Continuous Curved Near Wake Analysis for a Lifting Surface

by

Melinda Dee Godwin

B.S., Aeronautical Engineering, University of Maryland (1989)

Submitted to the Department of Aeronautics and Astronautics
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE IN AERONAUTICS AND ASTRONAUTICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1991

© Massachusetts Institute of Technology, 1991. All rights reserved.
CONTINUOUS CURVED NEAR WAKE ANALYSIS FOR A LIFTING SURFACE

BY

MELINDA DEE GODWIN

Submitted to the department of Aeronautics and Astronautics on July 22, 1991, in partial fulfillment of the requirements for the Degree of Master of Science in Aeronautical Engineering.

Abstract

Accurate modeling of the near wake remains a significant problem in helicopter rotor aerodynamics. Motivated by the inability of present models to achieve close agreement with experimental near-wake geometries, an improved representation of the near-wake of a one-bladed rotor is developed. Although this problem has been addressed by others, close agreement with experimental wake geometries has not been achieved.

A major problem in matching experimental results using filamentary models involves specification of the proper vortex core size for a curved point vortex. This arbitrary core size is needed to eliminate the logarithmic induced velocity singularity and solutions have been shown to vary significantly depending on the chosen value. The present study uses a continuous vortex sheet rather than discrete filaments, avoiding the need for a finite core size. The model represents a one-bladed hovering rotor trailing a continuous vortex sheet. A free-wake analysis using the Biot-Savart law is then applied to determine the wake geometry in a fixed plane perpendicular to the initial position of the blade. The singularity associated with the circulation distribution at the tip is represented using a self-similar solution developed by Pullin. Several circulation distributions are studied, including an elliptical distribution and a distribution typical of a hovering rotor. For the latter, the intent is to study the character of the vortex roll-up and to determine if, counter to experimental evidence, a mid-span vortex occurs as it has using past models.

Results from the current study may be used in conjunction with discrete models to predict the proper core size to be used. In this manner, the greater computational efficiency of discrete models may be retained while obtaining the accuracy of a more rigorous near-wake
representation. This research could also be extended to forward flight and incorporated in an existing full-wake analysis code as the near-wake component.
# Table of Contents

Abstract ................................................................................................................. 3  
Acknowledgment ................................................................................................. 6  
List of Figures ....................................................................................................... 7  
Nomenclature ....................................................................................................... 8  
1. Introduction ...................................................................................................... 9  
   1.1 Historical Note ........................................................................................... 9  
   1.2 Previous Rotor Wake Research ................................................................ 9  
   1.3 Outline of the Present Research ................................................................ 11  
2. Description of the Model .................................................................................. 12  
   2.1 Coordinate System .................................................................................... 13  
   2.2 Non-Dimensional Parameters .................................................................. 15  
   2.3 Definition of Linear Region ...................................................................... 17  
   2.4 Exponential Stretching Technique ............................................................ 19  
   2.5 Pullin Similarity Solution .......................................................................... 22  
   2.6 Vortex Sheet Model ................................................................................... 25  
3. Analytical Method ............................................................................................. 26  
   3.1 Application of the Biot-Savart Law ......................................................... 26  
   3.2 Axial Velocity Component ....................................................................... 31  
   3.3 Radial Velocity Component ..................................................................... 39  
   3.4 Tangential Velocity Component ............................................................... 43  
   3.5 Self-Induced Velocity of the Tip Vortex ................................................... 43  
   3.6 Smoothing .................................................................................................. 44  
4. An Elliptically Loaded Blade ............................................................................ 45  
   4.1 Effect of Smoothing .................................................................................. 46  
   4.2 Dependence of Wake Solution on Time Step ........................................... 48  
   4.3 Dependence of Solution on Linear Region Size ...................................... 50  
   4.4 Calculated Wake Geometry ....................................................................... 52  
5. Translating Circular Disk ................................................................................. 56  
6. Typical Hovering Rotor Loading ...................................................................... 60  
8. Conclusions and Future Recommendations .................................................. 66  
References ............................................................................................................ 69  
Appendix - Computer Programs .......................................................................... 71  


Acknowledgment

My sincerest thanks to my advisor, Mark Drela, for all the time and effort he spent assisting me in developing this thesis. His wisdom and patience was greatly appreciated.

I would also like to thank the NSF PYI Program, along with Earl Murman and NASA Langley (NASA Grant NAG-1-507) for making this research and my masters possible.

Finally, I would like to thank my husband Mark for his support, assistance, and understanding through the final stages of this thesis.
List of Figures

Figure 2.1-1 Basic Coordinate System .................................................................14
Figure 2.1-2 Definition of s Coordinate ...............................................................14
Figure 2.1-3 Top View of Rotor ...........................................................................15
Figure 2.3-1 Determination of Linear Region Size .............................................18
Figure 2.4-1 Filament Distribution ........................................................................20
Figure 2.5-1 Pullin Solution Applied to Elliptically Loaded Blade .................24
Figure 3.1-1 Problem Set-Up ..............................................................................28
Figure 3.1-2 Application of Biot-Savart Law .....................................................30
Figure 4.1-1 Effect of Smoothing ........................................................................47
Figure 4.2-1 Effect of Time Step .........................................................................49
Figure 4.3-1 Effect of Linear Region Size ..............................................................51
Figure 4.4-1 Wake Profiles for the Elliptically Loaded Blade .........................53
Figure 4.4-2 Wake Profiles for the Elliptically Loaded Blade .........................54
Figure 4.4-3 Wake Profile for the Elliptically Loaded Blade, $\Psi = 45^\circ$ ....55
Figure 5-2 Circulation & Wake Distribution for the Oblate Spheroid ............58
Figure 5-3 Downwash Distribution for the Oblate Spheroid ...........................59
Figure 6-1 Typical Hovering Rotor Circulation & Wake Distribution ............61
Figure 6-2 Wake Profile for Typical Hover Circulation, $\Psi = 10^\circ$ ............62
Figure 6-3 Wake Profile for Typical Hover Circulation, $\Psi = 20^\circ$ ............63
Figure 6-4 Wake Profile for Typical Hover Circulation, $\Psi = 30^\circ$ ............64
Figure 6-5 Wake Profile for Typical Hover Circulation, $\Psi = 45^\circ$ ............65
Nomenclature

dh  segment of integration strip which produces a velocity at a point in space.
l  radius vector from dh to a point in space
r  radial displacement corresponding to s
R  radial displacement of marker corresponding to s
s  location where circular integration strips are located
S  location along vortex sheet where induced velocity is calculated
z  vertical displacement of wake corresponding to s
Z  vertical displacement of marker corresponding to s

2δ  width of linear region
Ψ  rotor azimuthal angle, radians
Γb  bound circulation
1. Introduction

1.1 Historical Note

Modeling the aerodynamics of a rotor has always been a difficult task due to the geometric and aerodynamic complexity of the wake. In the early 1900's, Kutta and Joukowsky related lift to vorticity in the flow, providing the fundamental basis for vortex modeling of rotor wakes. This approach utilizes the Biot-Savart law which allows determination of the induced velocity at any position in the wake and modeling of the wake deformation and displacement in unsteady flows.

1.2 Previous Rotor Wake Research

Many different analyses and models evolved using vortex modeling, including the vortex sheet and filamentary models discussed below. The most recent development in vortex modeling has been free-wake analysis, defining the complete wake geometry using the calculated induced velocities. Free-wake analyses are physically more correct than previous models and, for that reason, many recent studies have been focusing on utilizing and optimizing this method of analysis.

Motivated by the inability of existing models to accurately predict near-wake geometry, this thesis focuses on an improved representation of the near-wake of a one-bladed rotor (the near wake is defined to extend from $\Psi = 0^\circ$ to $\Psi = 45^\circ$). Although there are a variety of near wake analysis programs using filamentary and vortex sheet models, the need to specify an arbitrary core size limits their general applicability. An example is an analysis by Miller$^1$ where the near wake is modeled
as a series of semi-infinite vortex elements which are then rolled up into two or three discrete vortices (before the first blade is encountered) according to Betz criteria. Since the mid-span and root vortices which result have not been detected in any of the relevant experimental studies, Miller states that an "exact prediction of the wake structure" can not be determined from the results, and the roll-up technique should only be viewed as a "convenient computational technique to account for the effects of the inboard trailing vorticity".

A vortex filament model developed by Brower considers the entire rotor wake. Brower represents the near wake as straight vortex segments and includes a correction for viscous core size (causing the induced velocity to approach zero rather than infinity as the vortex element is approached). Since Brower uses straight filaments, he adds a correction factor for self-induced effects, since actual vortex filaments leaving the blade are curved and will induce a velocity on themselves. Brower found that the choice of core size had a significant effect on the induced velocities at nearby points, and that the choice of core size should be related to the computational mesh size. Most papers written on filamentary models note difficulties in selecting a proper core size due to the lack of a physically correct predictive model and an incomplete understanding of how core size is related to specific flows.

Vortex sheet models for near wake analysis using a free-wake approach have also been developed. An example is the model developed by Tanuwidjaja. The near-wake is modeled as a vortex sheet with tip and root vortices. This method uses the Biot-Savart law and divides the blade and wake into nodes which define discrete segments, and sheet elements, which form the vortex sheet. The
induced velocities, however, are not calculated at these nodes, due to singularity problems at the discrete sheet elements. Instead, new locations between the nodes called "centers" are defined. The induced velocity is then calculated at these centers and linear interpolation is used to find the velocities at the nodes. Since, however, Tanuwidjaja uses discrete vortex sheet elements, the arbitrariness that stems from the need to specify a finite core size is not eliminated.

1.3 Outline of the Present Research

In this study, unlike the iterative analyses by Miller, the model is given a circulation distribution that is assumed correct, and it then calculates the induced velocities and resulting wake geometry by integrating across the wake using the Biot-Savart law. The induced velocities in the wake are calculated at sequential increments in time, so "snap-shots" of the resulting wake geometry are obtained downstream of the rotor. This process is repeated until an azimuthal angle of $\Psi=45$ degrees is reached. This is a continuous model, therefore no finite core size (except for the tip vortex) is needed, which eliminates the arbitrariness in the solution due to core size specification.

As previously stated, the present research considers the near wake of a one bladed rotor using a free-wake analysis. The entire wake geometry is not considered, since the intermediate and far wakes are comparatively well understood (although computationally time consuming to analyze). Rather it is the intent to focus on a more physically correct representation of the near wake, yielding results which more closely match experimental data. By obtaining a more
accurate near wake geometry, this program may be utilized in the following ways.

1. To assist in selecting the appropriate core size to be used in filamentary models where core size is included as a parameter.
2. To test other wake geometry programs to investigate the accuracy of their solutions.
3. As the near-wake component in a full-wake analysis code.

Miller states, "It will never be possible to model exactly the dynamics and aerodynamics of a system as complex as a rotary wing vehicle. ....To this end the simplest forms of analysis, which retain the essence of the phenomena involved, are of greatest value during the crucial stages of design and flight evaluation."8 The difficulty lies in knowing how to simplify a model while maintaining an acceptable level of accuracy. While the model outlined in this thesis may be physically more correct, it may not be practical to efficiently implement in free-wake codes modeling the entire wake. The model is more computationally time-consuming and mathematically difficult than models such as Miller's fast free wake analysis. The best alternative would be to use the knowledge gained from this research to improve the results of more simplistic models, thereby achieving good results while minimizing computational expense.

2. Description of the Model

The present axisymmetric model represents the near wake of a one bladed rotor as a vortex sheet rather than by a series of discrete
filaments. This sheet is composed of 500 integration intervals which contain the vorticity in the wake. In reality these integration strips follow helical paths downstream of the rotor. Miller has shown, however, that the effect of the inclination of these integration intervals on the induced velocity is negligible. This model is therefore developed using integration intervals which are planar circles perpendicular to, and centered about the z-axis (see figure 3.1-1). This results in an axisymmetric flow pattern where the axial, radial, and tangential velocity components are independent of $\Psi$. Since the circular integration intervals remain perpendicular to the axis of symmetry as the wake convects downstream, the tangential component of the velocity will always be negligible. The purpose of this model is to investigate how the wake deforms downstream of the rotor blade, and to develop insight into the question of whether or not a discrete mid-span vortex is formed. First, a description of the coordinate system is provided, followed by an explanation of the primary non-dimensional parameters. The "linear region" concept used to eliminate the computational singularity in the induced velocity equations is then described. Next, the exponential stretching technique used to define integration strips (containing the vorticity in the wake) is discussed along with the similarity solution developed by Pullin to handle the tip singularity. Finally, a description of the vortex sheet will be given.

2.1 Coordinate System

The coordinate system for the wake geometry used in this analysis is three-dimensional, consisting of radial (r), azimuthal ($\Psi$) and vertical (z) components, as shown in figure 2.1-1.
The actual wake deformation is viewed in two dimensions in the $r$-$z$ plane (as shown in the side view above), producing two-dimensional, time dependent wake profiles. To designate locations along the profile, a sheet coordinate, "s", is introduced.

To simplify the form of the equations, it is assumed that the two-dimensional wake is always viewed from a fixed position ($\Psi=0^\circ$) with
the blade moving away at a positive angle. This approach eliminates additional angular velocity terms.

**Figure 2.1-3 Top View of Rotor**

2.2 Non-Dimensional Parameters

All of the parameters used in the equations are non-dimensionalized to insure flexibility in running test cases and generalizing results.

*Length*

The sheet geometry is parameterized in terms of its arc length $s$, non-dimensionalized by the blade radius. The sheet geometry is
therefore defined by the coordinates r(s) and z(s), which are also non-dimensionalized by the blade radius.

In the numerical scheme, the marker locations on the sheet where the velocity is calculated are denoted by uppercase symbols S, R, and Z, while the locations of the integration strips containing the vorticity are denoted by lowercase s, r, and z.

\[ R = \frac{\text{marker r-location}}{\text{blade radius}}, \quad Z = \frac{\text{marker z-location}}{\text{blade radius}} \]

**Angle**

The angle Ψ refers to the azimuthal angle (in radians) about the axis of rotation extending from -π to π. This angle is shown in Figure 2.1-3.

**Time**

Time is non-dimensionalized by \( \Omega \), the rotational speed of the rotor (i.e. \( t^* = t \Omega \)). As an example, an azimuthal angle of \( \pi/4 \) radians corresponds to a non-dimensional time \( t^* = \pi/4 \), or more generally, \( \Delta t^* = \Delta \Psi \). The near wake is broken into 45 increments, and therefore the outer time loop in the program moves through 45 one-degree time steps.

**Velocity**

The non-dimensional velocity is simply the ratio of the non-dimensional length and time described above:
\[
V^* = l^*/t^* = \frac{l}{(\text{blade radius})(t)(\Omega)} = \frac{V}{(\Omega)(\text{blade radius})}
\]

2.3 Definition of Linear Region

The linear region is a specified area on either side of a marker within which the equations are modified to analytically remove the singularity that occurs when an integration interval approaches a marker location.

It is necessary to integrate equations 3.14 and 3.30 over the sheet length \( s \), since the \( r \) and \( z \) terms are both functions of \( s \). To facilitate this integration, \( r \) and \( z \) terms are linearized with respect to \( s \). This is accomplished through the following relations:

\[
\begin{align*}
R - r &= K_1(S - s) \\
Z - z &= K_2(S - s)
\end{align*}
\]

where \( K_1 = \frac{dr}{ds} \bigg|_{s=S} \) \quad \( K_2 = \frac{dr}{ds} \bigg|_{s=S} \)

These linear relations are then used to eliminate \( r \) and \( z \) in the equation, resulting in an equation dependent only on \( s \). This allows the singularities in the integral to be analytically removed. Since these equations (i.e. 3.14, 3.30) can only be defined over small portions of the wake (because of the linearizations above), the modified form of the equation is only used when an integration interval is in close proximity to a marker (in the linear region) and the unmodified original equations are used elsewhere. The size of the linear region is a variable, and it
will be shown that the wake geometry is independent of the precise value chosen as long as the linearity assumption for \( r \) and \( s \), and \( z \) and \( s \) remains valid. For the test cases shown, the lesser of the two distances from the marker being considered to the markers on either side is chosen as \( \delta \) and the linear region is twice this distance (half on each side of the marker as shown in figure 2.3-1).

\[
\delta = \text{MIN}(0.02, 0.03) = 0.02
\]

\[
S_{\text{low}} = S(j) - \delta
\]

\[
S_{\text{high}} = S(j) + \delta
\]

width of linear region = \( 2\delta \)

Figure 2.3-1 Determination of Linear Region Size

In summary, the linear region represents a specified area about each marker within which the governing equations are modified to eliminate the singularity which would otherwise occur as an integration interval approaches a marker. The only difference between the
modified and unmodified equations is the elimination of higher order terms.

