SCHOOL OF ENGINEERING

EG-353 Level 3 Research Project

SESSION 2008/09

Name: Christopher Walton (368404)

Project Title: The 3D Aerodynamics of Lifting Surfaces Using Vortex Lattice Methods

Supervisor: Professor K. Morgan

Research Centre: C2EC

Discipline: Aerospace Engineering

Submission date: 27th April 2009

The 3D Aerodynamics of Lifting Surfaces Using Vortex Lattice Methods

Christopher Walton (368404)
School of Engineering – Swansea University

1. Introduction

For first order aircraft design it is advantageous to have a rapid means of generating good approximations for key parameters such as lift, drag and moments generated.
The Vortex Lattice Method (VLM) offers the means to determine these values in a computationally straightforward manner by use of vortex filaments in an inviscid flow model. Following a period of research of the theoretical method, a MATLAB program was designed, developed and tested as an implementation of the VLM.

2. Scope and Aims

Following the initial study a clearly defined a set criteria for the computational implementation of the VLM program were identified. These were broadly defined as follows;

-A program which can handle complex lifting surface geometries with a user-friendly interface.

-Accessible with average levels of desktop computational power.

-Output a range of useful aerodynamic data, graphs and plots.

-Suitable as a first order design tool for a task such as Unmanned Aerial Vehicle Development.

-Could also find use with university undergraduate students both in terms of understanding VLMs and also as a design tool.

3. Numerical Method

The VLM serves as an extension of Prandtl's Lifting Line Theory [1], with a wing represented by a system of vortex elements, rather than a single horseshoe vortex.

Given that the project is limited to looking at low-speed aircraft design, it is possible to consider that the system being modelled has a high Reynolds number. Given this assumption it can be found [2] that it is possible to disregard viscous effects close to the surface of the solid boundary (i.e. a wing) and model only the far-field flow. By discounting viscous effects the model only gives a result for the pressure distribution around the solid surface.
For attached flow around a wing, typically found at small angles of attack, pressure distribution is the primary source of forces experienced.

By considering only low-speed (no compressibility effects), inviscid flow at small angles of attack to the solid boundary, it is possible to model the velocity potential field as a solution to the Laplace Equation;

$$\frac{\partial^2\varphi}{\partial x^2}+\frac{\partial^2\varphi}{\partial y^2}+\frac{\partial^2\varphi}{\partial z^2}=0 \qquad \textbf{(1)}$$

Fundamentally the boundary conditions used to solve the system simply state that there can be no flow penetrating the solid surface of the wing.

$$\underline{V}\cdot\underline{\boldsymbol{n}}=0 \qquad (2)$$

Where; V – Total Velocity
n – Normal Vector to Surface

With further derivation [3], and taking advantage of assumptions such as the flow being irrotational, the final boundary condition is derived;

$$\frac{v}{V_\infty}=\frac{df}{dx}-\alpha \text{ on } y=f(x) \qquad (3)$$

Where; v x-component of local disturbance
V_∞ Freestream Velocity
f(x) Surface coordinate function
α Freestream Angle of Attack

Using this result the pressure and velocity at the surface can be related, finding that for the pressure coefficient (C_P),

$$C_p=-2\frac{u}{V_\infty} \qquad (4)$$

Through extensive derivation, combining the results in (3) and (4), it is found that the lift generated by a wing can be broken down through superposition as contributions due to the camberline, thickness and equivalent forces generated by a flat plate. Considering that lift is a result of the difference in pressure between the upper and lower surfaces, it can be demonstrated [3] that where linear boundary conditions exist (4), such as thin aerofoils, thickness effects can be ignored and the load on a wing can be expressed as,

$$\Delta C_p=2\left(C_{P_{Camber}}+C_{P_{Angle\,of\,Attack}}\right) \qquad (5)$$

Through superposition principles, it is possible to make a valid model of a wing by imposing cambered surface boundary conditions, such as in (3), on to an equivalent flat plate.

As with Thin Airfoil Theory, the flat plate can be modelled using horseshoe vortices. What sets the VLM apart is that it uses a lattice of horseshoe vortices, which allows for the modelling of more complex geometries.

The horseshoe vortex is composed of two semi-infinite vortex filaments bounding a finite-length filament, each with the same

vortex strength. This arrangement complies with the Kelvin & Helmholtz conditions.

With suitable rearrangement of the Biot-Savart Law, the induced velocity generated by each element of a horseshoe vortex can be assessed at a control point.

$$V_P = \frac{\Gamma}{4\pi} \frac{\underline{r_1} \times \underline{r_2}}{\left|\underline{r_1} \times \underline{r_2}\right|^2} \left[r_0 \left(\frac{\underline{r_1}}{\left|\underline{r_1}\right|} - \frac{\underline{r_2}}{\left|\underline{r_2}\right|} \right) \right] \qquad (6)$$

Where r1 and r2 are vectors between the ends of each vortex filament and the control point.

By discretising the camberline-surface and placing a horseshoe vortex and control point on each panel, it is possible to construct a system of linear equations which satisfy the boundary conditions;

$$\begin{bmatrix} C_{m=1,n=1} & \cdots & C_{m=1,n=N} \\ \vdots & \ddots & \vdots \\ C_{m=N,n=1} & \cdots & C_{m=N,n=N} \end{bmatrix} \begin{bmatrix} \Gamma_1 \\ \vdots \\ \Gamma_N \end{bmatrix} = - \begin{bmatrix} V_1 \cdot \boldsymbol{n} \\ \vdots \\ V_N \cdot \boldsymbol{n} \end{bmatrix} \qquad (7)$$

$Coefficient\ Matrix \quad Vortex\ Strength \quad Boundary\ Conditions$

Where the coefficient matrix is the velocity induced on a control point at panel (n) due to a unit strength horseshoe vortex source on panel (m).
The vortex strengths can be solved for using standard methods such as Gaussian Elimination.

4. Computational Implementation

The task within MATLAB can be broken down into three distinct steps which feed into the next; the Pre-Processor, Processor and Post-Processor.

The pre-processor reads the user input file and constructs a geometry based on the arbitrary input. Typically a wing is defined by aspects such as root and tip chord lengths, span, sweep, dihedral, twist and root and tip aerofoil profiles. The nodal points of the wing are then stored in a Panel Matrix.

The Processor places a horseshoe vortex and control point on each of the generated panels. It generates the Influence Coefficient Matrix by scanning through each panel, determining the velocity induced on the control point due to all sources in turn.

It then scans through every panel again, determining the boundary conditions due to the freestream velocity.
With both matrices constructed, it is possible to solve for the Vortex Strengths.

The post-processor converts the vortex strengths into engineering units, such as lift, drag, and by further analysis, calculates the aerodynamic centre of a configuration and the pitching moment coefficient.

5. Validation

Extensive use was made of a VLM based program (Tornado) which had itself been extensively validated as part of an MSc research project [4].
The program developed during the study, SwanVLM, was found to return results which closely match those given by Tornado, with very minor deviations due to differences in input geometries.
It was also found that the problem was accessible with desktop levels of computing power, with complex geometries solved in less than 10 minutes. Simple wing geometries can be solved in as little as 30 seconds.
SwanVLM also returns a range of tabulated data, graphs and animations based on the vortex strengths.

6. Conclusions

Extensive research showed that the Vortex Lattice Method has reached maturity and that little further work can be done to refine the method.
The bulk of the original work in this project has been in developing SwanVLM, which offers a very user-friendly and fast route to determining useful aerodynamic data of a model.
SwanVLM was found to return valid data for all tested wing configurations, which included sweep, dihedral, taper and twist.
This project would provide a sound basis on which to further develop computational aerodynamic techniques.

