
CLArray=AeroData(1:end,2);
CDArray=AeroData(1:end,3);

for i=1:numel(AlphaArray)-1
  if AlphaArray(i+1)>=Alpha && AlphaArray(i)<=Alpha
    LowIndex=i;
  end
end

HighIndex=LowIndex+1;

CLHigh=CLArray(HighIndex);
CLLow=CLArray(LowIndex);
CDHigh=CDArray(HighIndex);
CDLow=CDArray(LowIndex);
AlphaHigh=AlphaArray(HighIndex);
AlphaLow=AlphaArray(LowIndex);

%Interpolation between the values to find the lift and drag coefficients at
%alpha

cl=((((CLHigh-CLLow)/(AlphaHigh-AlphaLow))*(Alpha-AlphaLow))+CLLow)*1.0;
cd=((((CDHigh-CDLow)/(AlphaHigh-AlphaLow))*(Alpha-AlphaLow))+CDLow)*1.0;

end
```

```
function [ u,v, K, E ] = FunVelocity( xl,rl,xm,rm )
%FunVelocity calculates the induced velocity at point m due to a vortex at
%point l.

x       = (xl-xm)/rm;
r       = rl/rm;
a       = x^2+(r-1)^2;
k       = sqrt(4*r/(x^2+(r+1)^2));
[ K,E ] = ellipke(k^2);
cons    = 0.5/(pi*rm*sqrt(x^2+(r+1)^2));
u       = -cons*(K-(1+2*(r-1)/a)*E);
v       = cons*x/r*(K-(1+2*r/a)*E);

end

function [ usig,vsig, K, E ] = FunSourceVelocity( xl,rl,xm,rm )
%Calculates the induced velocity due to the source distribution

x = (xl-xm)/rm;
r = rl/rm;
a = x^2+(r-1)^2;
k = sqrt(4*r/(x^2+(r+1)^2));
[ K,E ] = ellipke(k^2);
cons = 0.5/(pi*rm*sqrt(x^2+(r+1)^2));
usig = cons*((2*x*E)/(a));
vsig = cons*(K-(1-(2*r*(r-1))/a)*E);

end
```