2.4 Exponential Stretching Technique

*Layout of Markers*

An exponential stretching technique is used to construct the initial lay-out of the markers (i.e. the locations where induced velocities are calculated). The stretching technique requires as input the desired number of markers and the desired distance between the first two and the last two markers. The routine then exponentially stretches between the second and the second to last markers. In general, a large distance between the first two points is specified and a small distance between the last two points. This results in a low population of markers near the root and a high density layout at the tip where, due to the tip vortex, the curvature of the wake is greatest (150 markers are generally used).

The vortex core, which is used as a filament and a marker, is not located through this technique, but through a method outlined in section 2.5.

*Layout of Integration Strips Containing the Wake Circulation*

The layout of the integration strips, unlike the layout for the markers, is modified at each time step and for each marker. At each time step, the sheet is broken into integration intervals to account for the vorticity in the wake. These intervals need to be denser near the tip (due to the strong vorticity and roll-up) and near the marker under consideration (due to the possibility of a singularity point being approached). To obtain a good distribution, a separate filament
distribution is established for each marker as it is considered. This layout is accomplished by utilizing an exponential stretching routine twice, once on each side of the marker. An example is pictured and described below.

\[ x = \text{marker} \]
\[ \bullet = \text{integration intervals} \]

\[ S = .3 \]

Figure 2.4-1 Filament Distribution

The lay-out of the filaments in region 1 are dictated by specifying the spacing between the first two and last two integration intervals and by specifying the number of intervals to be placed in this region (the filament lay-out in region 3 is obtained in an identical fashion). In the "linear region" (i.e. region 2), the layout of the filaments is slightly different. In this region, 6 points are always evenly distributed, keeping the number of integration strips constant in this area. This method assures that the number of strips within the linear region remains constant, or increases or decreases monotonically (see step 4 on the following page). If the number of integration intervals is allowed to
randomly fluctuate, instabilities occur in the wake due to truncation errors. These truncation errors arise due to the loss of higher-order terms, which occurs because the equations used in the linear region have been modified to handle the singularities which arise in this area. The lay-out of the filaments can be summarized as follows:

1. The linear region variable (δ) is added and subtracted from each side of the marker.

2. The area from the root to the left end of the linear region is stretched, and integration strips are located.

3. The area from the right end of the linear region to the tip is stretched and integration strips are located.

4. Six integration strips are located in the linear region, equal distances apart. At the root and the tip, where the linear region is only one-sided, 4 integration strips are used.

5. The midpoints between all these locations are then taken, starting from the root of the blade. These midpoints define the locations of the integration intervals.

6. This 5 step process is then repeated for each marker, at every time iteration.
2.5 Pullin Similarity Solution

Due to the singular roll-up at the outer edge of the sheet, a discrete tip vortex forms instantaneously. Applying the Biot-Savart law directly without accounting for this singularity leads to erroneous results. One way to resolve this singularity is to use the self-similar solution developed by Pullin.\textsuperscript{10,11} Pullin considers a semi-infinite vortex sheet with $\Gamma=2a|R|^{1/2}$ (where $R$ is the location along the blade, and "a" is a scaling parameter), and develops a self-similar solution which produces an initial wake geometry and circulation distribution past a certain matching point (in this case $r/R=0.95$) at time $\tau=0+$. Pullin derives a similarity solution where:

$$Z_0(\Gamma,\tau) = (at)^{2/3} \zeta(\lambda)$$  \hspace{1cm} (1)

and

$$\lambda = \frac{\Gamma}{(a^{4/3} t^{1/3})}$$  \hspace{1cm} (2)

$\zeta$, $\lambda$: similarity parameters

$a$, $t$: scaling parameters

$Z_0$: dimensional self-similar shape function for the sheet.

Using Pullin's solution along with initial values for $\zeta$ and $\lambda$ (obtained by choosing a matching point along the vortex sheet), the above two equations can be solved for $t$ and $a$ (the scaling parameters), yielding the initial circulation distribution and wake geometry beyond $r=0.95$ for
\( \tau = 0^+ \). An example is given below for the circulation distribution \( \Gamma(r) = 0.02 \sqrt{1-r^2} \), where \( \zeta_o = -3.24R \) and \( \lambda_o = 3.6 \), and \( s = 0.95 \) where \( s \) is the location along the sheet defined as the matching point.

(1) Substitute \( \Gamma \) and \( Z \) into (1) and (2):

\[
\frac{Z_o^{3/2}}{\zeta^{3/2}} \cdot \frac{t}{a} = \frac{((1-s)^{3/2})}{(3.24)^{3/2} \cdot a} \quad (1a)
\]

Note: \( (1-s) \) is necessary because Pullin has \( \zeta = 0 \) at the tip, and the coordinate system in this thesis is opposite.

\[
t = \frac{\Gamma^3}{(\lambda^3 \cdot a^4)} = \frac{0.02 \sqrt{1-(1-s)^2}^3}{(3.6)^3 \cdot a^4} \quad (1b)
\]

(2) Solve (1a) and (1b) for \( t \) and \( a \). In this case \( t = 0.1373 \) and \( a = 0.01396 \).

(3) Using \( t \) and \( a \), the wake geometry and circulation distribution are defined beyond 95% blade radius by scaling Pullin's solution accordingly (see figure 2.5-1).
Figure 2.5-1 Pullin's Solution Applied to an Elliptically Loaded Blade
Determine the tip strength and location. Pullin gives the total value of $\lambda$ for the four spirals and the core as 1.342. By dumping this circulation into the core and re-locating the core (although in this case the amount the core is moved is negligible), the core strength is found to be $\lambda=1.342$ and its location is $\zeta = -.308\, e_r, \, \eta = .498\, e_z$, whose values can be scaled to $\Gamma, \, z$ and $r$ values.

One problem encountered when using this solution is in step 3 where Pullin's solution is scaled for the appropriate test case. Since Pullin's wake and circulation distribution were not available in tabular form, several values were estimated from the plots in references 10 and 11 and then splined. Due to unavoidable errors in transcribing this data, however, small perturbations would amplify and the wake would rapidly go unstable. To resolve this problem, Pullin's solution is applied to a specific case study (in this instance the ellipse), and then the resulting wake and circulation distributions are fitted with equations. By using the data from these equations (which contains the Pullin solution), the singularity due to $d\Gamma/dr$ approaching infinity at the blade tip is eliminated and an initial wake geometry and circulation distribution are defined.

2.6 Vortex Sheet Model

The basic wake model consists of a continuous vortex sheet with a discrete tip vortex. The wake is defined by two independent parameters, $R$ and $Z$, where markers are placed. These markers are also the locations where the induced velocities are calculated, and are
therefore the points tracked downstream of the rotor blade which define the wake evolution in time. The number of markers used to represent the wake is a variable. Two other important parameters used throughout the analysis, \( r \) and \( z \), are variables of integration, and represent the location of the integration strips containing the vorticity in the wake. The remaining two sheet geometry parameters are \( S \) and \( s \), both referring to the sheet coordinate. The parameter \( S \) is the actual location of the markers along the sheet, while \( s \) is the location of the integration strips along the sheet. The number of integration strips is also a variable. These integration strips are modeled as circles from \( \Psi = -\pi \) to \( \Psi = \pi \) as shown in figure 3.1-1. Relating these six parameters:

\[
\Delta S^2 = \Delta R^2 + \Delta Z^2 \\
\Delta s^2 = \Delta r^2 + \Delta z^2
\]

3. Analytical Method

3.1 Application of the Biot-Savart Law

The two-dimensional near-wake geometry is obtained using a free-wake analysis technique. Using this approach, the wake assumes an equilibrium (or "force-free") position by moving according to the induced velocities at each marker location. These velocities are obtained by applying the Biot-Savart law at each marker and integrating along the sheet.

The wake is composed of circular integration intervals containing the circulation in the wake. These intervals are of infinitesimal strength and create a continuous vortex sheet. They are termed integration
intervals rather than vortex filaments, since they do not require a finite core size, as is the case with conventional filamentary models. In this continuous sheet model, the coordinate s is defined positive outward along the sheet. The location of the circular integration strips along the sheet are defined by lower case s, while the locations of the markers, where the induced velocities are calculated, are defined by S (see figure 3.1-1).
Figure 3.1-1 Problem Set-Up
To calculate the induced velocity at any point in the wake due to a filament of strength $\Gamma$, the Biot-Savart law can be applied as follows:\textsuperscript{12}

$$V_i = \int \frac{\Gamma}{4\pi} \left\{ \frac{\mathbf{dh} \times \mathbf{l}}{|l|^3} \right\} d\Psi$$  \hspace{1cm} (3.1)

$\Gamma$: strength of the vortex filament.

$\mathbf{dh}$: segment of the filament which induces a velocity at a point in space.

$l$: radius vector from $\mathbf{dh}$ to any point in space.

\textbf{Application of the Biot-Savart Law}

For a vortex sheet composed of circular integration intervals initially in the plane of the rotor, this equation becomes (see figure 3.1-2):

$$V_i = \int \int \frac{d\Gamma}{ds} \frac{1}{4\pi} \frac{\mathbf{dh} \times \mathbf{l}}{|l|^3} d\Psi \, ds$$  \hspace{1cm} (3.2)
Now referring to figures 3.1-1 and 3.1-2 and realizing there is an additional component of the $l$ vector in the $z$-plane at all times after $\tau=0$, it is clear that:

$$l = (R-r \cos \Psi)r - (r \sin \Psi)t + (Z-z)z$$  \hspace{1cm} (3.3)

Since the $dh$ vector in equation 3.2 is tangent to the circular integration strips (which remain parallel to the rotor plane at all times downstream), it is only composed of components in the $r$-$t$ plane. This vector can be represented as:

$$dh = (r \sin \Psi d\Psi)r - (r \cos \Psi d\Psi)t$$  \hspace{1cm} (3.4)
Therefore, after computing the cross-product $\mathbf{d}\mathbf{h} \times \mathbf{l}$ and substituting back into the Biot-Savart law (equation 3.2) the following equations are obtained:

$$V_a = \text{axial induced velocity} = \frac{1}{4\pi} \int_{0}^{s_{\text{max}}} \frac{d\Gamma}{ds} r \int_{-\pi}^{\pi} \frac{R \cos(\Psi) - r}{|l|^3} d\Psi ds \quad (3.5)$$

$$V_r = \text{radial induced velocity} = -\frac{1}{4\pi} \int_{0}^{s_{\text{max}}} \frac{d\Gamma}{ds} r \int_{-\pi}^{\pi} \frac{r \cos(\Psi) (Z-z)}{|l|^3} d\Psi ds \quad (3.6)$$

$$V_t = \text{tangential induced velocity} = -\frac{1}{4\pi} \int_{0}^{s_{\text{max}}} \frac{d\Gamma}{ds} r \int_{-\pi}^{\pi} \frac{r \sin(\Psi) (Z-z)}{|l|^3} d\Psi ds \quad (3.7)$$

These equations represent the radial, axial, and tangential induced velocities calculated at each marker location.

3.2 Axial Velocity Component

As shown above, the equation derived for the axial velocity using the Biot-Savart law is:

$$4\pi V_a = \int \frac{d\Gamma}{ds} r \int \frac{R \cos(\Psi) - r}{|l|^3} d\Psi ds \quad (3.8)$$
where:

\[ \beta = \left\{ R^2 - 2Rr \cos \Psi + r^2 + (Z-z)^2 \right\}^{3/2} \]  

(3.9)

Now defining \( F \) as:

\[ F = \frac{R \cos(\Psi) - r}{l/l^3} \]  

(3.10)

and substituting \( ll^3 \) into this expression, \( F \) becomes:

\[ F = \frac{R \cos \Psi - r}{[R^2 - 2Rr \cos \Psi + r^2 + (Z-z)^2]^{3/2}} \]  

(3.11)

It is apparent that a numerical singularity is encountered when \( \cos \Psi \) approaches 1, and \( r \) approaches \( R \) (when \( r=R, z=Z \)). To resolve this singularity, \( \cos \Psi = 1 - \frac{\Psi^2}{2} \) may be substituted into 3.11 to yield the factor which must be added and subtracted to \( F \) so the equation is well-behaved. Upon completing this substitution, it is apparent that the factor \( H \) (defined below) when used as shown in 3.13, eliminates the singularity in 3.11.

\[ H = \frac{R-r}{[(R-r)^2+Rr\Psi^2+(Z-z)^2]^{3/2}} \]  

(3.12)

\[ 4\pi V_a = \int_{d\Gamma} d\Sigma \int \left\{ [F(r,R,\Psi)-H(r,R,\Psi)] \right\} d\Psi d\Sigma + \int_{d\Gamma} d\Sigma \int H(r,R,\Psi) d\Psi d\Sigma \]  

(1) (2) (3)

(3.13)
In this manner, as $\cos \Psi$ approaches 1, and $r$ approaches $R$, the numerator and denominator of the first two terms in 3.13 approach zero at the same rate. The third term in equation 3.13 is now integrated analytically with respect to $\Psi$ from $-\pi$ to $\pi$ and defined as "A" where:

$$A = \int_{-\pi}^{\pi} H(r,R,\Psi) d\Psi = \frac{\frac{d\Gamma}{ds} r(R-r)2\pi}{[(R-r)^2 + (Z-z)^2] [(R-r)^2 + (Z-z)^2 + Rr\pi^2]^{1/2}}$$

(3.14)

This analytical solution (3.14) now must be integrated over $s$ numerically. Once again, however, the denominator goes to zero faster than the numerator as $r$ approaches $R$ (as an integration strip approaches a marker). The elimination of this singularity is slightly more complex because the integral is over $s$, and the relationships between $r$ and $s$, and $z$ and $s$ are known only in spline form. To solve this problem, the $r$ and $z$ terms are linearized with respect to $s$ as outlined in section 2.3. In this manner, $r$ and $z$ can be represented as simple linear functions of $s$, and the integration of equation 3.14 with respect to $s$ becomes straightforward. This linearization and the modified form of 3.14 that follows will only be used over the linear region. At all other locations along the vortex sheet, equation 3.14 will be integrated numerically in its original form.

The linear relations defined in equation 2.3.1 will now be substituted into 3.14 for integration over $s$, resulting in:

33
\[ A_{\text{lin}} = 2\pi \int \frac{\frac{d\Gamma}{ds} K_1[R-K_1(S-s)]}{C(S-s)[C(S-s)^2+R^2\pi^2-RK_1(S-s)\pi^2]^{1/2}} \, ds \]  \hspace{1cm} (3.15)

where:

\[
C = K_1^2 + K_2^2 = 1
\]

\[
K_1 = \frac{dr}{ds} \bigg|_{s=S}
\]

\[
K_2 = \frac{dz}{ds} \bigg|_{s=S}
\]

Before the integration over \( s \) can be completed, however, it is apparent that an additional term \( B \) (below) needs to be added and subtracted from \( A_{\text{lin}} \) to avoid the singularity which occurs as \( s \) approaches \( S \):

\[
B = \frac{2K_1 d\Gamma}{C(S-s)} \]  \hspace{1cm} (3.16)

Equation 3.15 can now be integrated over \( s \) from 0 to \( s_{\text{max}} \), and the third term in equation (3.13) results in:

\[
r \frac{d\Gamma}{ds} \int Ads = \int \frac{d\Gamma}{ds} \frac{r(R-r)2\pi}{[(R-r)^2+(Z-z)^2]^{1/2}[(R-r)^2+(Z-z)^2+Rr(\pi)^2]^{1/2}} \, ds \]  \hspace{1cm} (3.17)

outside the linear region and:

\[
3.15 + 3.16 - 3.16 =
\]
\[
2\pi \int \left\{ \frac{d\Gamma}{ds} K_1[R-K_1(S-s)]}{C(S-s)[C(S-s)^2 + R^2\pi^2 - RK_1(S-s)\pi^2]^{1/2}} \right\}
\]

\[
- \frac{2K_{1d}s}{C(S-s)} ds + \frac{2K_{1d}}{C} \log \left| \frac{S-s_{\delta 1}}{S-s_{\delta 2}} \right|
\]  

within the linear region.

In the last term of 3.18, \( s_{\delta 1} \) refers to the first integration strip encountered inside the linear region, and \( s_{\delta 2} \) refers to the last integration strip encountered inside the linear region.

When considering a specific marker and stepping through the integration strips (i.e. integrating along the sheet), 3.17 is used until \( s \) comes within + or - \( \delta \) (which defines the linear region- see figure 2.4-1) of the marker. At this point, 3.18 is used until this linear region is exited, and then use of 3.17 is resumed.