8. References

[1]. **Prandtl, L.** *Applications of modern hydrodynamics to aeronautics.* s.l. : NASA, 1923. NACA-TR-116.
[2]. **Katz, Joseph and Plotkin, Allen.** *Low-Speed Aerodynamics.* s.l. : Cambridge University Press, 2001. 978-0-521-66552-0.
[3]. **Mason, WH.** *Class notes for AOE 4114, Applied Computational Aerodynamics*. Dept of Aerospace and Ocean Engineering, Virginia Polytechnic Institute and State University. s.l. : Author - Internet, 1998. http://www.aoe.vt.edu/~mason/Mason_f/CAtxtTop.html.
[4]. **Melin, Tomas.** *A Vortex Lattice MATALAB Implementation for Linear Aerodynamic Wing Applications.* Royal Institute of Technology (KTH). s.l. : Author, 2000. MSc Thesis (http://www.redhammer.se/tornado/tornado_files/thesis.pdf).

The 3D Aerodynamics of Lifting Surfaces Using Vortex Lattice Methods

by

Christopher Walton

A dissertation

presented to Swansea University

in fulfilment of the

requirements for the degree of

Aerospace Engineering (BEng)

in

Engineering

School of Engineering, Swansea University, 2009

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this research dissertation. This is a true copy of the dissertation, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Signed:_____

Date: _____

Abstract

This thesis presents the results of a study in to the Vortex Lattice Method and methods of implementing it in MATLAB (a computational mathematics program) for arbitrary wing configurations.

For first order aircraft design it is advantageous to have a rapid means of generating good approximations for key parameters like lift, drag and moments generated. The Vortex Lattice Method offers the means to determine these values in a computationally straightforward manner by use of vortex filaments in an inviscid flow model.

Given a user inputted lifting surface configuration, a camberline-surface representation is discretised, on which a series of control points and source vortex singularities are placed. The vortex strengths are then determined such that the boundary conditions (no normal flow through the solid boundary) are satisfied at the control points.

The MATLAB program ‘SwanVLM’ was developed to implement this technique. Utilising a regularly distributed discretisation scheme and horseshoe vortex elements, it is able to accept lifting surface configurations through a Microsoft Excel spreadsheet, calculate vortex strengths and post-process the results into engineering units and other similarly relevant results.

An extensive validation was performed, comparing results from SwanVLM to those from a similar VLM based program, Tornado. The validation cases showed that SwanVLM was able to output good data for all standard wing configurations.

Acknowledgements

I would like to thank my research project tutor, Professor Ken Morgan, for giving me the freedom to pursue the project at my own pace, letting me make the odd mistake and for giving an occasional push in the right direction.

I'd like to thank the second marker for their time and consideration in assessing this project.

Thanks to all those who made the whole experience at Swansea so enjoyable.

"No one can realize how substantial the air is, until he feels its supporting power beneath him. It inspires confidence at once."

— Otto Lilienthal

Table of Contents

List of Figures

Figures: 3-1, 3-2, 3-3, 3-5, 3-6, 4-2, 4-3, 4-4 and 4-6

Reproduced from Fundamentals of Aerodynamics by Anderson [1]

Figures: 3-4 and 4-8

Reproduced from Low-Speed Aerodynamics by Katx & Plotkin [2]

Figures: 4-1, 4-5, 4-7 and 4-9

Reproduced from AOE 4114 Class Notes, Mason [3]

Figure: 5-4

Reproduced from MSc Report by Melin [4]

List of Tables

Chapter 1

Introduction

1.1 Scope and Goals

For any task requiring the design of an aerodynamic surface it is advantageous to have the ability to predict how this surface will behave in a fluid. Historically there have been a great many approaches to this task, with the majority of modern developments in this field being made possible by access to computers, and their ability to carry out large numerical simulations.

The purpose of the research project on which this thesis is based was primarily to study the fundamental theory of one such numerical technique, the Vortex Lattice Method (VLM). Furthermore it set out to investigate implementing the method in a useful manner using tools such as MATLAB (a high level numerical computation programming language).

As the initial study progressed it became possible to more clearly identify a set of desirable outcomes for the computational implementation of a VLM program. These were broadly defined as follows;

-A program which can handle complex lifting surface geometries with a user-friendly interface.

-Accessible with average levels of desktop computational power.

-Output a range of useful aerodynamic data, graphs and plots.

-Suitable as a first order design tool for a task such as Unmanned Aerial Vehicle Development.

-Could also find use with university undergraduate students both in terms of understanding VLMs and also as a design tool.

1.2 Background

In NASA Technical Report, SP-405[5], John DeYoung provides an excellent précis on the development of the VLM.

VLMs are firmly rooted as an extension to Prandtl's Lifting Line Theory[6], which set about modelling the circulation around a thin aerofoil using a system of vortex filaments conforming to the Biot-Savart Law and Helmholtz's Theorems.

The term, Vortex Lattice Method, was first introduced by V.M. Faulkner in 1946 [7], during which time the Lifting Line Theory was being extended as a numerical technique. The primary concept was to treat lifting surfaces as a lattice composed of many horseshoe vortices, thus allowing for the modelling of more complicated geometries in a potential flow.

It should be noted that as the technique treats surfaces in a potential flow, the method provides a simulation of a surface in incompressible, inviscous, irrotational flow. Additionally thickness effects are ignored, and only small angles of attack are treated. The net effect of these assumptions is that the method provides a good approximation to forces experienced in the linear region of the Lift vs. Angle of Attack plot, a point expanded upon later in the theory section.

Overall the method lends itself well to computational numerical techniques, which is evidenced by the observation that VLMs weren't significantly treated and expanded upon until the 1970s and onwards.

Chapter 2

Literature Review

As previously alluded to in Section 1.2, Vortex Lattice Methods saw two main periods of academic attention.

The first period can be traced to work commenced by the likes of Faulkner[7] and other such mathematicians, who studied extensions of Prandtl's Lifting Line approach. At this stage the VLM reached relative maturity as a theoretical approach, and is well documented by many of the collated papers contained within NASA's Technical Report, SP-405[5]. However at this stage it was not possible to readily address the large scale numerical computations required to solve the system of equations which renders the results VLMs offer.

From the late 1970's burgeoning computational advances permitted sufficient numerical power to make VLMs viable. It was at this stage where efforts were made to translate the VLM from the theoretical mathematical treatment of earlier years to an approach capable of being handled by computer systems.

These efforts came to a publically accessible head through a workshop funded by NASA in the late 70's. The workshop produced what could be considered a baseline reference computer code, its final version designated VLM4.997[3].

From this point onwards, many authors of academic aeronautical texts covered both the earlier developed fundamentals of VLMs, and latterly the computational implementations and modifications thereof.

Anderson's, 'Fundamentals of Aerodynamics' [1], serves as an excellent entry into understanding the method itself. By the author's own admission, the relevant chapter only offers a flavour of what the VLM is about, however the book features comprehensive guides to the Biot-Savart Law and Helmholtz's Theorems, and thus the composition of the Horseshoe Vortex which underpins the technique. Moreover the book explains the salient points of modern computational methods concerning Computational Fluid Dynamics (CFD), such as meshing, pre and post processing and so forth.

Leading on from this discourse on VLMs is Katz & Plotkin's, 'Low Speed Aerodynamics' [2]. Their book also gives excellent coverage to the theoretical aspects, but additionally extends to a detailed guide on how to implement a VLM for simple wings in basic programming language terms (the book itself features a FORTRAN interpretation of the method).