Now reviewing the first two terms in 3.13 (F-H) there is one more singularity to be eliminated. This is again resolved by substituting \( \cos \Psi = 1 - (1/2)\Psi^2 \) in the numerator and denominator of \( F \) (as shown in 3.19) to yield the additional term needed to eliminate the singular behavior of equation 3.19.

\[
F-H = \frac{R - \frac{R\Psi^2}{2} - r}{[R^2 + r^2 - 2Rr + Rr\Psi^2 + (Z-z)^2]^{3/2}} - \frac{R - r}{[R^2 + r^2 - 2Rr + Rr\Psi^2 + (Z-z)^2]^{3/2}}
\]  

(3.19)
The two terms in 3.19 will not cancel unless $R \Psi^2/2$ is included in the numerator of the second term. This factor must also be subtracted from 3.19 so the equation remains unchanged. Therefore:

\[
F - H = \frac{R \cos(\Psi) - r}{[R^2 - 2Rr \cos \Psi + r^2 + (Z-z)^2]^{3/2}}
\]

\[
\frac{R - r - \frac{R \Psi^2}{2}}{[R^2 + r^2 - 2Rr + Rr \Psi^2 + (Z-z)^2]^{3/2}}
\]

\[
\frac{R \Psi^2}{2} \frac{2}{[R^2 + r^2 - 2Rr + Rr \Psi^2 + (Z-z)^2]^{3/2}}
\]

(3.20)

The first two terms of equation 3.20 will now be renamed $F - H'$. The third term of equation 3.20 is integrated analytically over $\Psi$ and becomes:

\[
\frac{\pi}{r[(R-r)^2 + (Z-z)^2 + Rr(\pi)^2]^{1/2}}
\]

\[
\frac{1}{2r \sqrt{Rr}} \left\{ \log \left[ \pi + \sqrt{\pi^2 + \frac{(R-r)^2 + (Z-z)^2}{Rr}} \right] - \log \left[ -\pi + \sqrt{\pi^2 + \frac{(R-r)^2 + (Z-z)^2}{Rr}} \right] \right\}
\]

(3.21)

All terms in 3.21 will be well-behaved when integrated along the sheet except the third term which exhibits singular behavior as $s$ approaches $S$. This singularity is analytically resolved by using a
Taylor's Series expansion, which reveals the additional term to be added and subtracted from this function so it becomes well-behaved. The resulting third term in 3.21 is:

\[
\frac{1}{2r\sqrt{b}} \left\{ \log \left[ -\pi + \sqrt{\pi^2 + \frac{(S - s)^2}{R[R-k_1(S-s)]}} \right] + \log \left[ \frac{(S-s)^2}{2\pi R^2} \right] \right\} (3.22)
\]

Because of finite precision arithmetic, (3.22) exhibits noisy behavior when:

\[
\frac{K_1(S-s)}{R} < 0.01
\]

It is therefore necessary to do an expansion about 1 to the first log term in 3.22, which will only be used when \( \frac{K_1(S-s)}{R} < 0.01 \). This first term becomes:

\[
\frac{1}{2r\sqrt{b}} \log \left[ \frac{1}{R} \right] \cdot \left( \log \left[ \frac{1}{K_1(S-s)} \right] \right). (3.23)
\]

Defining the total component of axial velocity (including the additional terms so the equation remains well-behaved over the linear region):

\[
V_a = \frac{1}{4\pi} \int \frac{d\Gamma}{ds} r \left\{ \int (F-H')d\psi + \frac{\pi}{r\sqrt{a+b(\pi)^2}} \right\}.
\]
\[
\frac{1}{2r\sqrt{b}} \left[ \log \left( \pi + \sqrt{\pi^2 + \frac{a}{b}} \right) - A^* \right] \right\} \, ds + \frac{d\Gamma}{ds} \bigg|_{s=S} \frac{8}{\pi \sqrt{b}} \int \log \left( \frac{(S-s)^2}{2\pi R^2} \right) \, ds
\]

+ (3.18) \hspace{2cm} (3.24)

where:

\[
A^* = \log \left( -\pi + \sqrt{\pi^2 + \frac{(S-s)^2}{R[R-K_1(S-s)]}} \right)
\]

when \( \frac{K_1(S-s)}{R} < 0.01 \)

\[
A^* = \log \left( \frac{1}{K_1(S-s)} \right)
\]

when \( \frac{K_1(S-s)}{R} \geq 0.01 \)

Outside the linear region, however, this equation simplifies to:

\[
V_a = \frac{1}{4\pi} \int \frac{d\Gamma}{ds} r \left\{ \int (F-H')d\Psi + \frac{\pi}{r\sqrt{a+b\pi^2}} - \right. \]

\[
\left. \frac{1}{2r\sqrt{b}} \left[ \log \left( \pi + \sqrt{\pi^2 + \frac{a}{b}} \right) - \log \left( -\pi + \sqrt{\pi^2 + \frac{a}{b}} \right) \right] \right\} \, ds + 2\pi \int \frac{rG'(s)(R-r)}{a[a+b\pi^2]^{1/2}} \, ds
\]

where:

\[
a = (R-r)^2 + (Z-z)^2
\]

\[
b = Rr
\]
3.3 Radial Velocity Component

The radial velocity component of the induced velocity is derived in a similar manner. First, from the Biot-Savart law, the general form of the radial velocity is:

\[ V_r = - \int \frac{d\Gamma}{ds} \int \frac{r \cos \Psi (Z-z)}{l^3} \, d\Psi \, ds \]  

where \( J \) is defined as:

\[ J = \frac{\cos \Psi (Z-z)}{[R^2 - 2Rr \cos \Psi + r^2 + (Z-z)^2]^{3/2}} \]  

The singularities in \( J \) are analytically removed using equations 3.28 and 3.29.

\[ K = \frac{Z-z}{[(R-r)^2 + Rr\Psi^2 + (Z-z)^2]^{3/2}} \]  

\[ 4\pi V_r = \int \frac{d\Gamma}{ds} \left\{ \int [J(r,R, \Psi) - K(r,R, \Psi)] \, d\Psi + \int K(r,R, \Psi) \, d\Psi \right\} \, ds \]  

The third term in equation 3.29 is now integrated analytically over \( \Psi \) and is found to be:
This equation must now be integrated over \( s \), but singularities are encountered identical to those in equation 3.14 in the previous section. This singular behavior is handled in a similar manner, transforming equation 3.30 into:

\[
2\pi \int \left\{ \frac{d\Gamma}{ds} \frac{K_2 [R - K_1 (S-s)]}{C(S-s)[C(S-s)^2 + R^2 \pi^2 - K_1 (S-s) \pi^2]^{1/2}} - \frac{d\Gamma}{ds} \frac{2K_2}{C(S-s)} \right\} ds
\]

\[
+ \frac{d\Gamma}{ds} \frac{2K_2}{C} \log\left[\frac{S-s_{s1}}{S-s_{s2}}\right]
\]

\( C = K_1^2 + K_2^2 = 1 \)

within the linear region and

\[
2\pi \int \frac{d\Gamma}{ds} \frac{r^2 \pi (Z-z)}{[(R-r)^2 + (Z-z)^2] \left[\left[(R-r)^2 + (Z-z)^2 + Rr(\pi)^2\right]^{1/2}\right]} \]

outside the linear region.

Returning to the first two terms in equation 3.29, \( J \) and \( K \), there is one additional singularity which is removed by adding and subtracting 3.33 to \( J-K \) which results in equation 3.34.
\[
\frac{\psi^2}{(Z-z)^2} \left[ ((R-r)^2 + Rr\psi^2 + (Z-z)^2)^{3/2} \right]
\] (3.33)

\[
J-K = \frac{\cos(\Psi) (Z-z)}{[R^2 - 2Rr \cos\Psi + r^2 + (Z-z)^2]^{3/2}}
\]

\[
\frac{(Z-z)^2 - (Z-z)\psi^2}{2} \left[ (R^2 + r^2 - 2Rr + Rr\psi^2 + (Z-z)^2)^{3/2} \right]
\]

\[
\frac{(Z-z)^2}{2} \left[ (R^2 + r^2 - 2Rr + Rr\psi^2 + (Z-z)^2)^{3/2} \right]
\] (3.34)

The first two terms of 3.34 will now be renamed J-K'. The third term, denoted D1, is then integrated analytically to yield:

\[
D1 = \frac{\text{del} \pi}{b \left[ (R-r)^2 + \text{del}^2 + b(\pi)^2 \right]^{1/2}}
\]

\[
\frac{\text{del}}{2b\sqrt{b}} \left[ \log\left(\pi+\sqrt{\pi^2+\frac{a}{b}}\right) - \log\left(-\pi+\sqrt{\pi^2+\frac{a}{b}}\right) \right]
\] (3.35)

The third term in 3.35 exhibits singular behavior identical to the third term in equation 3.21, and is handled in a similar manner (which translates into the B* terms in 3.36). The only difference is in the numerically integrated term which is approximated as zero due to its symmetry about a marker location.
Defining the total component of radial velocity (including the additional terms to eliminate the singularity):

\[ V_r = \frac{1}{4\pi} \int \frac{d\Gamma}{ds} \left[ \int r (J-K')d\Psi + \frac{(Z-z)\pi}{R\sqrt{a+Rr(\pi)^2}} \right. \]

\[ - \frac{1}{2R\sqrt{b}} \left\{ (Z-z)\log(\pi + \sqrt{\pi^2 + \frac{a}{b}}) - B^* \right\} \]

\[ ds + (3.31) \]

(3.36)

where:

\[ B^* = K_2(S-s) \log \left[ \frac{-\pi + \sqrt{\pi^2 + \frac{(S-s)^2}{R[R-K_1(S-s)]}}}{\frac{(S-s)^2}{2\pi R^2}} \right] \text{ when } \frac{K_1(S-s)}{R} < 0.01 \]

\[ B^* = K_2(S-s) \log \left[ \frac{1}{K_1(S-s)} \right] \text{ when } \frac{K_1(S-s)}{R} \geq 0.01 \]

If, however, the program is not in the linear region, this equation simplifies to:

\[ V_r = \frac{1}{4\pi} \int \frac{d\Gamma}{ds} r \left[ \int (J-K\text{bar})d\Psi + \frac{(Z-z)\pi}{b\sqrt{a + b(\pi)^2}} \right. \]

\[ - \frac{1}{2R\sqrt{b}} \left\{ (Z-z)\log(\pi + \sqrt{\pi^2 + \frac{a}{b}}) - B^* \right\} \]

\[ ds + (3.31) \]

(3.36)
\[
\frac{(Z-z)}{2b \sqrt{b}} \left\{ \log(\pi + \sqrt{\pi^2 + \frac{a}{b}}) - \log(-\pi + \sqrt{\pi^2 + \frac{a}{b}}) \right\} ds + 2\pi \int \frac{d\Gamma}{ds} \frac{r(Z-z)}{a[a+b\pi^2]^{1/2}} ds
\]

where \( a = (R-r)^2 + (Z-z)^2 \) \( b = Rr \) \hspace{1cm} (3.37)

3.4 Tangential Velocity Component

The general form of the tangential induced velocity, developed using the Biot-Savart Law, is defined as:

\[
V_t = -\int \frac{d\Gamma}{ds} \int \frac{r \sin\Psi (Z-z)}{ll^3} d\Psi ds \hspace{1cm} (3.38)
\]

This term has negligible effect on the wake geometry for the following reasons. The denominator must be small (or the numerator very large) to obtain a significant component of \( V_t \). Since \( l \) (the denominator) is the distance between the interval of integration and the marker position, \( l \) is only small when \( \Psi \) is near \(-\pi\) or \( \pi \) (see figure 3.1-2). When \( \Psi \) is near \(-\pi\) or \( \pi \), however, \( \sin\Psi \) is approximately zero, so the tangential induced velocity is always small and has no significant effect on the wake geometry and is therefore ignored. This is one of the bases of the axisymmetry assumption made in this thesis.

3.5 Self-Induced Velocity of the Tip Vortex

Since the tip vortex has a finite strength and is curved, a self-induced velocity term is included. This self-induced velocity is found to be:
\[ w = \frac{\Gamma}{4\pi R} \left\{ \ln \frac{8R}{a} - \frac{1}{4} \right\} \]  

(3.39)

where \( R \) is equal to the radial location of the vortex and "a" is its core radius.\textsuperscript{13}

3.6 Smoothing

Under certain conditions, minor instabilities occur in the wake beyond 90% radius. If these instabilities are not eliminated, they are carried into the vortex (as the sheet is stretched) and the vortex roll-up becomes unstable. For this reason, a smoothing routine was introduced. This routine, called \textit{filter}, uses a second order difference scheme to smooth certain points in the wake.

Filter solves:

\[ -t_{\text{fil}}^2 \frac{d^2r}{ds^2} + r = r_0 \]

which smooths \( r_0(s) \) into \( r(s) \). This is done in an identical manner for \( z(s) \). It is therefore evident that if \( t_{\text{fil}} = 0 \), then \( r=r_0 \) and there is no smoothing, alternatively if \( t_{\text{fil}} \) approaches infinity, then \( \frac{d^2r}{ds^2}=0 \) (i.e. \( r(s) \) is linear in \( s \), and the wake sheet is "smoothed" straight).

The degree of smoothing is therefore controlled locally by \( t_{\text{fil}} \). \( T_{\text{fil}} \) is set to a specific value (usually between 0.01 and 0.02) if the marker being considered is in need of smoothing and 0 if no smoothing is necessary.

Although a general criterion was followed in defining \( t_{\text{fil}} \) as either zero or some finite value, modifications were necessary for each test case as defined below.
Circular Disk Distribution

For this fundamental test case, which looks at the downwash on a translating disk, no smoothing was necessary.

Elliptical Circulation Distribution

Since the elliptical distribution is monotonically decreasing, geometric inflection points in the wake are used to enact the smoothing routine. The markers in need of smoothing are located by considering subsequent sets of four points (forming 3 vectors), and multiplying the cross product of the first and second vectors times the cross product of the second and third vectors. If this product is found to be negative, an inflection point has been located and \( t_{fil} \) is set to 0.01 for the marker being considered, otherwise \( t_{fil} \) is set to 0.

Miller Distribution

For this distribution there is a natural inflection point in the wake at approximately 95% radius.\(^{14}\) To avoid erroneous smoothing in this area, the smoothing routine used for the elliptical distribution was only applied from 0.96-1.0 R. In addition, a stronger smoothing parameter (0.015) was used.

4. An Elliptically Loaded Blade

For an elliptically loaded blade at \( \tau=0 \):

\[
\Gamma(r) = .02 \sqrt{1-(r)^2}
\]

\( r \rightarrow [0,1]; \, z=0 \)
Due to the singularity at the tip in the initial circulation distribution, a tip vortex is instantaneously formed. Pullin's self-similar solution is applied here, as discussed in Section 2.5. By following the steps outlined in that section, the scaling parameters $a$ and $t$ are found to be 0.01396 and 0.1373. These parameters are then used to scale the outer portion of the circulation distribution and wake geometry appropriately, while also giving the strength, location and radius of the tip vortex. The resulting circulation distribution and initial wake geometry are shown in Figure 2.5-1.

For this classical case, several issues are investigated as the blade passes through 45 degrees and the wake convects downstream.

(1) effect of smoothing on wake geometry
(2) effect of time-step on wake geometry
(3) effect of linear region size on wake geometry
(4) evolution of the wake from $\Psi = 0$ to $\Psi = 45$ degrees

4.1 Effect of Smoothing

As previously stated, the smoothing parameter ($t_{fil}$) used for this distribution was 0.01. In Figure 4.1-1, the wake at $\Psi = 30$ degrees is shown before and after smoothing.
Figure 4.1-1 Effect of Smoothing Wake Geometry
It is important to note that while all inflections are eliminated, the curvature of the wake and the extent of vortex roll-up are not changed significantly. Originally, a smoothing routine was adapted which smoothed every marker along the sheet, rather than only those near an inflection point. The result was a significant loss of curvature in the wake beyond 95% radius along with a large decrease in the diameter of the vortex.

4.2 Dependence of Wake Solution on Time Step

A Predictor-Corrector difference method was utilized for the time-step scheme, providing second-order accuracy. The wake was normally set to convect in one degree increments downstream of the blade. To validate convergence, however, half degree time steps were also tested on the elliptical distribution. The resulting wake geometries at various time steps downstream were compared with wake geometries using one degree increments. Figure 4.2-1 shows one of these wake comparisons at $\Psi = 5$ degrees. The slight difference seen between the two roll-up's indicates the coarseness of the time step is borderline.
Figure 4.2-1 Effect of Time Step on Wake Geometry
4.3 Dependence of Solution on Linear Region Size

The linear region, as explained in Section 2.3, is a region where the equations for the induced velocity are modified to eliminate singularities that arise as a variable of integration approaches a marker location. The difference between the modified and unmodified equations involves only higher order terms. These terms should have negligible effect on the resulting induced velocities, and therefore the solution should be independent of the size of the linear region (as long as the linear relations in section 2.3 are preserved). Test cases were run for several linear region sizes, and the resulting wake geometries confirmed that the solution is in fact independent of linear region size. Figure 4.3-1 shows a comparison between a constant linear region size of $\delta = 0.003$ versus a variable linear region size (depending on marker location) that was normally used.
Figure 4.3-1 Effect of Linear Region Size on Wake Solution
4.4 Calculated Wake Geometry

In Figures 4.4-1, 4.4-2, and 4.4-3 the wake is shown convecting downstream as the blade moves from 0 to 45 degrees. As expected, the tip vortex convects downstream slower than the rest of the vortex sheet. There is a significant roll-up in the near-wake, but the vortex remains well-behaved (assisted by the smoothing routine). Only the tip is shown in the following figures since the wake remains relatively flat inboard.
Figure 4.4-1 Wake Profiles for the Elliptically Loaded Blade
WAKE GEOMETRY

-0.0300

0.0033

0.0200

0.9500 0.9667 0.9833 1.0000 RADIUS

0.0033

0.0200

0.9500 0.9667 0.9833 1.0000 RADIUS

ELLiptical DISTRIBUTiON AT PSI = 30 DEGREES

18 Jul 91 14:15:26

ELLiptical DISTRIBUTiON AT PSI = 40 DEGREES

18 Jul 91 14:21:27

Figure 4.4-2 Wake Profiles for the Elliptically Loaded Blade
Figure 4.4-3 Wake Profile for the Elliptically Loaded Blade at $\Psi = 45^\circ$.
If the sheet is allowed to convect further downstream, the extent of roll-up may lead to instabilities which could be handled using a dumping technique outlined by Hoeijmakers. Using this approach, the vortex sheet is only allowed four revolutions, and any vorticity past that point is fed into the vortex core. The core is then re-located to preserve the center of vorticity. The amount of roll-up is checked at each iteration and the process is repeated.

5. Translating Circular Disk

As a check case, a circulation distribution was developed for an oblate ellipsoid (with minor axis $b$ and major axis $a$), moving parallel to its axis of rotation $b$, in the limit as its minor axis goes to zero. For this case, an irrotational flow due to a circular disk with radius "a" results as shown in Figure 5-1. Since the boundary condition states that no flow can pass through this disk, all particles on the disk must descend at the same rate as the disk, producing a constant downwash condition. The intent of this test case is to validate the basic analytic theory developed in this thesis and its application in the present code.

![Figure 5-1 Oblate Spheroid Collapsed to Circular Disc](image)

In Batchelor, the velocity potential for this flow is given as:

$$\Phi = \frac{U}{2} (x^2 + y^2)$$
\[ \Phi = -\frac{2aU}{\pi} \cos \eta \]

where \( \zeta \) and \( \eta \) are the elliptical coordinates obtained through the transformation:

\[ x + i\sigma = (a^2 - b^2)^{1/2}\sinh(\zeta + i\eta) \]

For the circular disk, using the coordinate system shown in figure 5-1, it is apparent \( x \) and \( b \) equal zero while \( a \) equals unity. This leaves the transformation \( \sigma = \sin(\eta) \), where \( \phi \) can be written as:

\[ \phi = -\frac{2aU}{\pi} (1 - \sigma^2)^{1/2} \]

The circulation distribution along the plate is simply the change in velocity potential across the plate giving:

\[ \Gamma(s) = \frac{-4aU}{\pi} (1 - r^2)^{1/2} \]

Using this distribution (see figure 5-2), a flat wake exhibiting constant downwash should result. Figure 5-3 shows the resulting downwash distribution for this test case. The circulation distribution was derived assuming a non-dimensional downwash of 0.02 and the code accurately reproduces this value.
Figure 5-2 Circulation and Wake Distribution for the Oblate Spheroid
Figure 5-3 Downwash Distribution for the Oblate Spheroid
Two abnormalities are seen in this plot, one at 80% radius and one at the tip. The slight increase in accuracy seen at .8 radius is due to a higher density of integration strips placed outboard of this location. This is necessary to correctly account for the vorticity in this area where the extent of curvature in the circulation distribution is significantly increased. At the tip spikes are seen on the outer 0.02%. These are due to the inability of the model handle \( \frac{d\Gamma}{ds} \) approaching infinity at the tip.

6. Typical Hovering Rotor Loading

The circulation distribution shown in figure 6-1 was chosen as the final test case because there has been considerable debate over the resulting wake geometry. This circulation distribution was obtained iteratively by Miller\(^{17} \) using the entire wake, but has been used in discrete near-wake analyses to try to understand the resulting wake behavior.\(^{18} \) The main point in question is whether or not a mid-span vortex exists. Full-wake discrete models\(^{19,20} \) have shown the formation of a mid-span vortex, however this contradicts experimental results. Since the continuous vortex sheet representation of the present approach is physically more correct, it was hoped that the character of the wake roll-up would more closely match experimental evidence. Figures 6-2 through 6-5 show the roll-up of the wake as the blade advances through 45 degrees.
Figure 6-1 Typical Hovering Rotor Circulation & Wake Distribution
Figure 6-2 Wake Profile for Typical Hover Circulation, $\Psi = 10^\circ$
Figure 6-3 Wake Profile for Typical Hover Circulation, $\Psi = 20^\circ$
Figure 6-4 Wake Profile for Typical Hover Circulation, $\Psi = 30^\circ$
Figure 6-5 Wake Profile for Typical Hover Circulation, $\Psi = 45^\circ$
It is seen in figure 6-3, that at $\Psi=20$ degrees an additional vortex starts to appear. It is induced at approximately 92% blade radius which corresponds with the peak in circulation shown in figure 6-1 (remember only smooth past 96% radius). This roll-up is opposite to that seen at the tip as expected, and becomes more pronounced as the blade convects downstream.

By comparing the elliptical and typical hover circulation distributions, it appears the the outer portion of the wake for the typical hover distribution should convect faster than the elliptical case, and the inner portion slower, which is confirmed by comparing the resulting wake geometries.

8. Conclusions and Future Recommendations

The program resulting from this research can be utilized to assist in resolving the arbitrariness of the finite core size in discrete models, and as the near-wake component of a hovering or forward flight full-wake code.

First, understanding more completely the physical behavior of the near-wake may give insight to why current computer models are not currently matching experimental results.

There seems to be a consensus that representing the wake as a vortex sheet rather than a series of discrete filaments leads to a more realistic model, but at the expense of theoretical and computational
simplicity. Since one method's advantage is the other's weak point, it seems a study comparing the two models would be helpful in determining a fairly simplistic but more physically correct model. It was hoped that a comparison between discrete versus continuous models could be done in this thesis, but was impossible due to the unavailability of a hovering, discrete, near-wake code. By creating a code similar to the one contained herein (but using a discrete model), a solid analysis could be done. By comparing the results between these two models, an appropriate core size may become apparent for the discrete model, making its solution much less arbitrary while keeping its simplicity in place.

It does not appear however, that doing this alone will result in the agreement of experimental and computational results, since the more complex continuous model still revealed a mid-span vortex which has yet to be documented experimentally. The reason for this occurrence is still unclear and will possibly only be resolved when viscous terms are included in the wake analysis.

The other primary use for this code, implementation into hovering and forward flight full-wake codes, can be accomplished with minimal changes.

For hovering cases, as outlined in reference 13, the integration strips are defined as semi-circles from $\Psi = 0$ to $\Psi = \frac{\pi}{2}$. By reviewing equations 3.5, 3.6, and 3.7, it is apparent this modification can be accomplished by simply dividing these equations by 2.
For forward flight, however, not only the limits of integration over $\Psi$ changes, but the actual wake geometry changes as well. The upper limit of $\Psi$ in this case defines the near wake, with the intermediate and far wakes following. Since the wake geometry is different for forward flight due to the additional forward flight velocity term, the location of the integration intervals will have to be modified. This can be accomplished by appropriately modifying the $l$ vector in equation 3.3 and carrying these modified terms throughout the equations. It should be emphasized that except for the $l$ vector, the overall numerical Biot-Savart analysis developed in this thesis remains unchanged.


7 Miller, R. H., *Methods for Rotor Aerodynamic and Dynamic Analysis*, *Progress in Aerospace Sciences*, 22(2), 1985, p. 120.


Appendix - Computer Programs
I have two things defined as "a" in this program but they do not interact.

This program is to calculate the two components of the induced velocity, Vz and Vr. This will be done using the Biot Savart Law. The induced velocity will be calculated at various positions downstream of the blade along the vortex sheet. The resulting wake will then be plotted. The sheet is not flat, but from a side view is curved and rolls up toward the tip where a tip vortex is located.

```
dimension delrsq(301), delsq(301), vrold(301),
& ls(2000), bs(301), lr(2000),
& dgambs(2000), x(301), gam(301),
& bz(301), bzold(301), dels(301),
& dgamls(2000), vindz(301), br(301),
& vindz(301), brold(301), gb(2000),
& gbs(301), rbs(301), zbs(301),
& zbr(301), s(2000), expls(2000),
& vzold(301)

dimension nper(2), ilin(2), isym(2)

real ls, lr, lz, kdphi, lsdan, lsdana
real integ2, integ1, k1, k2

data ilin/1,0/
data isym/1,2/

character * 30 titl(2)
character * 30 pltitl
```
\[ \pi = 4 \cdot \arctan(1) \]

\[
\frac{\pi}{4}/50 \cdot 0.0161 \text{ deg} \ldots t* = t(\omega) \\
\text{when } t*=1, \ t=\text{time to go one radian, therefore vary } t*=0. \text{ to } \pi/4
\]

\[ \delta = 0.01745 \]

\[ \text{TIME}=0. \]

```
WRITE(*,76)
76 FORMAT(1X,'WHAT IS NSTOP? - 99 if prog stopped')
READ(5,77) NSTOP
77 FORMAT(12)

WRITE(*,885)
885 FORMAT(1X,'WHAT CASE ARE YOU RUNNING? 1-DISK, 2-ELL, 3-TYP HOV')
READ(*,886) NCASE
886 FORMAT(12)
write(*,886) ncase
```

tfil is the parameter which controls the strength of the local smoothing.

```
WRITE(*,80)
80 FORMAT(1X,'WHAT IS TFIL?')
READ(*,81) SMINC
81 FORMAT(F10.4)
```

cThe following is for fildis.f

```
WRITE(*,90)
90 FORMAT(1X,'Enter amark: ellip-.9, disk-.8, Typ Hover-.9')
READ(5,* ) amark
WRITE(*,91) amark
91 FORMAT(1X,'amark equals',1X,F10.4)
WRITE(*,922)
922 FORMAT(1X,'Enter vardint:/',/,'ellip-.0002, disk-.0002, typical hover-.002')
READ(5,* ) vardint
WRITE(*,93) vardint
93 FORMAT(1X,'vardint equals',1X,F10.4)
WRITE(*,94)
94 FORMAT(1X,'****amark and vardint are found in fildis.f')
```

C******THIS IS MAIN SECTION TO MAKE PROGRAM ABLE TO STOP AND START
C SEE END OF CODE TO SEE WHAT IS WRITTEN TO THESE FILES.

```
IF(NSTOP.EQ.99) THEN
   WRITE(*,854)
854 FORMAT(1X,'RESTARTING VORTEX')
   WRITE(*,854)
```
open(unit=63, file='wakestore.dat', status='old')
open(unit=64, file='gamma.dat', status='old')
open(unit=65, file='miscstore.dat', status='old')

READ(65,*) NUMBR, TIME, II, RCORE, ZCORE
II = II+1

nbr = numbr
nvortex = numbr+1
if (ncase .eq. 1) then
    nmark = nbr
else
    nmark = nvortex
endif

do 78 i=1, nmark
    READ(63,*) BR(I), BZ(I)
    if (i .ne. nvortex) then
        READ(64,*) GAM(I)
    endif

78 continue

close(unit=63)
close(unit=64)
close(unit=65)

ENDIF

IF (NSTOP .EQ. 99) GOTO 150

c ***** END MOST ADDITIONS TO MAKE PROGRAM ABLE TO STOP AND START

  c for translating disk case
  if (ncase .eq. 1) then
    call disk(BR, BZ, BS, BS MAX, GB, TBS, NUM4, NUMBR)
a = 1
t = 1
  endif

  c for elliptical distribution case
  if (ncase .eq. 2) then
    write(*, *) ncase
    call ellip(BR, BZ, BS, BS MAX, GB, TBS, NUM4, NUMBR, a, t)
  endif

  c for typical hovering rotor case
  if (ncase .eq. 3) then
    call typhov(BR, BZ, BS, BS MAX, GB, TBS, NUM4, NUMBR, A, T)
  endif

  write(*, 101) bs max

101 format(1x, 'bs max = ', F10.4)
c nбр is number of markers along sheet
nбр=numбр

c nvortx is number of markers along sheet plus tip vortex
nvortx=numбр+1

c splining gamma versus sheet location
CALL SPLIND(GB,GBBTBS,TBS,NUM4,999.,999.)

open(unit=64, file='gamma.dat',status='unknown')

c loop to define gamma at every marker location, the gamma for a
c specific marker will NOT change since the markers are moving
c along a streamline.
do 92 j = 1,nбр
   gam(j) = seval(bs(j),gb,gbbtbs,tbs,num4)
   write(64,*) gam(j)
92 continue

close(unit=64)

***************plotting**********************
c call plotting routine for gamma vs marker position
nper(1) = nбр
titl(1) = 'gam vs. bs'
pltitl = 'bs "gam "'
call grinit(5,6,'gam vs bs')
call grklin(ilin,isym,nper,titl,1,bs,gam,pltitl,599)

***************end plotting*******************

c now for location of vortex core.
c the location for the core for typical hover and elliptical
c distribution is nearly identical, so approximated as
c that here.

RCORE = .9952
ZCORE = -.007684

***************plotting**********************
c call plotting routine for initial wake geometry
nvortx = numбр + 1
br(nvortx) = .9952
bz(nvortx) = -.007684
nper(1) = nбр
nper(2) = 1
titl(1) = 'init wake'
titl(2) = 'vortex pos'
pltitl = 'br "bz "'
call grinit(5,6,'bz vs br ')
call grklin(ilin,isym,nper,titl,2,br,bz,pltitl,599)

***************end plotting*******************
150 continue

nvortex = numbr + 1
NBR = NUMBR

c VORTEX CORE STRENGTH

GAMCOR = -1.342*(a**(4./3.))*(t**(1./3.))

c VORTEX CORE RADIUS

CORAD = .058*((a*t)**(2./3.))

IF(NSTOP.EQ.99) goto 54

c there will be 2 outer main loops in this program where
c ii => loop 15 outer time loop
c jj => loop 2007 outer loop for predictor corrector
c now loops for axial velocity
c j => loop 12 which goes through marker stations (for vz component)
c k => loop 13 which goes through integration strips (for vz component)
c iii=> loop 14 which goes through azimuthal stations (for vz component)
c now loops for radial velocity
c j => loop 121 which goes through marker stations (for vr component)
c k => loop 131 which goes through integration strips (for vr component)
c iii=> loop 141 which goes through azimuthal stations (for vr component)

do 15 ii=1,45

54 CONTINUE

DO 2007 JJ = 1,2

write(*,*) ii

c ALREADY FOUND BS'S IN PULLIN SUBROUTINE WHEN II = 1 SO SKIP TO C 25 CONTINUE
IF (II.EQ.1.and.jj.eq.1) GO TO 25

BS(1) = BR(1)

c THIS NBR NOT NVORTEX CAUSE VORTEX DOESNT HAVE BS, NOT PART OF SHEET
c finding bs locations of markers by using pythag. theorem on br and bz
do 10 i = 2,NBR

delrsq(i) = (br(i) - br(i-1))**2.
\[ delzsq(i) = (bz(i) - bz(i-1))^2. \]
\[ dels(i) = \sqrt{delrsq(i) + delzsq(i)} \]
\[ bs(i) = dels(i) + bs(i-1) \]

10 continue
\[ BSMAX = BS(NBR) \]

25 continue

c smoothing routine which locally smoothes points
\[ \text{call smooth(br,bz,bs,nbr,bsmax,ii,SMINC)} \]
\[ \text{write(*,1053) bsmax} \]
\[ 1053 \text{ format(lx,'bsmax after smooth=',f10.4)} \]

c can spline gam and bs cause gam at br = gam at bs originally
\[ \text{c and gamma stays same at any marker location since} \]
\[ \text{c moving along streamline. (marker locations are tracked by bs.)} \]

\[ \text{call splind(gam,gbs,bs,nbr,999.,999.)} \]
\[ \text{call splind(br,rbs,bs,nbr,999.,999.)} \]
\[ \text{call splind(bz,zbs,bs,nbr,999.,999.)} \]
\[ \text{call splind(bz,zbr,br,nbr,999.,999.)} \]

if(jj.eq.1) then
    time = delta
endif

nvortx = nbr + 1

self = 0.0

c if running translating disk case no vortex core
\[ \text{if(ncase.eq.1) then} \]
\[ \quad nmark = nbr \]
\[ \text{else} \]
\[ \quad nmark = nvortx \]
\[ \text{endif} \]

do 12 j = 1,nmark
    if(j.eq.nvortx) then
        br(j) = rcore
        bz(j) = zcore
dgambs(j) = gamcor
        go to 160
    endif
endif

c if not at vortex core get the deriv of gamma wrt bs through
\[ \text{c spline of gamma vs bs.} \]


dgambs(j) = deval(bs(j),gam,gbs,bs,nbr)