While the example program is simple in its nature and leaves the more difficult business of meshing to the reader, it provides an excellent starting point from which to work. It also crucially provides a basic numerical technique for assessing the velocity induced by a horseshoe vortex system (via the Biot-Savart Law). This effective method is applicable to even the more advanced implementations.

A useful accompaniment to Katz & Plotkin's book is W.H. Mason's class notes[3] for a course he runs at Virginia Tech. The notes feature much the same content as the other sources; however it expands on many of the points much to the benefit of the reader. Of particular relevance to this research project was Mason's coverage of how to treat cambered wings within the VLM.

I concluded my literature review by looking at two MSc theses which were directly relevant to my work.

The first of which was Tomas Melin's, 'A Vortex Lattice MATALAB Implementation for Linear Aerodynamic Wing Applications' [4]. In many ways this thesis closely mirrored the aims and goals of my own work, however on closer examination it became clear that our routes, and to some extent desired outcomes, differed in many ways. Therefore much of the thesis didn't factor in the work; however his MATALB implementation was used to validate the results of SwanVLM. The usage of his program, Tornado, is further documented in the validation section.

The other MSc thesis reviewed was Sofia Werner's, 'Application of the Vortex Lattice Method to Yacht Sails' [8]. The thesis presented a rich discussion of various Vortex Lattice Methods, and a more mathematical approach to appreciating the nuances of each method. Of particular note was a study into the merits of different mesh grid distributions (regular, half cosine, full cosine, etc) and quantising their effects on the accuracy of the results.

Chapter 3

The Aerodynamics of Lifting Surfaces

The fluid flows and forces which build up in a true 3-dimensional aerodynamic system can rapidly become very complex. However it is possible to quantify where the majority of these forces arise from, and thus break them down and treat them individually. It is further possible to impose appropriate simplifications of these effects and derive a system which can be readily modelled.

3.1 Sources of Forces and Moments

The only means by which a force can be transmitted to a body moving through a fluid is by the pressure distribution and shear stress distribution over the body surface. The shear force acts tangentially to the surface of the solid body, and the pressure distribution acts normal to the surface (Figure 3-1). By integrating these forces over the total surface, the total force experienced can be calculated.

The result of this integration is a resultant aerodynamic force, R, and a moment, M (Figure 3-2).

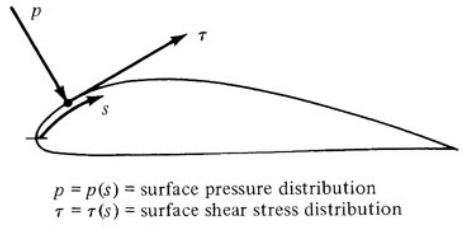


Figure 3-1: Surface Pressure and Surface Shear Vectors

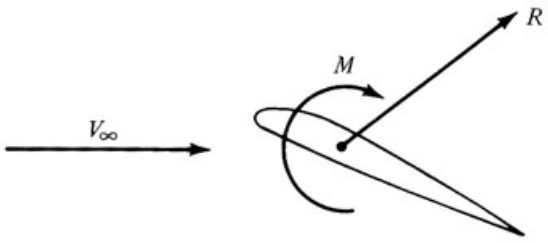


Figure 3-2: Aerodynamic Force and Moment

The aerodynamic force due to the shear and pressure distributions can then be decomposed into component vectors (Figure 3-3).

The flow far away from the body is denoted by V_∞, the freestream velocity.

The Angle of Attack, the angle at which the wing meets the freestream, is denoted by α and is defined as the angle between the freestream and the chord line of the wing (the chord line, c, being the line between the leading edge and trailing edge of a wing).

By definition lift (L) is considered as the component of R perpendicular to V_∞, and drag (D) is the component of R parallel to V_∞.

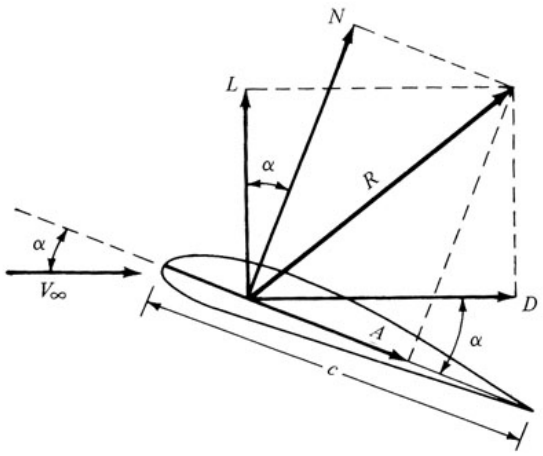


Figure 3-3: Components of Aerodynamic Force Vector

3.2 Simplification of Flow Behaviour

If we consider the wing to exist in a flow with a particular set of assumptions, it is possible to reduce the modelling problem and introduce a number of simplifications.

3.2.1 Viscous vs. Inviscid Flow

One such possible assumption regards the Reynolds number of the flow. Typically most General Aviation aircraft operate in a high Reynolds number flow ($Re>10^5$). As derived in Katz & Plotkin [2], the relative magnitudes of viscous terms in the momentum equation (in the x-direction) become very small and can be ignored in certain cases.

In a flow field around a solid boundary with high Reynolds number flow, it is found that there are two distinct regions characterised by having either strong viscous phenomena, and as such they can't be neglected, or regions where they're weak, and can be neglected.

The flow away from solid boundaries exhibits negligible viscous effects, which allows us to neglect them and treat the flow as inviscid.

Close to the solid boundary, shear effects between the flow and the boundary begin to develop and it is no longer valid to neglect viscous behaviour.

It should also be noted that for low-speed (Mach<0.3) flows, compressibility effects can be neglected, and the flow again treated as inviscid.

Therefore by considering the flow as inviscid only we have a system which provides good approximations for pressure distribution forces only with attached flows.

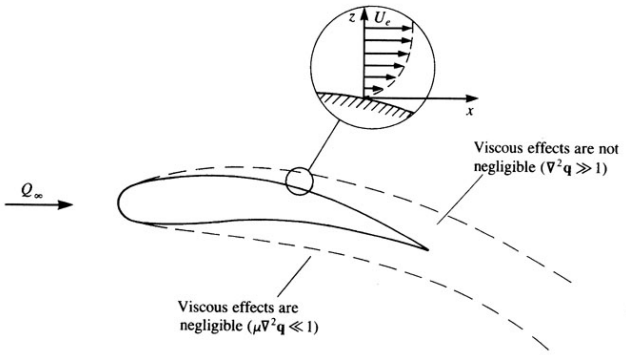


Figure 3-4: Flow regions in a high Reynolds number flow

3.2.2 Linear Region of Lifting Surface Behaviour

For an aerofoil placed in a flow, viscous effects remain negligible in the far field (from the boundary) as long as the flow remains attached to the aerofoil.

If the Angle of Attack (AoA) of an aerofoil is varied and the lift force measured, it is often possible to directly observe the point at which the flow separates (Figure 3-6). This point is denoted as the stalling AoA, and is characterised by having unattached, turbulent flow trailing from the upper surface of the wing (Figure 3-5). In this domain, it is no longer valid to consider the flow as inviscid.