Call-out of filaments and definition of danger zone in fildis

CALL FILDIs(bs,bsmax,numbr,expls,xaxis,yaxis,npoin,klm,j,n6r,
&
    bslow,bshigh,dan,amark,vardint)

160 continue

singf3 = 0.
f3ext = 0.0
fhfunc = 0.
HP = 0.
klm = 0
lkm = 0
MARK = 0
NPT = 0
nflag = 0
analt1 = 0.

c n6r due to expls(K+1)
c n6r is number of filaments along sheet
c nvor is number of filaments along sheet plus vortex core filament.

nvor = n6r+1

c For translating disk case do not have vortex core (which is a
c filament and a marker for other cases) therefore no last filament.

    if(ncase.eq.1) then
        nfilmt = n6r
    else
        nfilmt = nvor
    endif

DO 13 K = 1,nfilmt

SING = 0.0
ALT = 0.0
f3dan = 0.0
f3else = 0.0

c when bs = vortex core and ls = vortex core
    if(k.eq.nvor.and.j.eq.nvortx) go to 185

c when ls = vortex core need to give location and strength of filament
    if(k.eq.nvor) then
        lr(k) = rcore
        lz(k) = zcore
        hls = 1.0
        dgiels(k) = gamcor
    else

78
C TAKE MIDPOINT TO OBTAIN LS(K) (WHERE FILAMENTS ARE
C LOCATED), AND USE MIDPOINT METHOD FOR INTEGRATION ALONG SHEET.

\[ LS(K) = \frac{EXPLS(K) + EXPLS(K+1)}{2}. \]
\[ HLS = EXPLS(K+1) - EXPLS(K) \]
\[ dgamls(k) = deval(ls(k),gam,gbs,bs,nbr) \]
\[ lr(k) = seval(ls(k),br,rbs,bs,nbr) \]
\[ lz(k) = seval(ls(k),bz,zbs,bs,nbr) \]

endif

nn = 72
phil = -pi
phiu = pi

fh = 0.0
hphi = (phiu - phil)/float(nn)

c loop for integration about azimuthal angle.

do 14 iii = 1,nn

c x = azimuthal angle (psi)

\[ x(iii) = phil + (float(iii) - .5)*hphi \]

onetop = br(j)*cos(x(iii)) - lr(k)
pt1 = br(j)**2 + lr(k)**2
pt2 = -2.0*br(j)*lr(k)*cos(x(iii)) + (bz(j) - lz(k))**2.
onebot = (pt1 + pt2)**.5

twotop = br(j) - lr(k) - (.5*br(j)*(x(iii)**2.))
pt1a = (br(j) - lr(k))**2. + br(j)*lr(k)*(x(iii)**2)
pt2a = (bz(j) - lz(k))**2

twobot = (pt1a + pt2a)**.5

c this is the integral f-h where h has a bar over it.
fhbar = onetop/onebot - twotop/twobot
fh = fh + fhbar

14 continue

fminh = hphi * fh

c now add in the analytical part that comes from hbar
\[ a = (br(j) - lr(k))**2. + (bz(j) - lz(k))**2 \]
\[ f1 = (pi/lr(k))*(1./((a+br(j)*lr(k)*(-pi)**2)**.5)) \]
\[ f2a = 1./((2*lr(k)*(br(j)*lr(k))**.5) \]
\[ f2b = a*log(pi + ((pi)**2 + a/(br(j)*lr(k))))*.5) \]
\[ f2 = f2a*f2b \]
\[ \text{func} = f_1 - f_2 \]

\[ \text{FINFHF} = (LR(K) \times DGAMLS(K) \times (FMNH + \text{FUNC})) \]

c now add on eqtns. 3.18 and eqtns. 3.22 and 3.23 when in linear zone
c and their complimentary parts when not in linear zone. These
c equations were integrated analytically over \( \phi \) but not \( s \).

c \( bs \) and \( ls \) not defined when \( j = \text{nvortx} \), and should never enter
c danger area when \( j = \text{nvortx} \) since vortex core is not along sheet.
if \( k = \text{nvor.} \text{or.} j = \text{nvortx} \) goto 167

\[ bslow = bs(j) - dan \]

if \( j = \text{numbr} \) then
  \[ bshigh = bs(j) \]
else
  \[ bshigh = bs(j) + dan \]
endif

if \( ls(k).gt.bslow \text{and} ls(k).lt.bshigh \) then

\[ \text{****these lines are for plotting purposes and for two analytical} \]
\[ \text{****integrals over} s. \]

\[ klm = klm + 1 \]

\[ \text{IF (KLM.EQ.1) THEN} \]
  \[ \text{NPT = KLM + LKM} \]
ELSE
  \[ \text{NPT = NPT + 1} \]
ENDIF

\[ \text{MARK = 1} \]

\[ \text{c if nflag is never set to one, then you are never inside danger zone} \]
\[ \text{c and you skip over analytical integrals over} s. \]

\[ nflag = 1 \]

\[ \text{lsdan(klm) = ls(k)} \]

\[ \text{c ***********end lines pertaining to above ** comment*************} \]

\[ \text{c br(nvortx) and bz(nvortx) will not go into deval cause nbr specified.} \]

\[ k1 = \text{deval(bs(j),br,rbs,bs,nbr)} \]
\[ k2 = \text{deval(bs(j),bz,zbs,bs,nbr)} \]

80
\[ a_1 = (\pi^2) \cdot (br(j)^2) \]
\[ b_1 = -(\pi^2) \cdot br(j) \cdot k_1 \]
\[ c_1 = k_1^2 + k_2^2 \]

\[ c \] should always be one (sometimes came out to be \(0.999999\), so just \(c \) made \(1.0\)).

\[ \text{if}(c_1.\!\!\!\!\!\!\ne.1) \text{ then} \]
\[ c_1 = 1 \]
\[ \text{endif} \]

\[ bs_\text{ls} = bs(j) - l_\text{s}(k) \]

***the additional terms that were integrated analytically wrt \(s\), for equation 3.18, 3.22 and 3.23, are found after 13 continue. They are analti for 3.18 and f3ext for 3.22 and 3.23.

**c equation 3.18**

\[ \text{chkttop} = k_1 \cdot 2 \cdot \pi \cdot dgamls(k) \cdot (br(j) - k_1 \cdot bs_\text{ls}) \]
\[ \text{chkbtop} = c_1 \cdot bs_\text{ls} \cdot \text{sqrt}(a_1 + b_1 \cdot bs_\text{ls} + c_1 \cdot (bs_\text{ls})^2) \]
\[ \text{chkttopa} = \frac{-2 \cdot k_1 \cdot dgambs(j)}{(c_1 \cdot bs_\text{ls})} \]
\[ \text{ALT} = (\text{chkttop} \cdot bs_\text{ls} - 2 \cdot k_1 \cdot dgambs(j) \cdot \text{chkbtop})/(bs_\text{ls} \cdot \text{chkbtop}) \]

**c for equation 3.22**

\[ \text{DANCHK} = (K_1 \cdot BSLS)/BR(J) \]

\[ \text{IF} (\text{DANCHK.GT.01}) \text{ THEN} \]
\[ f_3\text{dan} = \frac{(da_1mls(k))/(2*(br(j)*l_\text{r}(k))^2))}{\text{alog(abs((-pi+ sqrt((-pi)**2+(bs_\text{ls}**2/(br(j)**2-br(j)*k_1*bs_\text{ls}))))/((bs_\text{ls}**2)/(2.*(br(j)**2)*pi))))}} \]
\[ \text{ELSE} \]

**c equation 3.23**

\[ f_3\text{dan} = \frac{(da_1mls(k))/(2*(br(j)*l_\text{r}(k))^2))}{\text{alog(abs(1/(1-(K_1*BSLS/BR(J))))}}} \]

**ENDIF**

**c now if not in linear zone do "else" for ALT and f3dan.**

\[ \text{else} \]
continue

c *** lines for plotting
lkm = lkm + 1

IF (MARK.NE.1) THEN
  NPT = LKM
ELSE
  NPT = NPT + 1
ENDIF

c *** end lines for plotting

c equation 3.17

topa = lr(k)*dgamls(k)*2.*pi*(br(j) - lr(k))
a = (br(j) - lr(k))**2 + (bz(j) - lz(k))**2
bota = a*(a + br(j)*lr(k)*((pi)**2.))**.5
SING = TOPA/BOTA

c third term in 3.21

f3el1 = (1./(2*lr(k)*(br(j)*lr(k))**.5))
f3el2 = alog(abs((-pi + ((-pi)**2 + a/(br(j)*lr(k))))**.5))
f3else = (lr(k)*dgamls(k))*f3el1*f3el2

endif

FHFUNC = FHFUNC + HLS*FINFHF

singf3 = singf3 + hls*(f3dan + f3else)

HP = HP + HLS*(SING+ALT)

c 13 is loop over integration strips
13 continue

npoin = lkm + klm

c if never entered linear zone, then do not do analytical integrals
that go with danger zone.

if(nflag.eq.0) go to 878

c analytical integral coming from linear zone for equation 3.18
  ex = bs(j) - lsdan(klm)
ex1 = bs(j) - lsdan(1)
integl = 2*k1*dgams(j)*alog(abs(ex1/ex))
analti = integl

C ANALYTICAL INTEGRAL COMING FROM LINEAR ZONE FOR 3.22 AND 3.23

F3EXT1 = -2*((BS(J) - LSDAN(KLM))*ALOG(ABS(BS(J)-LSDAN(KLM)))
&+ LSDAN(KLM)*((ALOG((SQRT(2.*PI))*BR(J))) + 1))
F3EXT2 = 2*((BS(J)-LSDAN(1))*ALOG(ABS(BS(J)-LSDAN(1)))
&+ LSDAN(1)*((ALOG((SQRT(2.*PI))*BR(J))) + 1))

F3EXT = ((dgams(j)/(2*(brj)*brj)**.5))*(F3EXT1 + F3EXT2)

878 continue
185 continue

c for the induced velocity in z direction

if (j.eq.nvortx) then
C include self induced velocity of vortex core on itself.

C GAMCOR NEEDS A NEGATIVE SO IT CONVECTS DOWN BY MAKING "SELF"
C POSITIVE. GAMCOR BY ITSELF IS A NEGATIVE NUMBER.

c self is velocity core induces on itself.
  self = (-gamcor/(4.*pi*br(j)))*(alog(abs((8*br(j))/
& (corad) - (1./4.)))
  vindz(j) = ((hp + fhfunc + analti)/(4*pi)) + self
else
  vindz(j) = (hp + fhfunc + analti + singf3 + f3ext)/(4*pi)
endif

c 12 is loop which goes through markers
12 continue

c ******************NOW FOR RADIAL VELOCITY COMPONENT***************
c start loops for integration for radial component
	nvortx = nbr + 1
c for translating disk no vortex core, therefore one less marker.
if(ncase.eq.1) then
    nmark = nbr
else
    nmark = nvortx
endif

c loop through markers
    do 121 j = 1,nmark

c define location and strength of core
if(j.eq.nvortx) then
    br(j) = rcore
    bz(j) = zcore
    dgambs(j) = gamcor
    go to 175
endif

dgambs(j) = deval(bs(j),gan,gbs,bs,nbr)

175    continue

TPSEX = 0.0
SNGTP5 = 0.0
FKTOT = 0.0
SUMKDP = 0.0
klma = 0
lkma = 0
mark1 = 0
npt = 0
analalt2 = 0.
mflag = 0
e = 0.

c add extra filament to include vortex core
   c vortex core is NOT part of sheet
nvor = n6r + 1

c no vortex core for case 1, translating disk
if(ncase.eq.1) then
    nfilmt = n6r
else
    nfilmt = nvor
endif
c now integrate along sheet by using Midpoint Method, visiting each
c integration interval.
   do 131 k =1,nfilmt
      
      ALTA = 0.0
      TP5DAN = 0.0
      TP5ELS = 0.0
      
      KDPHI = 0.0
      
      c when j = nvortx, and k = nvor
      if (j.eq.nvortx.and.k.eq.nvor) goto 186
      
      c when k = nvor (at vortex core) define location and strength
      if (k.eq.nvor) then
         lr(k) = rcore
         lz(k) = zcore
         dgamls(k) = gamcor
         hls = 1.0
      else
         
      c must redefine ls for radial velocity because ls distribution is
c dependent on each marker location, and therefore gets overwritten
c each time.
         ls(k) = (expls(k) + expls(k+1))/2.
         hls = expls(k+1)-expls(k)
         dgamls(k) = deval(ls(k),gam,gbs,bs,nbr)
         lr(k) = seval(ls(k),br,rbs,bs,nbr)
         lz(k) = seval(ls(k),bz,zbs,bs,nbr)
      endif
      
      delz = bz(j)-lz(k)
      PHIL = -PI
      PHIU = PI
      NN = 72
      fK = 0.0
      
      c hphi is distance between integration strips
      hphi = (phiu - phil)/float(nn)
      
      do 141 iii = 1,nn
         
         x(iii) = phil + (float(iii) - .5)*hphi
      
      c now do the integration over j-kbar

      top1 = cos(x(iii))*delz
      bot1 = (br(j)**2 + lr(k)**2-2.*br(j)*lr(k)*
      & cos(x(iii)) + delz**2)**1.5
      
      top2 = delz - (delz*x(iii)**2)/2.
      
      & top2 = ((br(j)-lr(k))**2+br(j)*lr(k)*x(iii)**2 +
      
      85
\( \text{delz}^2)^{1.5} \)

\[ f_{kbar} = \frac{\text{top1}}{\text{bot1}} - \frac{\text{top2}}{\text{bot2}} \]

\[ f_k = f_k + f_{kbar} \]

141 \hspace{1cm} \text{continue} \]

\[ f_{kint} = hphi \cdot f_k \]

c now add in the analytical part that comes from kbar
c that was integrated analytically over psi (equation 3.35 -
c not including third term which is handled later).

\[ \text{dum} = (b_r(j) - l_r(k))^2 + \text{delz}^2 \]

\[ \text{top3} = \text{delz} \cdot \pi \]

\[ \text{bot3} = b_r(j) \cdot l_r(k) \cdot (\text{dum} + b_r(j) \cdot l_r(k) \cdot (\pi)^2)^{0.5} \]

\[ \text{top4} = \text{delz} \cdot (\text{alog}(\pi + \sqrt{(\pi)^2 + \text{dum} / (b_r(j) \cdot l_r(k))})) \]

\[ \text{bot4} = 2 \cdot b_r(j) \cdot l_r(k) \cdot (b_r(j) \cdot l_r(k))^{0.5} \]

\[ \text{analkb} = \frac{\text{top3}}{\text{bot3}} - \frac{\text{top4}}{\text{bot4}} \]

\[ f_{kdone} = \left( \frac{\text{lr}(k) \cdot d_gaml}_s(k) \cdot (f_{kint} + \text{analkb}) \right) \]

c now add on 3.31, and third term in 3.35 that were integrated
\c analytically over phi, but contained singularities when integrated
\c over s, and therefore needed to be dealt with in the linear zone.
c Bs not defined when j=nvortx; and should never be in danger zone
c when at core location since it does not lie along sheet.

\[ \text{if (k.eq.nvor.or.j.eq.nvortx) goto 164} \]

\[ b_{slowa} = b_s(j) - d_a \]

\[ b_{shgha} = b_s(j) + d_a \]

\[ \text{if(} l_s(k) \text{.gt.} b_{slowa} \text{.and.} l_s(k) \text{.lt.} b_{shgha} \text{) then} \]

\[ \text{c ***these lines are for plotting and analytical integration that} \]
\c ***is necessary for 3.31, and third term in 3.35 when integrated
\c ***over linear zone. \]

\[ \text{klma} = \text{klma} + 1 \]

\[ \text{IF (KLMA.EQ.1) THEN} \]

\[ \text{NPT1} = \text{KLMA} + \text{LKMA} \]

\[ \text{ELSE} \]

\[ \text{NPT1} = \text{NPT1} + 1 \]

\[ \text{ENDIF} \]

\[ \text{MARK1} = 1 \]

86
mflag = 1

**

lsdana(klma) = ls(k)

k1 = deval(bs(j),br,rbs,bs,nbr)
k2 = deval(bs(j),bz,zbs,bs,nbr)

a1 = (pi**2)*(br(j)**2)
b1 = -(pi**2)*br(j)*k1
c1 = k1**2 + k2**2

if(c1.ne.1) then
  c1 = 1
endif

bsls = bs(j) - ls(k)

c the additional analytically integrated terms for 3.31 and the third
c term in 3.36 are found following 3.31 continue.