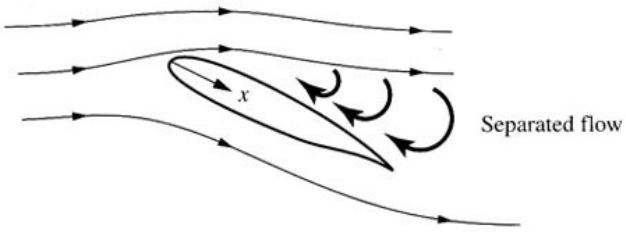


Figure 3-5: Illustration of Region of Separated Flow

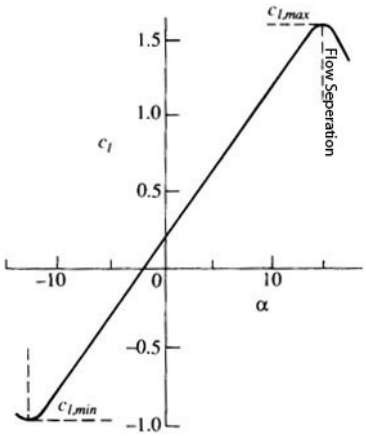


Figure 3-6: Lift vs. Angle of Attack for a Typical Aerofoil

3.2.3 Potential Flow

By only considering a model using the previously established assumptions of low-speed, inviscid far-field flow for small angles of attack relative to the solid boundary, it is possible to begin to apply mathematical approaches to the system.

As derived in Chapter 2 of Katz & Plotkin [2], it is possible to consider the flow velocity as a potential flow.

Potential flow is described with the velocity potential, which is a function of both time and space. The flow velocity, v, can be expressed as the gradient of a velocity potential, φ.

$$\underline{v} = \underline{\nabla \varphi} \tag{3.1}$$

Vector calculus defines the curl of a gradient as zero,

$$\underline{\nabla} \times \underline{\nabla}\varphi = 0 \tag{3.2}$$

Where ‘x’ denotes the Cross Product.

Therefore the curl of the vector field, v, can be found to be equal to zero, and thus the vorticity of the field is zero. This result is valid within our inviscid flow assumptions.

$$\underline{\nabla} \times \underline{v} = 0 \tag{3.3}$$

Furthermore, in incompressible flow, the velocity, v, has zero divergence,

$$\underline{\nabla} \cdot \underline{v} = 0 \tag{3.4}$$

Where the middle dot denotes the Dot Product.

The result being that the velocity potential satisfies Laplace’s Equation,

$$\frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2} + \frac{\partial^2 \varphi}{\partial z^2} = 0 \tag{3.5}$$

The Vortex Lattice Method is fundamentally an approach to solving the Laplace Equation, and it is this task which the numerical method focuses on.

3.3 Summary of Key Assumptions

The method we’re proceeding with makes the following key assumptions;

-The flow is assumed to be inviscid, irrotational and incompressible, therefore;

-The system will only model a wing experiencing attached ($\alpha<\alpha_{stall}$) low-speed, high Reynolds number flow, therefore;

-It is only suitable for predicting forces due to pressure distribution on the wing.

Chapter 4

Numerical Method

In summary, the Vortex Lattice Method is a solution of the Laplace Equation using vortex singularities, with boundary conditions drawn from the wing surface.

The following sections outline the formulation of the boundary conditions and the problem itself. Much of the derivation in this section is drawn from that performed in Mason's notes [3] with some intermediate steps omitted.

4.1 Boundary Conditions

4.1.1 Boundary Conditions on the Wing Surface

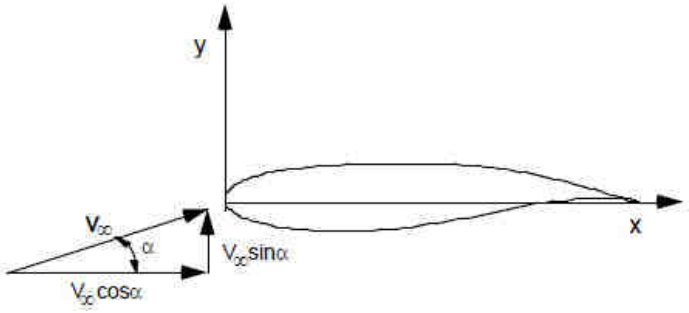


Figure 4-1: Aerofoil Geometry

Fundamentally the solid surface boundary condition can be satisfied by stating that the normal component of velocity to the wing surface is equal to zero,

$$\underline{V} \cdot \underline{\boldsymbol{n}} = 0 \tag{4.1}$$

On the wing surface line (considering the Angle of Attack only and using a 2D aerofoil geometry as specified in Figure 4-1),

$$F(x, y) = \ 0 = y - f(x) \tag{4.2}$$

Where,

$$\boldsymbol{n} = \frac{\underline{\nabla} F(x,y)}{\left|\underline{\nabla} F(x,y)\right|} \tag{4.3}$$

Furthermore we can state the total velocity, V, as,

$$\underline{V} = V_\infty + q(x,y) \tag{4.4}$$

Where q(x,y) is a disturbance velocity.

We have previously assumed that the flow is irrotational; therefore the velocity can be expressed in component form, u and v, as functions of the velocity potential,

$$\begin{aligned} u &= \frac{\partial \varphi}{\partial \mathrm{x}} \\ v &= \frac{\partial \varphi}{\partial \mathrm{y}} \end{aligned} \tag{4.5}$$

Expanding gives,

$$\begin{aligned} U_{Total} &= V_\infty \cos\alpha + u(x,y) \\ V_{Total} &= V_\infty \sin\alpha + v(x,y) \end{aligned} \tag{4.6}$$

With appropriate substitution and recalling the relationship in (4.2),

$$v = (V_\infty \cos\alpha + u)\frac{df}{dx} - V_\infty \sin\alpha \tag{4.7}$$

Using a small angle approximation and division by V_∞,

$$\frac{v}{V_\infty} = \left(1 + \frac{u}{V_\infty}\right)\frac{df}{dx} - \alpha \tag{4.8}$$

And by regarding the u component of the disturbance compared to the freestream as inferior to unity, we derive the boundary condition,

$$\frac{v}{V_\infty} = \frac{df}{dx} - \alpha \text{ on } y = f(x) \tag{4.9}$$

4.1.2 Thin Aerofoil Theory Pressure Relation

At this stage it would also be useful to draw in our assumptions and prior derivation to form a relationship between the pressure and the velocity of the flow.

The exact relationship between the Coefficient of Pressure and velocity in incompressible flow is given by,

$$C_p = 1 - \left(\frac{v}{V_\infty}\right)^2 \tag{4.10}$$

Substituting the combined components of V_∞ and the disturbance velocity following a small angle substitution (4.6), we find,

$$v^2 = V_\infty^2 + 2V_\infty u + u^2 + (V_\infty \alpha)^2 + 2V_\infty \alpha v + v^2 \tag{4.11}$$

If we expand the terms, divide by V_∞ and substitute into (4.10),

$$C_p = 1 - 1 - 2\frac{u}{V_\infty} - \frac{u^2}{V_\infty^2} - \alpha^2 - 2\alpha\frac{v}{V_\infty} - \frac{v^2}{V_\infty^2} \tag{4.12}$$

Furthermore, if we assume that α and the v-components of the disturbance velocity ratio are much smaller than 1, we can neglect some terms and find,

$$C_p = -2\frac{u}{V_\infty} \tag{4.13}$$

Equation (4.13) shows us that for small disturbances, the pressure is a linear function of velocity, and that it is suitable to use superposition to summate different sources of lift.

4.1.3 Decomposition of the Aerofoil

The load on the aerofoil overall can be expressed as,

$$\Delta C_p = C_{P_{Lower}} - C_{P_{Upper}} \tag{4.14}$$

As expanded upon in Chapter 6 of Mason's Notes [3], it is possible to break down the contributions to lift of a wing as a superposition of lift effects from the thickness, the camber line, and an equivalent flat plate.

Combining these contributions and substituting them into (4.14), it is found that,

$$\Delta C_p = 2\left(C_{P_{Camber}} + C_{P_{Angle\ of\ Attack}}\right) \tag{4.15}$$

The result shows that for conditions where the linear boundary conditions are valid, lift due to thickness is not a first order effect and can be discounted.