c equation 3.31

cktop = k2*2*pi*dgams(k)*(br(j)-k1*bsls)

ckbot = c1*bsls*sqrt(a1+b1*bsls+c1*(bsls)**2)

ALTA = (cktop*bsls-2.*k2*dgamsbbs(j)*ckbot)/(bsls*ckbot)

c third term in 3.35, after it is handled like third term in equation
c 3.21 for axial velocity.

tp5chk = (k1*bsls)/br(j)

if (tp5chk.gt..0l) then

  tp5dan = ((bz(j)-lz(k))/br(j))*
&dgams(k)/(2*(br(j)*lr(k)**.5))*
&alog(abs((-pi+ sqrt((-pi)**2+(bsls**2/(br(j)**2-br(j)*kl*bsls))))
& /(bsls**2)/(2.*br(j)**2*pi)))

else

c when ls(k) really close to bs(j) use singularity fix plus additional

c expansion about 1.

  tp5dan = ((bz(j)-lz(k))/br(j))*
&dgams(k)/(2*(br(j)*lr(k)**.5))*
&alog(abs(1/(1-(K1*BSLS/BR(J)))))

endif

else

164 continue
c ***lines for plotting
lkma = lkma + 1

IF (MARK1.NE.1) THEN
   NPT1 = LKMA
ELSE
   NPT1 = NPT1 + 1
ENDIF

***end lines for plotting

c equation 3.31 when not in linear zone
top6 = lr(k)*dgams(k)*2*pi*delz
bot6 = dum*(dum+br(j)*lr(k)*((pi)**2))**.5

kdphi = top6/bot6

***third term in 3.35 when not in linear zone
TPSEL1 = ((BZ(J)-LZ(K))/BR(J))*
         (1./(2*lr(k)*(br(j)*lr(k))**.5))

TPSEL2 =
         alog(abs((-pi + ((-pi)**2 - dum/(br(j)*lr(k))**.5)))))
TPSELS = LR(K)*DGAMLS(K)*tp5el1*tp5el2

endif

c now doing midpoint method numerical scheme
fktot = fktot + hls*fkdone

SNGTP5 = SNGTP5 + HLS*(TPSDBN + TPSELS)

sumkdp = sumkdp + hls*(kdphi+alta)

continue

131

c summing up integration intervals inside and outside of linear zone.
npoi1 = lkma + klma

c if mflag = 0 never entered linear zone and do not do analytical
integration that comes from this zone.
if(mflag.eq.0) go to 879

***now for analytical terms coming from linear zone for equation 3.31
ex2 = bs(j) - lsdana(klma)
ex3 = bs(j)-lsdana(1)
integ2 = 2.*k2*dgams(j)*alog(abs(ex3/ex2))
analt2 = integ2
c analytical term stemming from third term in 3.35. Integral
c approximated as zero due to symmetry of graph of equation about
c marker location (graph not perfectly symmetrical
c that's why this is an approximation).
TPSEX = 0.0

`879  continue
186  continue`

c for the induced velocity in the r direction
c negative sign in next two cases is for change in coordinate

\[ \text{vindr}(j) = \frac{- (\text{sumkpdp} + \text{fktot} + \text{analt2} + \text{SNGTPS} + \text{TPSEX})}{4.0 \times \pi} \]

121  continue

c PREDICTOR-CORRECTOR TIME STEP
if(ncase.eq.1) then
nmark = nbr
else
nmark = nvortx
endif
IF(JJ.EQ.1) THEN
DO 911 J=1,NMARK
VZOLD(J) = VINDZ(J)
VROLD(J) = VINDR(J)
911  CONTINUE
ENDIF

IF(JJ.EQ.2) THEN
DO 912 J=1,NMARK
VINDZ(J) = (VZOLD(J) + VINDZ(J))/2.
VINDR(J) = (VROLD(J) + VINDR(J))/2.
912  CONTINUE
ENDIF
C END ADDITION FOR PREDICTOR-CORRECTOR

C now for the distance the particle is moved due to velocity in
C radial direction.
do 909 j = 1,nmark

IF(JJ.EQ.1) THEN
OLDCRE = RCORE
broid(j) = br(j)
ENDIF

\[ \text{br}(j) = \text{vindr}(j) \times \text{time} \]

909  continue
C now for the distance the particle is moved due to velocity in
C axial direction.

DO 999 J = 1,nmark

IF(JJ.EQ.1) THEN
  OLDCZE = ZCORE
  bzold(j) = bz(j)
ENDIF

bz(j) = vindz(j)*time

999 CONTINUE

C NOW WRITING RESULTS TO FILE INCASE CODE NEEDS TO BE STOPPED AND
C STARTED. ONLY WRITE WHEN JJ=2, CAUSE THAT'S THE SECOND ITERATION
C FOR PREDICTOR- CORRECTOR WHICH IS WHAT IS USED ON NEXT II ITERATION
C FOR LOOP 15.

if(ii.eq.1.and.jj.eq.2) then
  open(unit=62, file='wakegeom.out',status='unknown')
endif

if(ii.ne.1.and.jj.eq.2) then
  open(unit=62, file='wakegeom.out',status='unknown',
       access='append')
ENDIF

if(jj.eq.2) then
  open(unit=63, file='wakestore.dat',status='unknown')
  open(unit=65, file='miscstore.dat',status='unknown')
endif

if(jj.eq.2) then
  write(62,*) ii
endif

C END SAVING RESULTS

c      PUTTING THE MARKERS AT ITS NEW LOCATION

do 957 i = 1,nbr
  br(i) = br(i) + brold(i)
  bz(i) = bz(i) + bzold(i)
if(jj.eq.2) then
  write(62,*) br(i),bz(i)
  write(63,*) br(i),bz(i)
endif

957 continue

c      PUTTING THE CORE AT ITS NEW LOCATION

  rcore = br(nvortx) + OLDCRE
  zcore = bz(nvortx) + OLDCZE
br(nvortx) = rcore  
bz(nvortx) = zcore  

C SAVING RESULTS AGAIN FOR STOP/START ABILITY  
if(jj.eq.2) then  
  write(62,*), br(nvortx), bz(nvortx)  
  close(unit=62)  
  write(63,*), br(nvortx), bz(nvortx)  
  write(65,*), numbr, time, ii, rcore, zcore  
  close(unit=63)  
  close(unit=65)  
endif  

C END SAVING RESULTS  

C **********USE THIS PLOTTING FOR DISK CASE BECAUSE  
C **********WANT TO SEE RESULTS ON FIRST ITERATION  
C **********OF PREDICTOR-CORRECTOR. AT SECOND  
C **********ITERATION WILL SCREW-UP DUE TO SPIKES  
C **********AT TIP.  

C call plotting routine for vindz vs. bz  
  vindz(nbr) = 0.  
  nper(1) = nmark  
  titl(1) = 'veloc'  
  pltitl = 'br ''vindz '' 
  call grinit(5,6,'vindz vs br ')  
  call grklin(ilin,isym,nper,titl,1,br,vindz,pltitl,599)  

C call plotting routine for vindr vs. br  
  vindz(nbr) = 0.  
  nper(1) = nmark  
  titl(1) = 'veloc'  
  pltitl = 'br ''vindr '' 
  call grinit(5,6,'vindr vs br ')  
  call grklin(ilin,isym,nper,titl,1,br,vindr,pltitl,599)  

C 2007 LOOP FOR SECOND ITERATION OF PREDICTOR CORRECTOR  
2007  CONTINUE  

C WAKE GEOMETRY PLOT AT EACH TIME STEP AFTER CONVERGED  
C USING PREDICTOR-CORRECTOR.  
C call plotting routine for br vs. bz  
  nper(1) = nbr  
  nper(2) = 1  
  titl(1) = 'wake disp'  
  titl(2) = 'vortex pos'  
  pltitl = 'br ''bz ''
call grinit(5,6,'bz vs br')
c    call grklin(ilin,isym,nper,titl,2,br,bz,pltitl,509)
c ********** end plotting **********

c 15 is outer time loop
15    continue

go to 995

676    write(*,677)
677    format(1x,'e equals zero!!')
678    write(*,679)
679    format(1x,'q equals zero!!')
680    write(*,681)
681    format(1x,'q is positive !!')
682    write(*,683)
683    format(1x,'c is negative !!')

995    continue

    stop
end

FUNCTION DEVAL(SS,X,XS,S,N)
DIMENSION X(1), XS(1), S(1)
C--------------------------------------------
C Calculates dX/dS(SS)
C XS array must have been calculated by SPLINE
C WRITTEN BY PROFESSOR MARK DRELA, M.I.T.
C--------------------------------------------
ILOW = 1
I = N

10 IF(I-ILOW .LE. 1) GO TO 11

IMID = (I+ILOW)/2
IF(SS .LT. S(IMID)) THEN
    I = IMID
ELSE
    ILOW = IMID
ENDIF
GO TO 10

11 DS = S(I) - S(I-1)
    T = (SS - S(I-1)) / DS
CX1 = DS*XS(I-1) - X(I) + X(I-1)

CX2 = DS*XS(I) - X(I) + X(I-1)

DEVAL = X(I) - X(I-1) + (1.-4.0*T+3.0*T*T)*CX1 + T*(3.0*T-2.)*CX2

DEVAL = DEVAL/DS

RETURN

END ! DEVAL

SUBROUTINE DISK(BR,BZ,BS,BSMAX,GB,TBS,ndis1,NUMBR,a,t)

C!!! DONT FORGET TO DIMENSION VARIABLES IN MAIN PROGRAM

*     gam(100),     R(2001),     Z(2001),     zbs(2000),

DIMENSION NPER(2),ILIN(2),ISYM(2)

DATA ILIN/1,2/

DATA ISYM/1,2/

CHARACTER * 10 TITL(2)

CHARACTER * 30 PLTITL

pi = 4.*atan(1.)

C *******FOR STRAIGHT LINE DISTRIBUTION********

num = 150
ntotal = num
numbr = 150
do 65 j = 2,ntotal
   r(1) = .0053
   r(ntotal) = sin((pi/2)*(149.5-1)/(num-1))
   r(j) = sin((pi/2)*(j-1)/(num-1))
   z(j) = 0.0
65 continue

C *******END STRAIGHT LINE DISTRIBUTION********

C NOW TURN R AND Z VALUES INTO S VALUES.

bs(1) = r(1)
do 71 i = 2,ntotal
   delrsq(i) = (r(i) - r(i-1))**2
   delzsq(i) = (z(i)-z(i-1))**2
   dels(i) = sqrt(delrsq(i) + delzsq(i))
   bs(i) = dels(i) + bs(i-1)
71 continue

smax = bs(ntotal)

write(*,87) bs(ntotal)
87 format(ix,'smax=',f10.7)
CALL SPLIND(Z,ZBS,BS,NTOTAL,999.,999.)
CALL SPLIND(R,RBS,BS,NTOTAL,999.,999.)

c define br(1) = .01 cause otherwise setexp calls br(1) = 0.
c and setexp in vortex cannot deal with this.

bs(1) = .01
bsmax = bs(numbr)
do 66 j=1,numbr
   br(j) = seval(bs(j),r,rbs,bs,ntotal)
   bz(j) = seval(bs(j),z,zbs,bs,ntotal)
66 continue

C CHANGED NVORTX=NUMBR+1 TO NVORTX=NUMBR FOR CIRCULAR DISK
C SINCE THERE IS NO TIP VORTEX

nvortx = numbr

NPER(i) = numbr
TITL(1) = 'INIT WAKE'
PLTITL = 'RADIUS Z/R'
CALL GRINIT(5,6,'PLOT OF INITIAL DISK WAKE')
CALL GRKLIN(ILIN,ISYM,NPER,TITL,1,BR,BZ,PLTITL,599)

u = .02
ndisi = numbr
do 27 i = 1,numbr
   tbs(i) = sin((pi/2)*(i-1)/(num-1))
   gb(i) = (-(4*u)/pi)*sqrt(1.-tbs(i)**2)
27 continue

NPER(i) = ndisi
TITL(1) = 'DISK CIRCULATION DISTRIBUTION'
PLTITL = 'TBS GB'
CALL GRINIT(5,6,'PLOT OF GB VS S')
CALL GRKLIN(ILIN,ISYM,NPER,TITL,1,tbs,gb,PLTITL,599)

CALL SPLIND(GB,GBBTBS,tbs,ndisi,999.,999.)
do 92 j=1,numbr
   gam(j) = seval(bs(j),gb,gbtbs,br,ndisi)
92 continue

NPER(i) = numbr
TITL(1) = 'DISK CIRCULATION DISTRIBUTION'
PLTITL = 'BS GAM'
CALL GRINIT(5,6,'PLOT OF GAM VS BS')

94
The part of this routine that uses Pullin circulation distribution works except since the points are hand-picked off a plot, waves in the wake soon come into play. If I could curve fit the data for the gamma distribution from .95 out that would be a solution, but can't find appropriate curve fit. Instead shifted ellipse over and up (using an equation) to try to closely represent pullin distribution. This works very well.

```
SUBROUTINE ELLIP(BR,BZ,BS,BSMAX,GB,TBS,ndisl,NUMBR,a,t)
C!!DONT FORGET TO DIMENSION VARIABLES IN MAIN CODE

  * rbs(2000), P(301), PBS(301), TBS(2001),

DIMENSION NPER(2),ILIN(2),ISYM(2)
DATA ILIN/1,2/
DATA ISYM/1,2/
CHARACTER * 10 TITL(2)
CHARACTER * 30 PLTITL

real lampul

NUM = 19
ZETA(1) = 64.7997
```
do 51 i = 1,num
   xplt(i) = zeta(i)
   yyplt(i) = eta(i)
51 continue

c Now the zeta and eta values above (which are pulled directly
c from Pullin's plots) will be curve-fitted and overwritten
c so instabilities do not occur due to errors in transcribing
c Pullin's data.

int1 = ((65-10)/5) + .1
int2 = ((10-3)/.05) + .1
int3 = ((3-1)/.01) + .1
int4 = ((1-.083)/.001) + 1.1
dec2 = int1 + int2
dec3 = dec2 + int3
ntotal = int1 + int2 + int3 + int4
write(*,*) ntotal
zet = 65
ainc = 0.0
do 61 i = 1,ntotal
  zet = zet - ainc
  zeta(i) = zet
  eta(i) = .04*(((1/zeta(i)) - (1./64.7997))
  ind = num + i
  xplt(ind) = zeta(i)
  yyplt(ind) = eta(i)
  if(i.ge.int1) then
      ainc = 5
  endif
  if(i.gt.int1.and.i.le.dec2) then
      ainc = .05
  endif
  if(i.gt.dec2.and.i.le.dec3) then
      ainc = .01
  endif
  if(i.gt.dec3) then
      ainc = .001
  endif
61 continue

C PLOT WITH NEW CURVE FIT VALUES WHERE PULLIN ZETA AND ETA HAVE
C ALREADY BEEN OVERWRITTEN IN LOOP 61.
NPER(1) = ntotal
TITL(1) = 'PULLIN'
PLTITL = 'ZETA ETA'
CALL GRINIT(5,6,'PLOT OF ETA VS ZETA')
c
CALL GRKLIN(ILIN,ISYM,NPER,TITL,1,zeta,eta,PLTITL,599)
c
CALL SPLIND(ETA,EBZ,ZETA,Ntotal,999.,999.)

A = .01396
T = .1373
DO 1 I = 1,NTOTAL
  c since old coordinate system was zero at tip,
c 1.- ..... is necessary.
  R(I) = 1. - ((A*T)**(2./3.))*ZETA(I))
  c negative is because Pullin has z pos upward.
  Z(I) = -(A*T)**(2./3.)*ETA(I)
1 CONTINUE
C NOW TURN R AND Z VALUES INTO S VALUES.

\[
s(1) = r(1)
\]
do 71 i = 2,ntotal
\[
\begin{align*}
delrsq(i) &= (r(i) - r(i-1))^2 \\
delzsq(i) &= (z(i) - z(i-1))^2 \\
dels(i) &= \sqrt{delrsq(i) + delzsq(i)} \\
s(i) &= \text{dels}(i) + s(i-1)
\end{align*}
\]
71 continue
write(*,87) s(ntotal)
87 format(ix,'smax=',f10.7)

CALL SPLIND(Z,ZBS,S,NTOTAL,999.,999.)
CALL SPLIND(R,RBS,S,NTOTAL,999.,999.)

c Setext lays-out the markers through an exponential
stretching technique.

DS1 = .025
DSN = .0005
NUMBR = 150
SMAX = S(NTOTAL)
CALL SETEX2(BS,DS1,DSN,SMAX,NUMBR)

c define br(1) = .01 cause otherwise setexp calls br(1) = 0.
c and setexp in vortex cannot deal with this.

bs(1) = .01
bsmax = bs(numbr)
do 66 j=1,numbr
\[
\begin{align*}
br(j) &= \text{seval}(bs(j),r,rbs,s,ntotal) \\
bz(j) &= \text{seval}(bs(j),z,zbs,s,ntotal)
\end{align*}
\]
66 continue
nvortx = numbr + 1
br(nvortx) = .9952
bz(nvortx) = -.007684

NPER(1) = numbr
nper(2) = 1
TITL(1) = 'INIT WAKE'
TITL(2) = 'CORE LOC'
PLTITL = 'RADIUS Z/R'
CALL GRINIT(5,6,'PLOT OF INITIAL PULLIN WAKE')

c CALL GRKLIN(ILIN,ISYM,NPER,TITL,2,br,bz,PLTITL,599)
C NOW FOR THE CIRCULATION DISTRIBUTION THAT IS TAKEN DIRECTLY
C FROM PULLIN'S PLOT (ALL THE LINES COMMENTED OUT).
C THIS IS NEVER USED, BUT OVERWRITTEN
C BY A SOLUTION THAT CLOSELY APPROXIMATES THIS SOLUTION IMMEDIATELY
C FOLLOWING ALL THE COMMENTS.

c    NUM2 = 15
C    TBS(1) = .9532
C    P (1) = .3063
C    TBS(2) = .9585
C    P (2) = .2878
C    TBS(3) = .9629
C    P (3) = .2709
C    TBS(4) = .9684
C    P (4) = .2532
C    TBS(5) = .9734
C    P . (5) = .2313
C    TBS(6) = .9776
C    P (6) = .2101
C    TBS(7) = .9815
C    P (7) = .1930
C    TBS(8) = .9847
C    P (8) = .1775
C    TBS(9) = .9877
C    P (9) = .1637
C    TBS(10) = .9903
C    P (10) = .1539
C    TBS(11) = .9931
C    P (11) = .1409
C    TBS(12) = .9956
C    P (12) = .1311
C    TBS(13) = .9977
C    P (13) = .1238
C    TBS(14) = .9998
C    P (14) = .1197
C    TBS(15) = 1.0018
C    P (15) = .1124

C    CALL SPLIND(P,PBS,TBS,NUM2,999.,999.)

C    DO 5 I = 1,NUM2

C    HA = .70741
C    HT = .00271
C    if(i.eq.num2) then
C    LAMPUL(I) = p(num2)/((HA**(4/3))*(HT**(1/3)))
C    else
C    endif
C    LAMPUL(I) = p(i)/((HA**(4/3))*(HT**(1/3)))

C    GB(I) = LAMPUL(I)*(A**(4/3))*(T**(1/3))
c xplt(i) = tbs(i)
c yplt(i) = gb(i)

CONTINUE

NPET(1) = NUM2
TITL(1) = 'Hoej'
PLTITL = 'tbs gb'
CALL GRINIT(5,6,'for tbs gt .95')
CALL GRKILN(ILIN,ISYM,PER,TITL,1,TBS,gb,PLTITL,599)

AINC = .01
NDIS = .95/AINC + .01
nend = ndis + 1
NUM3 = Nend + 1
NUM4 = Nend + NUM2 - 1
DO 4 I = NUM3,NUM4

NUM5 = I-NDIS
GB(I) = GB(NUM5)
TBS(I) = TBS(NUM5)

CONTINUE

rad = -AINC
ind = nend
DO 6 I = 1,Nend
rad = rad + AINC
tbs(i) = rad
tbs(1) = 0.
tbs(nend) = .95
gb(i) = .02*((1.-tbs(i)**2)**.5
if(i.le.NUM) then
ind = ind + 1
tbs(ind) = xplt(i)
gb(ind) = yplt(i)
endif
6 continue

npts = nend + NUM2
NPET(1) = Npts
TITL(1) = 'actual'
PLTITL = 'tbs gb'
CALL GRINIT(5,6,'Pullin Circ')
CALL GRKILN(ILIN,ISYM,PER,TITL,1,TBS,gb,PLTITL,599)

***********ONLY LEAVE NDIS1=NUM4, WHEN NOT USING SHIFTED ELLIPSE
ndis1 = num4

C******************************************************************************

C******************************************************************************

C THIS VERY CLOSELY APPROXIMATES PULLIN'S SOLUTION.

ainc = .01
ndis = 1./ainc + 1.01
rad = -ainc
ndis1 = ndis + 1
DO 27 I = 1,NDIS1
    rad = rad + ainc
    tbs(i) = rad
    tbs(1) = 0.
    tbs(ndis) = 1.0
    tbs(ndis1) = 1.0018
    gb(i) = .02*(((1.-tbs(i)-.01)**2)**.5 - (.02* &
                   sqrt(1.-(1.0018 - .01)**2)) + .0022
27    continue

CALL SPLIND(GB,GBTBS,TBS,Ndis,999.,999.)
NPER(1) = Ndis1
TITL(1) = 'Approx Pullin'
PLTITL = ''TBS "GB ''
CALL GRINIT(5,6,'PLOT OF GB VS S')
CALL GRKLIN(ILIN,ISYM,NPER,TITL,1,TBS,GB,PLTITL,599)
C********end shifted ellipse

return
end ! pullin

SUBROUTINE FILDIS(BS,bsmax,NUMBR,expls,XAXIS,YAXIS,NPOIN,klm,j, &
                   n6r,bslow,bshigh,dan,amark,vardint)

C Melinda Godwin March 16,1991

101
& xaxis(2000), xplt3(2000),

dimension nper(2), ilin(2), isym(2)
real ls
data ilin/0,1/
data isym/1,2/
character * 20 titl(2)
character * 25 pltitl

c ***set-up for linear zone********

NFLAG = 0

c setting del (half linear region size)
danbac = bs(j) - bs(j-1)
danfor = bs(j+1) - bs(j)

if (j.eq.1) then
danbac = 1000
endif

if (j.eq.numbr) then
danfor = 1000
endif

if (danbac.lt.danfor) then
dan = danbac
else
dan = danfor
endif

nxy = numbr - 1
if (j.eq.nxy) then
dan = danfor
nflag = 1
endif

bshigh = bs(j) + dan
bslow = bs(j) - dan

if (j.eq.1) then
bslow = bs(1)
endif
if (j.eq.numbr) then
bshigh = bs(numbr)
endif

danzne = bshigh - bslow

c nspac = 5 gives 5 spacings in linear zone, therefore 4 fil

if (j.eq.numbr.or.j.eq.1) then
  c use only half the number of integration intervals plus one
  c when the linear zone is one sided.
  c This puts extra filament at end so linear
  c area is treated like all other linear zones.
  c
    nspac = 3
  else
    nspac = 6
  endif

filspc = danzne/nspac

IF (NFLAG.NE.1) THEN
  nfil = 5
ELSE
  c For second to last marker need an extra integration interval since
  c there is no non-linear zone to right of this marker (there is only
  c the linear zone). This extra integration interval is necessary
  c so later the midpoint can be taken between the second to last and
  c last (now added) integration interval.
    NFIL = 6
ENDIF

if (j.eq.numbr.OR.J.EQ.1) then
  nfil = 3
endif

c **************end set-up for linear zone**************

c start set-up for how integration intervals are layed-out to the
  c left of the linear zone.

dint = .002

if (j.eq.1) then
  nml=0
  goto 299
else

103
nnl = (bslow/dint) + 1
endif

if (nnl.le.2) then
  nnl = 3
endif

if (nnl.le.3) then
  ds1 = .0005
  dsn = .0005
else
  ds1 = .0005
  dsn = .001
endif

smax = bslow

call setex2(s,ds1,dsn,smax,nnl)

do 21 i = 1,nnl
   k = nnl-i+1
   expls(i) = s(i)
   xplt(i) = expls(i)
   yplt(i) = 0.
21  continue

expls(1) = 0.
xplt(i) = 0.

299  continue

c end set-up for integration intervals to left of linear zone.
c begin set-up for integration intervals in linear zone.

klm = 0
nnew = nnl + 1
nnew1 = nnl + nfil
if (j.eq.1) then
  updat = bslow - filspc
else
  updat = bslow
endif

do 23 i = nnew,nnew1
   klm = klm + 1
   updat = updat + filspc

104
expls(i) = updat
xpltl(klm) = expls(i)
yppltl(klm) = 0.0
23 continue

c end set-up for linear zone

c begin set-up for integration intervals to right of linear zone

if (bshigh.ge.bsmax) then
  nnr = 0
else
  c The following is done to get more integration strips out near the
c tip for markers near the tip because of the high curvature in this
c area. It was necessary for the translating disk case to keep
accuracy high, but was too computationally time consuming to
apply to the ellip distrib in full form (because of stretching),
c and not applied to the typical hovering case at all because in this
c case the sheet is stretched even more than for the elliptical case.
c It is important to note that as the sheet stretches bsmax increases
c and since this dictates the number of integration intervals the
c number of integration intervals increases as well (more integ.
c interv. means increase in computational time).
c The values given to amark and vardint can be found in main code.
  if(bs(j).gt.amark) then
    dint = vardint
    endif
  if(bs(j).le.9) then
    dint = .002
  endif
  nnr = (bsmax-bshigh)/dint + 1
endif

dint = .002

if (nnr.le.3.and.nnr.ne.0) then
  nnr = 4
endif

n5r = nnr + nnl + nfl
n6r = nnr + nnl + nfl - 1
nlpd = nnl + nfl + 1

if(nnr.eq.0) goto 160

if (nnr.le.4) then
  ds1 = (bsmax - bshigh)/5
  dsn = ds1
else
  ds1 = .001
  dsn = .001
endif
if(bs(j).gt.8) then
  dsl = (bsmax - bshigh)/nnr
  dsn = dsl
endif

smax = bsmax-bshigh
call setex2(s,ds1,dsn,smax,nnr)

ind = nnl
do 22 i = nlpd,n5r
  m = i-nlpd+1
  ind = ind + 1
  temps(i) = s(m)
  expls(i) = bshigh+temps(i)
  xplt(ind) = expls(i)
  yplt(ind) = 0.0
  xplt3(m) = expls(i)
  yplt3(m) = 0.0
22 continue

expls(n5r) = bsmax

160 continue
c end set-up for integration intervals to right of linear zone.
c **************************LOTTING**************************
npoin = nnr + nnl + nfil
do 150 i = 1,npoin
  if(i.le.klm) then
    xaxis(i) = xpltl(i)
    yaxis(i) = ypltl(i)
  else
    index = i - klm
    xaxis(i) = xplt(index)
    yaxis(i) = yplt(index)
  endif
150 continue

if (bs(j).ge.995.and.bs(j).le.997) then
  nx = npoin - klm
  nper(1) = klm
  nper(2) = nx
  titl(1) = 'fildis'
  titl(2) = 'fildis'
  pltitl = 'expls '0 '
call grinit(S,6,'exp stretched ls')
SUBROUTINE FILTER(T,Y,N,TFIL)
PARAMETER (NX=1000)
REAL T(N), Y(N), tfil(n)
REAL AA(NX), BB(NX), CC(NX)

C WRITTEN BY PROFESSOR MARK DRELA, MIT
C
IF(N.GT.NX) STOP 'FILTER: Array overflow'
C
CC(1) = 0.
AA(1) = 1.0
DO 10 I=2, N-1
  TFSQ = TFIL(i)**2
  DTM = T(I) - T(I-1)
  DTP = T(I+1) - T(I)
  DT = 0.5*(T(I+1) - T(I-1))
  CC(I) = -TFSQ / (DTP*DT)
  AA(I) = TFSQ * (1.0/DTP + 1.0/DTM)/DT + 1.0
  BB(I) = -TFSQ / (DTM*DT)
10 CONTINUE
BB(N) = 0.
AA(N) = 1.0
C
CALL TRISOL(AA,BB,CC,Y,N)
C
RETURN
END ! FILTER

SUBROUTINE SETEXP(S,DS1,SMAX,NN)
C Sets geometrically stretched array S:
C
C WRITTEN BY PROFESSOR MARK DRELA, M.I.T.
C
C S(i+1) - S(i) = r * [S(i) - S(i-1)]
C
C S (output) array to be set
C DS1 (input) first S increment: S(2) - S(1)
C SMAX (input) final S value: S(NN)
C NN (input) number of points
C
C REAL S(1)
C
107
SIGMA = SMAX/DS1
NEX = NN-1
RNEX = FLOAT(NEX)
RNI = 1.0/RNEX

C
C----- solve quadratic for initial geometric ratio guess
AAA = RNEX*(RNEX-1.0)*(RNEX-2.0) / 6.0
BBB = RNEX*(RNEX-1.0) / 2.0
CCC = RNEX - SIGMA

C
DISC = BBB**2 - 4.0*AAA*CCC
DISC = AMAXI( 0.0 , DISC )

C
IF(NEX.LE.1) THEN
STOP 'SETEXP: Cannot fill array. N too small.'
ELSE IF(NEX.EQ.2) THEN
RATIO = -CCC/BBB + 1.0
ELSE
RATIO = (-BBB + SQRT(DISC))/(2.0*AAA) + 1.0
ENDIF

C
IF(RATIO.EQ.1.0) GO TO 11
C
C----- Newton iteration for actual geometric ratio
DO 1 ITER=1, 100
SIGMAN = (RATIO**NEX - 1.0) / (RATIO - 1.0)
RES = SIGMAN**RNI - SIGMA**RNI
DRESDR = RNI*SIGMAN**RNI &
        = (RNEX*RATIO**((NEX-1) - SIGMAN) / (RATIO**NEX - 1.0)

C
DRATIO = -RES/DRESDR
RATIO = RATIO + DRATIO
C
IF(ABS(DRATIO) .LT. 1.0E-5) GO TO 11
C
1 CONTINUE
WRITE(6,*) 'SETEXP: Convergence failed. Continuing anyway ...'

C
C----- set up stretched array using converged geometric ratio
11 S(1) = 0.0
DS = DS1
DO 2 N=2, NN
S(N) = S(N-1) + DS
DS = DS*RATIO
2 CONTINUE
C
RETURN
END ! SETEXP

SUBROUTINE SETEX2(S,DS1,DSW,SMAX,N)
DIMENSION S(N)
C............................................................................................
C Sets array S stretched so that a prescribed spacing is
obtained at each end. The interior spacing is a blend
of two geometric stretchings "shot" from each end.

WRITTEN BY PROFESSOR MARK DRELA, M.I.T.

PARAMETER (NDIM=2000)
REAL S1(NDIM), SN(NDIM)

with bigger FEND, the actual end increments will get closer to DS1 & DSN,
but the interior spacing might get screwy.
DATA FEND / 0.5 /

IF(N.GT.NDIM) STOP 'SETEX2: Array overflow.'