Additionally it is also possible to impose the camber line boundary conditions on to the representative flat plate, and derive an estimation of the load.

4.2 Vortex Singularities

As covered in Anderson's, 'Fundamentals of Aerodynamics' [1], it is possible to model flow within our simplified system using four elemental flows. These are uniform, source, doublet and vortex flow.

Of particular interest to this study is vortex flow. It is possible to use vortex flow sources to model lifting surfaces and derive results relevant for our engineering problem. This section serves largely as an overview of the concept and omits most of the detailed derivations and proofs recounted in Anderson's book.

In brief, if we consider a flow where we have no radial velocity about a point, and only regular tangential velocity, we derive vortex flow (Figure 4-2).

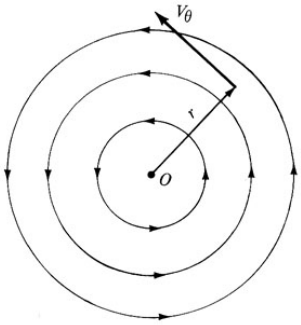


Figure 4-2: Point Vortex Flow

It is possible to quantify the tangential velocity of this flow as,

$$V_\theta = -\frac{\Gamma}{2\pi r} \tag{4.16}$$

Where; Γ – Vortex Strength

If we were to then take a line of these vortex point sources stretching into infinity, we would have a vortex filament, as detailed in Figure 4-3.

These vortex filaments in potential flow are described by a series of theorems and can be used to form the basis of a lifting surface in potential flow, for which a solution of the governing Laplace Equation can be found, and thus the potential velocities.

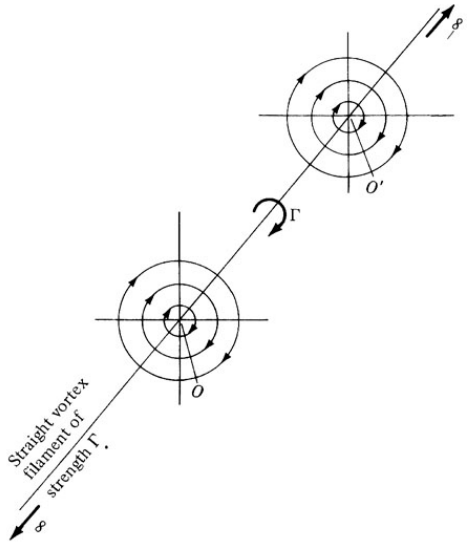


Figure 4-3: Vortex Filament

4.2.1 Vortex Theorems

The properties of vortex filaments were fundamentally defined by Kelvin and Helmholtz, and are as follows,

i.) Along a vortex line the circulation (Γ) is constant

ii.) A vortex filament cannot begin or end abruptly in a fluid. The vortex line must be closed, extend to infinity, or end at a solid boundary.

Furthermore the circulation about any section is the vortex strength.

iii.) An initially irrotational, inviscid fluid will remain irrotational.

A related result is that, (quoted from Mason [3]),

"A sheet of vortices can support a jump in tangential velocity [i.e. a force], while the normal velocity is continuous. This means you can use a vortex sheet to represent a lifting surface."

These properties allow us to formulate a representation of a wing using vortex filaments.

4.2.2 Biot-Savart Law

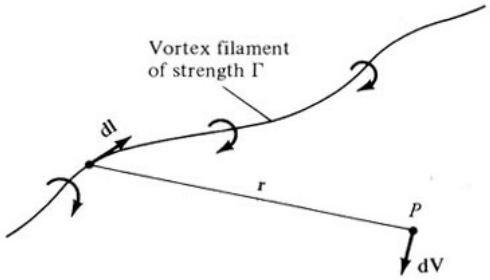


Figure 4-4: Vortex Filament Layout

The Biot-Savart Law is a fundamental relation for vortex flow in an irrotational, inviscid system. Using the nomenclature from Figure 4-4, it defines that,

$$\underline{dV} = \frac{\Gamma}{4\pi}\frac{\underline{dl} \times \underline{r}}{|\underline{r}|^3} \tag{4.17}$$

As per our Kelvin & Helmholtz relations, the vortex strength along a filament is constant, and we can therefore integrate over the length of filament to find the total induced velocity,

$$V_P = \frac{\Gamma}{4\pi}\int \frac{\underline{dl} \times \underline{r}}{|\underline{r}|^3} \tag{4.18}$$

Mason [3] derives that for a finite length vortex we find the result (based on the vortex detailed in Figure 4-5),

$$V_P = \frac{\Gamma}{4\pi h}(\cos\theta_1 - \cos\theta_2)\boldsymbol{e} \tag{4.19}$$

By further rearrangement (considering the cosine terms by their vector definitions, Figure 4-4), it is found that,

$$V_P = \frac{\Gamma}{4\pi} \frac{\underline{r_1} \times \underline{r_2}}{\left|\underline{r_1} \times \underline{r_2}\right|^2} \left[r_0 \left(\frac{\underline{r_1}}{\left|\underline{r_1}\right|} - \frac{\underline{r_2}}{\left|\underline{r_2}\right|} \right) \right] \tag{4.20}$$

Where 'x' denotes the Cross Product.

It is also worth noting that the vortex strength term is contained linearly in the induced velocity formula, and that it can be expressed as,

$$\underline{V_m} = C_{m,n}\Gamma_n \tag{4.21}$$

Where V_m is the velocity induced at a point m, due to a source at point n and C is the influence coefficient.

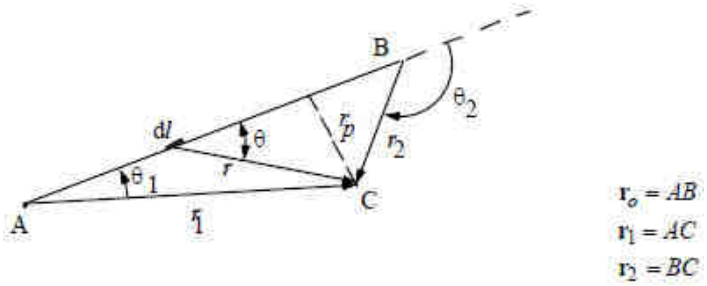


Figure 4-5: Vortex Filament Vector Arrangement

4.2.3 Kutta–Joukowski Theorem

At this stage it is also important to consider the Kutta-Joukowski Theorem, which relates the circulation (Vortex Strength), and lift force experienced by a cylinder. The result of interest to this study is that for the line integral of a closed loop around a cylinder, we find that, in vector form,

$$\underline{F} = \rho\left(\underline{V} \times \underline{\Gamma}\right) \cdot l \tag{4.22}$$

Where; F - Force

ρ – Density

L – Length of Vortex Filament

X – Cross Product

Middle Dot – Dot Product

An interesting result of this formula is that due to the cross product term, no lift force is induced by a vortex filament orientated streamwise in a flow.

4.2.4 The Horseshoe Vortex

By using a system of vortex filaments we can replace a finite wing with a horseshoe vortex (Figure 4-6), as detailed by Prandtl's Lifting-Line Theory.

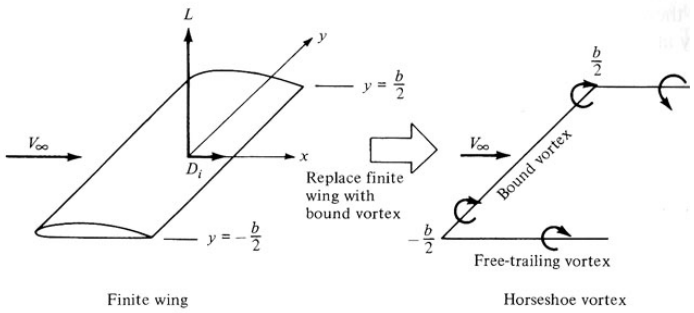


Figure 4-6: Replacing a Wing with a Horseshoe Vortex

The horseshoe vortex is composed of two semi-infinite vortex filaments bounding a finite-length filament, each with the same vortex strength. This arrangement complies with the Kelvin & Helmholtz conditions.

Using equation (4.20) on each segment of the horseshoe and summing the results renders the velocity induced by the horseshoe as a whole,

$$V = V_{A\infty} + V_{AB} + V_{B\infty} \tag{4.23}$$

(Vortex Elements detailed as per Figure 4-7)

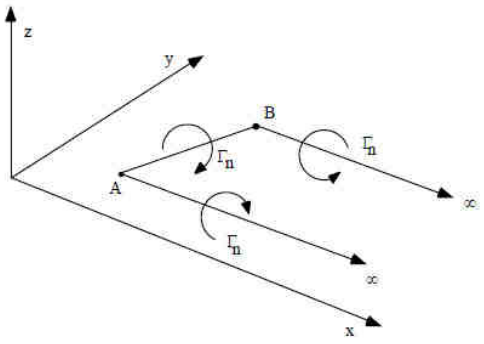


Figure 4-7: Horseshoe Vortex Arrangement

4.3 The Vortex Lattice Method

The Vortex Lattice Method serves as an extension to lifting line theory in that it treats the lifting surface as a system of many horseshoe vortices, as detailed in Figure 4-8. This allows the treatment of more complicated geometries than the lifting line approach offers.

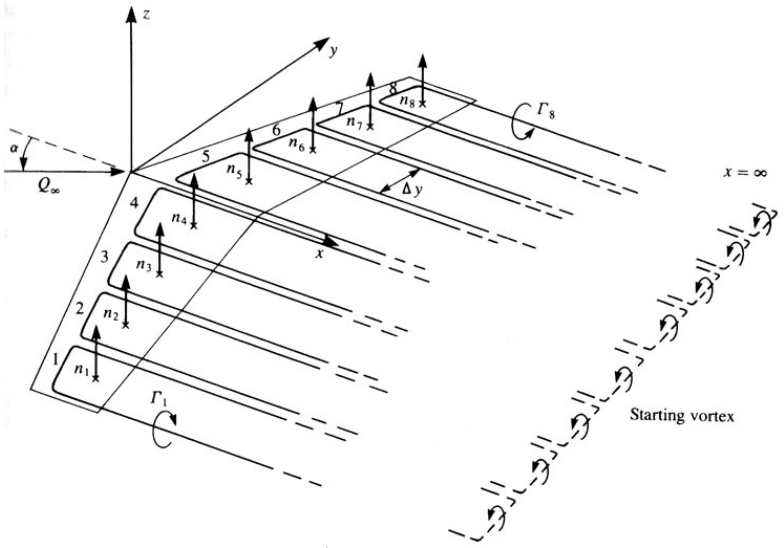


Figure 4-8: Vortex Lattice Wing

4.3.1 Schema

The method can be defined by a series of key steps, as follows;

i.) Discretise the planform into regular quadrilateral shapes.

ii.) Place a horseshoe vortex on each panel.

The bound section of the horseshoe is placed on the quarter panel point for this study. (Section 4.3.2 elaborates this choice)

iii.) Place a control point on the panel, where the boundary condition is to be satisfied.

This study places the control point at the three quarter panel chord point (Section 4.3.2).

iv.) Assume a flat wake which satisfies the Kutta Condition (Detailed in 4.3.3).

v.) Construct a linear system of equations to calculate the vortex strength required by each panel to satisfy the boundary condition.

This approach is illustrated by Figure 4-9.

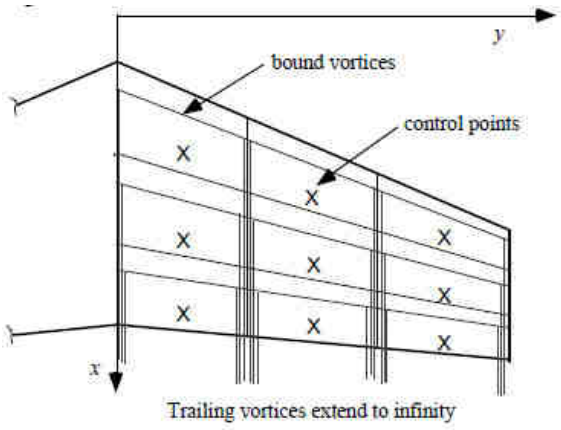


Figure 4-9: Vortex Lattice Scheme

4.3.2 Vortex Placement

Mason [3] carries out a detailed derivation of the optimal point to locate the bound vortex section and the control point for a horseshoe vortex.

By considering a 2-dimensional flate plate with a point source, and relating the flow tangency condition, induced velocity at the control point and the Kutta-Joukowsky Theorem, it is found that,

$$r = \frac{c}{2} \tag{4.24}$$

Where r is the distance between the point source and the control point, and c is the chord of the aerofoil.

By taking a similar approach with a parabolic aerofoil camber model, it is found that,

$$\frac{a}{c} = \frac{1}{4} \tag{4.25}$$

Where a is the distance between the vortex point and the leading edge.

We therefore see that placing the bound vortex at the quarter chord point, and the control point at the three quarter chord point, these formulae are satisfied.

Mason states that this approach is not necessarily the most mathematically precise, but works well enough to be used as a ‘rule of thumb’.

4.3.3 Kutta Condition

We also need to pay particular attention to the geometry of the wake and ensure that the trailing vorticies leave the wing in a proper manner. This problem was addressed by the German mathematician, W.H. Kutta.

The Kutta Condition states that;

"The flow leaves the sharp trailing edge of an aerofoil smoothly and the velocity there is finite."

For our infinitely thin vortex sheet camberline representation, it is sufficient to approximately satisfy the condition by orientating the wake such that it leaves the trailing edge planar to the flat reference surface, and only model small Angles of Attack.

4.3.4 Solving the System

If we recall Equation (4.21), the velocity induced at the control point on panel m, due to a horseshoe vortex on panel n, is,

$$\underline{V_m} = C_{m,n}\Gamma_n \tag{4.26}$$

We find that the total induced velocity at m, due to N total source panels is,

$$\underline{V_m}_{Total} = \sum_{n=1}^{N} C_{n,m}\Gamma_n \tag{4.27}$$

If we include the freestream velocity and express the total velocity at panel m in its component vector form, we find the result,

$$\begin{aligned}\underline{V_m} = (V_\infty \cos\alpha \cos\beta + u_{ind})\boldsymbol{i} + (-V_\infty \sin\beta + v_m)\boldsymbol{j} \\ +(V_\infty \sin\alpha \cos\beta + w_m)\boldsymbol{k}\end{aligned} \tag{4.28}$$

Where; α – Freestream Angle of Attack

β – Freestream Sideslip Angle

The system can be solved by satisfying the no penetrating flow condition at the wing surface boundary for the total velocity. Recalling the boundary condition from Equation (4.1),

$$\underline{V} \cdot \boldsymbol{n} = 0 \tag{4.29}$$

If we then define the wing surface as a function of co-ordinate position,

$$F(x, y, z) = 0 \tag{4.30}$$

We can combine the two and form,

$$\underline{V} \cdot \frac{\underline{\nabla F}}{\left|\underline{\nabla F}\right|} = \underline{V} \cdot \underline{\nabla F} = 0 \tag{4.31}$$

Substituting this term into (4.28), and carrying through the calculation we find,

$$\sum_{n=1}^{N} \left(\frac{\partial F}{\partial x} C_{m,n_i} + \frac{\partial F}{\partial y} C_{m,n_j} + \frac{\partial F}{\partial z} C_{m,n_k}\right) \Gamma_n = -V_\infty \left[\cos\alpha \cos\beta \frac{\partial F}{\partial x} - \sin\beta \frac{\partial F}{\partial y} + \sin\alpha \cos\beta \frac{\partial F}{\partial z}\right] \tag{4.32}$$

$$for\ m = 1,2,3 \ldots N$$

Where; N – Total number of panels

m – Source Panel

n – Control Panel

This approach forms a linear system of equations which can be easily solved for the vortex strengths (Γ).