---- calculate spacing arrays each having the prescribed end increment
CALL SETEXP(S1,DS1,SMAX,N)
CALL SETEXP(SN,DSN,SMAX,N)

---- blend spacing arrays with power-function weights
DO 10 I=1, N
   IN = N-I+1
   SS1 = S1(I)
   SSN = SMAX - SN(IN)

   WT1 = FLOAT(N-I)**FEND
   WTN = FLOAT(I-1)**FEND

   WTI = (1.0 - SSN/SMAX)**FEND
   WTN = (SS1/SMAX)**FEND

   S(I) = (SS1*WTI + SSN*WTN) / (WT1 + WTN)
10 CONTINUE

---- check for monotonicity
DO 20 I=2, N
   IF(S(I) .LE. S(I-1)) THEN
      WRITE(6,*) 'SETEX2: Warning. Returned array not monotonic.'
      RETURN
   ENDIF
20 CONTINUE

RETURN
END ! SETEX2

FUNCTION SEVAL(SS,X,XP,S,N)
REAL X(1), XP(1), S(1)
C
WRITTEN BY PROFESSOR MARK DRELA, M.I.T.
C
IF((SS-S(N)) .GT. 0.1*(S(N)-S(N-1)) .OR.
& (SS-S(1)) .LT. 0.1*(S(1)-S(2)) ) then
   write(*,172)
C 172  format('seval:outside range')
endif

ILOW = 1
I = N

10 IF(I-ILOW .LE. 1) GO TO 11
C
IMID = (I+ILOW)/2
IF(SS .LT. S(IMID)) THEN
   I = IMID
ELSE
   ILOW = IMID
ENDIF
GO TO 10
C
11 DS = S(I) - S(I-1)
   T = (SS-S(I-1)) / DS
   CX1 = DS*XP(I-1) - X(I) + X(I-1)
   CX2 = DS*XP(I) - X(I) + X(I-1)
   SEVAL = T*X(I) + (1.0-T)*X(I-1) + (T-T*T)*((1.0-T)*CX1 - T*CX2)
RETURN
END ! SEVAL

SUBROUTINE SMOOTH(BR,BZ,BS,NBR,bsmax,ii,sminc)
C
MELINDA GODWIN
C MAY 23,1991

DIMENSION BR(301), BZ(301), BS(301), DELRSQ(301), DELZSQ(301),
& DELS(301), TFIL(2000)

character * 12 titl(3)
character * 16 pltitl

write(*,877) sminc
877 format(ix,'tfil=',f10.4)

if(ii.eq.1) then
   write(*,47)
47 format(ix,'INPUT CASE NUMBER, 1-DISK, 2-ELLIP, 3-TYP HOW ROTOR',
&/,
read(S,*) mcase

write(*,48) mcase
48 format(1x,'YOU CHOSE CASE',1x,12,1x,
       & 'ENTER 1 IF TRUE, 2 IF FALSE',/,
read(S,*) nchoice

if(nchoice.eq.2) then
write(*,49)
49 format(1x,'INPUT CASE NUMBER, 1-DISK, 2-ELLIP, 3-TYP HOW ROTOR')
read(S,*) mcase

write(*,50) mcase
50 format(1x,'YOU ARE RUNNING CASE',1x,i2)
endif

endif

nend = nblr-2

do 2003 i = 2,nend,

  delr1=br(i)-br(i-1)
  delz1=bz(i)-bz(i-1)
  delr2=br(i+1)-br(i)
  delz2=bz(i+1)-bz(i)
  delr3=br(i+2)-br(i+1)
  delz3=bz(i+2)-bz(i+1)
  cross1 = delr1*delz2 - delz1*delr2
  cross2 = delr2*delz3 - delz2*delr3

  flag=cross1*cross2

  if (mcase.eq.1.or.mcase.eq.2) then
    start = 0.0
  else
    start = .96
  endif

  if(flag.lt.0..and.br(i).gt.start) then
    tfil(i) = sminc
  else
    tfil(i)=0.0
  endif

  if(flag.lt.0.and.i.eq.nend) then
    if(sminc.eq.0.) then
      111.
2003 continue
	nolast = nbr

call filter(bs,br,nolast,tfil)
call filter(bs,bz,nolast,tfil)

bs(1) = BR(1)
do 18 i = 2,nbr
   delrsq(i) = (br(i) - br(i-1))**2.
   delzsq(i) = (bz(i) - bz(i-1))**2.
   dels(i) = sqrt(delrsq(i) + delzsq(i))
   bs(i) = dels(i) + bs(i-1)
18 continue
BSMAX = BS(NBR)

return
end

SUBROUTINE SPLIND(X,XS,S,N,XS1,XS2)
DIMENSION X(1),XS(1),S(1)
PARAMETER (NMAX=2000)
DIMENSION A(NMAX),B(NMAX),C(NMAX)

C---------------------------------------------------
C WRITTEN BY PROFESSOR MARK DRELA, M.I.T.
C---------------------------------------------------
C Calculates spline coefficients for X(S).
C Specified 1st derivative and/or usual zero 2nd
C derivative end conditions are used.
C To evaluate the spline at some value of S,
C use SEVAL and/or DEVAL.
C S independent variable array (input)
C X dependent variable array (input)
C XS dX/dS array (calculated)
C N number of points (input)
C XS1,XS2 endpoint derivatives (input)
C If = 999.0, then usual zero second
C derivative end condition(s) are used
C---------------------------------------------------

IF(N.GT.NMAX) STOP 'SPLIND: array overflow, increase NMAX'
C
112
DO 1 I=2, N-1
  DSM = S(I) - S(I-1)
  DSP = S(I+1) - S(I)
  B(I) = DSP
  A(I) = 2.0*(DSM+DSP)
  C(I) = DSM
  XS(I) = 3.0*(((X(I+1)-X(I))*DSM/DSP + (X(I)-X(I-1))*DSP/DSM)
  1 CONTINUE
C
IF(XS1.EQ.999.0) THEN
C----- set zero second derivative end condition
  A(1) = 2.0
  C(1) = 1.0
  XS(1) = 3.0*(X(2)-X(1))/(S(2)-S(1))
else if (xs1.eq.-999.) then
  A(1) = 1.0
  C(1) = 1.0
  XS(1) = 2.0*(X(2)-X(1))/(S(2)-S(1))
ELSE
C----- set specified first derivative end condition
  A(1) = 1.0
  C(1) = 0.
  XS(1) = XS1
ENDIF
C
IF(XS2.EQ.999.0) THEN
  B(N) = 1.0
  A(N) = 2.0
  XS(N) = 3.0*(X(N)-X(N-1))/(S(N)-S(N-1))
else if (xs2.eq.-999.0) then
  A(N) = 1.0
  C(N) = 1.0
  XS(N) = 2.0*(X(N)-X(N-1))/(S(N)-S(N-1))
ELSE
  A(N) = 1.0
  B(N) = 0.
  XS(N) = XS2
ENDIF
C
C---- solve for derivative array XS
CALL TRISOL(A,B,C,XS,N)
C
RETURN
END ! SPLIND

SUBROUTINE TRISOL(A,B,C,D,KK)
DIMENSION A(1),B(1),C(1),D(1)
C-----------------------------------------
C Solves KK long, tri-diagonal system
C             A     C     D       |       |
C             B     A     C     D       |       |
C             B     A       .         |       |
C             .     .     C       .       |
C
113
The righthand side D is replaced by the solution. A, C are destroyed.

DO 1 K=2, KK
   KM = K-1
   C(KM) = C(KM) / A(KM)
   D(KM) = D(KM) / A(KM)
   A(K) = A(K) - B(K) * C(KM)
   D(K) = D(K) - B(K) * D(KM)
   1 CONTINUE

D(KK) = D(KK) / A(KK)

DO 2 K=KK-1, 1, -1
   D(K) = D(K) - C(K) * D(K+1)
   2 CONTINUE

RETURN
END ! TRISOL
SUBROUTINE TYPHOV(BR,BZ,BS,BSMAX,GB,TBS,ndisl,NUMBR,A,T)
c Uses Pullin self-similar solution with typical hover gamma distrib.

C DONT FORGET TO DIMENSION VARIABLES IN MAIN PROGRAM

   * P(301), PBS(301), TBS(2001), BS(2001),

DIMENSION NPER(2),ILIN(2),ISYM(2)
DATA ILIN/1,2/
DATA ISYM/1,2/
CHARACTER * 10 TITL(2)
CHARACTER * 50 PLTITL

real lampul

NUM = 19
ZETA(1) = 64.7997
ETA (1) = 0.0
ZETA(2) = 45.0
ETA (2) = 0.0
ZETA(3) = 25.9200
ETA (3) = 0.0
ZETA(4) = 6.48
ETA (4) = 0.0
ZETA(5) = 3.2400
ETA (5) = 0.0
ZETA(6) = 2.0600
ETA (6) = 0.0
ZETA(7) = 1.7250
ETA (7) = .0063
ZETA(8) = 1.4438
ETA (8) = .0125
ZETA(9) = 1.0863
ETA (9) = .0281
ZETA(10) = .8875
ETA (10) = .0438
ZETA(11) = .7125
ETA (11) = .0613
ZETA(12) = .5375
ETA (12) = .08625
ZETA(13) = .3625
ETA (13) = .1375
ZETA(14) = .2250
ETA (14) = .2163
ZETA(15) = .1781
ETA (15) = .2531
ZETA(16) = .1375
ETA (16) = .3
ZETA(17) = .10625
ETA (17) = .3625
zeta(18) = .09
ta(i) = zeta(i)
eta(i) = eta(i)
continue

C Now the zeta and eta values above (which are pulled directly
C from Pullin’s plots) will be curve-fitted and overwritten
C so instabilities do not occur due to errors in transcribing
C Pullin’s data.

int1 = ((65-10)/5) + .1
int2 = ((10-3)/.05) + .1
int3 = ((3-1)/.01) + .1
int4 = ((1-.083)/.001) + 1.1
dc2 = int1 + int2
dc3 = dc2 + int3
ntotal = int1 + int2 + int3 + int4
write(*,*) ntotal
zet = 65
ainc = 0.0
do 51 i = 1,ntotal
zet = zet - ainc
zeta(i) = zet
eta(i) = .04*(((1/zeta(i)) - (1./64.7997))
ind = num + 1
xplt(ind) = zeta(i)
yplt(ind) = eta(i)
if(i.le.int1) then
  ainc = 5
endif
if(i.gt.int1.and.i.le.dec2) then
  ainc = .05
endif
if(i.gt.dec2.and.i.le.dec3) then
  ainc = .01
endif
if(i.gt.dec3) then
  ainc = .001
endif
61 continue

C PLOT WITH NEW CURVE FIT VALUES WHERE PULLIN ZETA AND ETA HAVE
C ALREADY BEEN OVERWRITTEN IN LOOP 61.
NPER(1) = ntotal
TITL(1) = 'PULLIN'
PLTITL = 'ZETA ETA'
CALL GRINIT(5,6,'PLOT OF ETA VS ZETA')
c CALL GRKLIN(ILIN,ISYM,NPER,TITL,I,zeta,eta,PLTITL,599)
c
CALL SPLIND(ETA,EBZ,ZETA,Ntotal,999.,999.)

A = .0418
T = .0458
DO I = 1, NTOTAL
  c since old coordinate system was zero at tip,
  c 1.- ...... is necessary.
  R(I) = 1. - ((A*T)**(2./3.)*ZETA(I))
  c Negative is because Pullin has z pos upward.
  Z(I) = -(A*T)**(2./3.)*ETA(I)
1 continue

C NOW TURN R AND Z VALUES INTO S VALUES.
s(i) = r(i)
do 71 i = 2, ntotal
   delrsq(i) = (r(i) - r(i-1))**2
   delzsq(i) = (z(i) - z(i-1))**2
   dels(i) = sqrt(delrsq(i) + delzsq(i))
   s(i) = dels(i) + s(i-1)
71 continue
write(*,87) s(ntotal)
87 format(1x,'smax=',f10.7)

CALL SPLIND(Z,ZBS,S,NTOTAL,999.,999.)
CALL SPLIND(R,RBS,S,NTOTAL,999.,999.)

 c Now s values will be stretched.

 DS1 = .025
 DSN = .0005
 NUMBR = 150
 SMAX = S(NTOTAL)
 CALL SETEX2(BS,DS1,DSN,SMAX,NUMBR)

 c define br(1) = .01 cause otherwise setexp calls br(1) = 0.
 c and setexp in vortex cannot deal with this.

 bs(1) = .01
 bsmax = bs(numbr)
 do 66 j=1,numbr
    br(j) = seval(bs(j),r,rbs,s,ntotal)
    bz(j) = seval(bs(j),z,zbs,s,ntotal)
66 continue

 nvortex = numbr + 1
 super = 2./3.
 scale = (a*t)**super
 br(nvortex) = 1.-.308*scale
 bz(nvortex) = -.498*scale

 NPER(1) = numbr
 nper(2) = 1
 TITL(1) ='INIT WAKE'
 TITL(2) ='CORE LOC'
 PLTITL = '-RADIUS "Z/R"'
 CALL GRINIT(5,6,'INITIAL WAKE FOR PRACTICAL DISTRIBUTION')

 c CALL GRKLIN(ILIN,ISYM,NPER,TITL,2,br,bz,PLTITL,599)

 c
 C NOW FOR THE CIRCULATION DISTRIBUTION EXACTLY FROM PULLIN'S
 C SOLUTION.
NUM2 = 15
TBS(1) = .9532
P (1) = .3063
TBS(2) = .9585
P (2) = .2878
TBS(3) = .9629
P (3) = .2709
TBS(4) = .9684
P (4) = .2532
TBS(5) = .9734
P (5) = .2313
TBS(6) = .9776
P (6) = .2101
TBS(7) = .9815
P (7) = .1930
TBS(8) = .9847
P (8) = .1775
TBS(9) = .9877
P (9) = .1637
TBS(10) = .9903
P (10) = .1539
TBS(11) = .9931
P (11) = .1409
TBS(12) = .9956
P (12) = .1311
TBS(13) = .9977
P (13) = .1238
TBS(14) = .9998
P (14) = .1197
TBS(15) = 1.0018
P (15) = .1124

CALL SPLIND(P,PBS,TBS,NUM2,999.,999.)

DO 5 I = 1,NUM2

    HA = .70741
    HT = .00271

    LAMPUL(I) = p(i)/((HA**(4/3))*(HT**(1/3)))
    GB(I) = LAMPUL(I)*(A**(4/3))*(T**(1/3))
    xplt(i) = tbs(i)
    yplt(i) = gb(i)

      5 CONTINUE

NPER(1) = NUM2
TITL(1) = 'Hoej'
PLTITL = '"tbs "gb "'
CALL GRINIT(6,6,'for tbs gt .95')
CALL GRKLIN(ILIN,ISYM,NPER,TITL,1,TBS,gb,PLTITL,599)

118
num99 = 9

c This is the part of the circulation distribution < .93, which
c was taken from plots referenced in thesis. Pullin's solution
c is not applied to this. It is only layed-out here so that
c the tip circulation before and after the curve fit can be plotted
c and compared.

    tempbs(1) = .1
    tempgb(1) = .0
    tempbs(2) = .2
    tempgb(2) = .0072
    tempbs(3) = .3
    tempgb(3) = .01
    tempbs(4) = .42
    tempgb(4) = .012
    tempbs(5) = .7
    tempgb(5) = .0152
    tempbs(6) = .84
    tempgb(6) = .018
    tempbs(7) = .915
    tempgb(7) = .0216
    tempbs(8) = .94
    tempgb(8) = .0208
    tempbs(9) = .95
    tempgb(9) = .0188

CALL SPLIND(TEMPGB,TGBTBS,TEMPBS,NUM99,999.,999.)

    ainc = .01
    disend = .94
    NDIS = disend/ainc + .01
    nend = ndis + 1
    rad = -ainc
    ind = nend

DO 662 I = 1,Nend

    rad = rad + ainc
    tbs(i) = rad
    tbs(i) = 0.
    tbs(nend) = disend
    gb(i) = seval(tbs(i),tempgb,tgbtbs,tempbs,num99)

    if(i.le.num2) then
        ind = ind+1
        tbs(ind) = xplt(i)
        gb(ind) = yplt(i)
    endif

662   continue
npts = num2 + nend
NPER(1) = Npts
TITL(1) = 'actual pullin solution'
PLTITL = 'tbs "gb "'
CALL GRINIT(5,6,'for tbs gt .95')
C END PULLIN'S EXACT SOLUTION

C NOW THE ABOVE CIRCULATION DISTRIBUTION IS APPROXIMATED
C BY AN EQUATION AND OVERWRITTEN TO INSURE SMOOTHNESS IN THE
C CIRCULATION DISTRIBUTION.

deltbs = .919
num2 = 8
  do 785 i = 1,num2
    tbs(i) = deltbs + .01
    gb(i) = .022-2.95*(tbs(i)-.929)**2
    xplt(i) = tbs(i)
    yplt(i) = gb(i)
    deltbs = tbs(i)
  785 continue
NPER(1) = num2
TITL(1) = 'approx soltn'
PLTITL = 'tbs "gb "'
CALL GRINIT(5,6,'for tbs gt .95')
CALL GRKLIN(ILIN,ISYM,NPER,TITL,1,TBS,gb,PLTITL,599)

C This is the part of the circulation distribution < .93, which
C was taken from plots referenced in thesis. Pullin's solution
C is not applied to this.
num99 = 7
  tempbs(1) = .1
  tempgb(1) = .0
  tempbs(2) = .2
  tempgb(2) = .0072
  tempbs(3) = .3
  tempgb(3) = .01
  tempbs(4) = .42
  tempgb(4) = .012
  tempbs(5) = .7
  tempgb(5) = .0152
  tempbs(6) = .84
  tempgb(6) = .018
  tempbs(7) = .915
  tempgb(7) = .0216

CALL SPLIND(TEMPGB,TGBTBS,TEMPBS,NUM99,999.,999.)

120
ainc = .01
rad = -ainc
disend = .92
NDIS = disend/aINC + .01
nend = ndis + 1
ind = nend

DO 6 I = 1,Nend

    rad = rad + ainc
tbs(i) = rad
tbs(i) = 0.
tbs(nend) = disend
    gb(i) = seval(tbs(i),tempgb,tgbtbs,tempbs,num99)

    if(i.le.num2) then
        ind = ind + 1
        tbs(ind) = xplt(i)
        gb(ind) = yplt(i)
    endif

   6   continue

ndis1 = npts

npts = nend + num2
CALL SPLIND(GB,GBB,TBS,NDIS,999.,999.)
NPER(1) = Npts
TITL(1) = 'GAMMA DIS'
PLTITL = 'S GAMMA ',
CALL GRINIT(5,6,'CIRCULATION DISTRIBUTION VS. SHEET LENGTH')
CALL GRKLIN(ILIN,ISYM,NPER,TITL,1,TBS,GB,PLTITL,599)

C**************************************************************************************

return
end ! pullin
c   stop
c   end