Essentially the system of equations generated state that the induced velocity at panel m due to N panels must be equal to the total velocity experienced by the panel.

Chapter 5

Computational Implementation

As detailed in Chapter 4, the numerical method for a vortex lattice representation of a wing can be mathematically demanding, however the end result essentially comes down to solving a system of linear equations. This task is very straightforward for a computer to handle.

The main difficulty in the task, as with many other CFD problems, is in the construction of an arbitrary surface, or mesh, on which the calculations can be applied.

This chapter presents the specifics of SwanVLM, the MATLAB code which was written to implement the Vortex Lattice Method.

5.1 Overview

SwanVLM comprises of functions which are used to complete three major tasks in the following order;

i.) Pre-Processor

Reading the user configuration file, constructing a camberline mesh representation, storing nodal co-ordinate points for later recall.

ii.) Processor

Constructs matrix of influence coefficients, matrix of boundary conditions, and solves for horseshoe vortex strengths.

iii.) Post-Processor

Converts vortex strengths into engineering units, calculates aerodynamic centre, graphs, charts, etc.

5.2 Pre-Processor

The pre-processor is where the main challenge of the overall task lays; the ability to accept an arbitrary wing configuration and construct it appropriately.

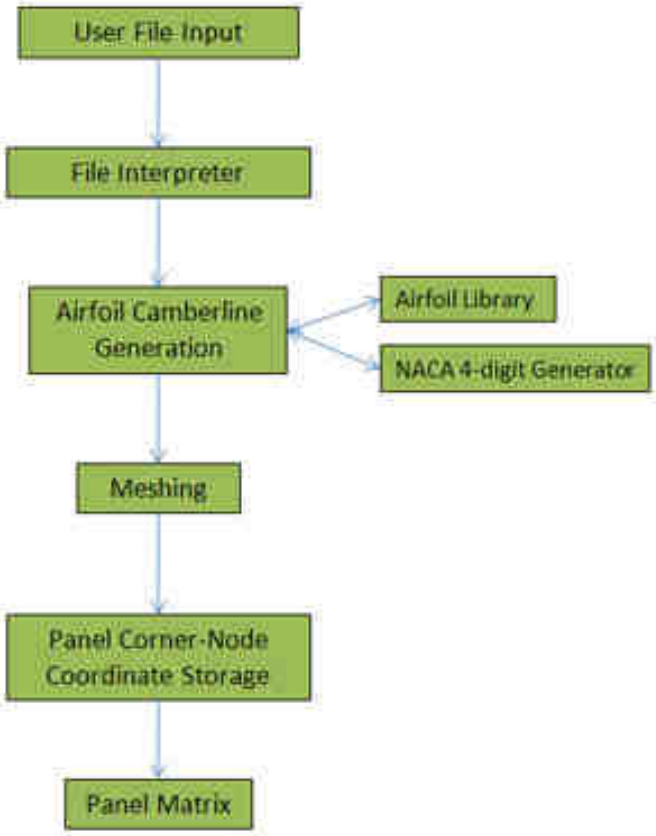


Figure 5-1: Pre-Processor Flow

5.2.1 Wing Definition and Input

One of the objectives of SwanVLM was that it should be user-friendly, so to that end the user is able to input the lifting surface(s) configuration via an Excel spreadsheet. An example of an input file and typical ouput mesh is given in Appendix A.

When initialised, the user inputs the file name, which is in turn read into MATLAB. The geometry is divided into a wing by wing, section by section basis, with the ability to define; the root and tip chord, root and tip profiles from an aerofoil library or NACA 4 digit number, span, leading edge sweep, dihedral, and root and tip incidence (allowing twist to be inputted). Additionally the desired number of chordwise and spanwise panels can be inputted.

The aerofoil library was sourced from the Aerospace Engineering Dept at the University of Illinois at Urbana-Champaign. It is a collection of over 1000 aerofoils, described in terms of x and y co-ordinates for the upper and lower surfaces of a unit chord length aerofoil. SwanVLM allows the user to input these files

as profile definitions, utilising a cubic spline fit to the upper and lower surfaces, with a camberline generated as the median distance between each surface.

It is also possible to input any of the NACA 4-digit range, with the necessary equations for coordinate plotting implemented in the program.

Additionally the desired number of chordwise and spanwise panels can be inputted, which can have a strong effect on the accuracy of the result. The issue of grid convergence is covered in the Validation section (6.1). The current version of SwanVLM has this value fixed at 5 by 60 panels; however it is relatively straightforward to open this option to the user.

5.2.2 Lofting, Coordinate Generation and Storage

The mathematical method by which the lifting surface coordinates are constructed is similar to the lofting approach.

The root camberline profile is placed in 3D space; the tip camberline profile is then placed with reference (sweep, dihedral, etc) to the root profile. The meshing function then generates equally spaced nodal points between corresponding root- and tip-profile nodes. Additional sections are added on by treating the preceding tip-profile as the root, and placing a new camberline tip-profile as before.

The Cartesian coordinates of these nodal points are then stored in a matrix on a panel-by-panel basis, as outlined in Figure 5-2. It should be noted that once the panel corner node coordinates are generated, it greatly simplifies later stages of the program if they can be recalled by reference to a panel number only.

Functions were also written at this stage to cater for recalling the generated geometry;

‘OrdRecall’ can be inputted with the panel number, and returns the corner node coordinates for that panel.

A more developed panel property inspection tool was also developed (‘PanelTool’), which when inputted with a panel number can return the panel’s angle with reference to the xy-plane (angle of attack) or zx-plane (sideslip angle), area, chord length and normal vector. All of these are later used for boundary condition generation and are must-have functions for geometry handling.

SwanVLM generates three meshes at the pre-processing stage for later calculations. A non-dimensionalised (with reference to the root chord) representative flat plate mesh is used for influence coefficient matrix generation, a non-dimensionalised cambered (i.e. 3D) mesh for boundary condition matrix generation, and a true representational mesh for graphical purposes at the post-processing stage.

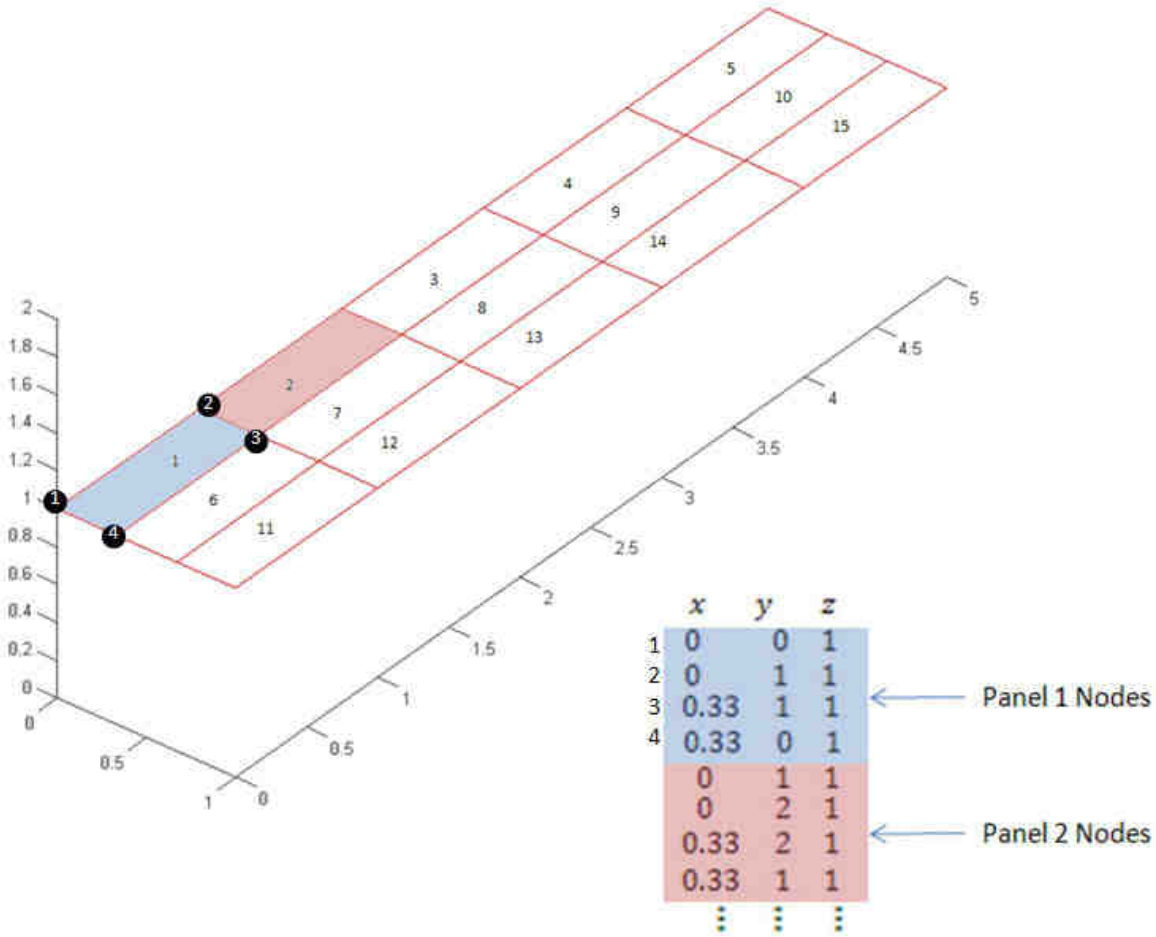


Figure 5-2: Planar Wing Mesh Example

5.3 Processor

The processor represents the core of the program; it is where the linear system of equations is generated and the vortex strength determined.

Using the nomenclature from Section 4.3.4, the problem can simply be defined as in Equation (5.1);

$$\begin{bmatrix} C_{m=1,n=1} & \cdots & C_{m=1,n=N} \\ \vdots & \ddots & \vdots \\ C_{m=N,n=1} & \cdots & C_{m=N,n=N} \end{bmatrix} \begin{bmatrix} \Gamma_1 \\ \vdots \\ \Gamma_N \end{bmatrix} = - \begin{bmatrix} V_1 \cdot \boldsymbol{n} \\ \vdots \\ V_N \cdot \boldsymbol{n} \end{bmatrix} \tag{5.1}$$

$$Coefficient\ Matrix \quad Vortex\ Strength \quad Boundary\ Conditions$$

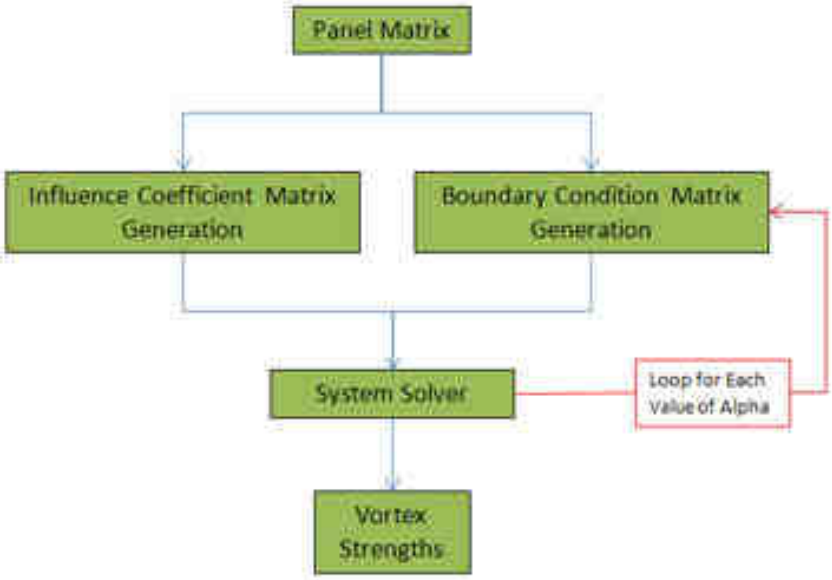


Figure 5-3: Processor Flow

5.3.1 Influence Coefficient Matrix Generation

The influence coefficient generation function employs a loop inside a loop; the outer loop scanning through the control points whilst the inner loop scans through the panel sources.

As per Figure 4-9, for a given source panel and control panel, the generation function within SwanVLM recalls the source panel's nodal coordinates and places a system of three vortex filaments on the panel. The horseshoe starts from 20 chord lengths behind the wing (a point considered suitable to satisfy the Biot-Savart Law starting vortex condition) up to and then across the panel's quarter chord line, then returns back

to a 20 chord length distance. SwanVLM maintains a planar (fixed) wake to the reference panel, which approximately satisfies the Kutta Condition (4.3.3) for small angles of attack.

The control point is placed centrally on the control panel's three-quarter chord line.

Using Equation (4.20) on each element of the horseshoe vortex, with the vortex strength set at unity, the induced velocity at the control point is determined. The downwash due to the trailing vortex elements only is also stored at this stage, as it allows for induced drag computation during post-processing.

5.3.2 Boundary Condition Matrix Generation

The key to SwanVLM being able to handle cambered surfaces is that it takes the boundary conditions from the cambered wing and imposes them on the flat reference mesh previously generated, as per the allowable superposition discussed in 4.1.1 and illustrated in Figure 5-4.

The boundary condition matrix generator uses a loop to scan through all the cambered wing mesh panels, determining the panel's orientation relative to the freestream and calculates the normal component of this relative to the cambered panel. The 'PanelTool' function (5.2.2) is used extensively to this end.

SwanVLM is designed to provide output over a range of wing attack angles. As seen in the derivation of the numerical problem (4.1), with a fixed wake only the boundary condition values change as the angle of attack is altered. SwanVLM generates a set of boundary conditions for each angle of attack, and solves the system using these conditions. By only generating boundary conditions for each angle, the computational time is greatly reduced.

Figure 5-4: Shifted Boundary Conditions

5.3.3 System Solution

With the influence coefficient and boundary condition matrices set up, the solution becomes very trivial within MATLAB using the backslash operator,

$$\{\Gamma\} = \{C\}\backslash\{V_{Boundary}\} \tag{5.2}$$

The influence coefficient matrix exhibits a strong diagonal trend (generally a source panel has the greatest induced velocity on itself), and is fully ranked. Typically though it is non-Hermitian, and is not positive-definite, therefore MATLAB performs Gaussian Elimination with Partial Pivoting.

5.4 Post-Processor

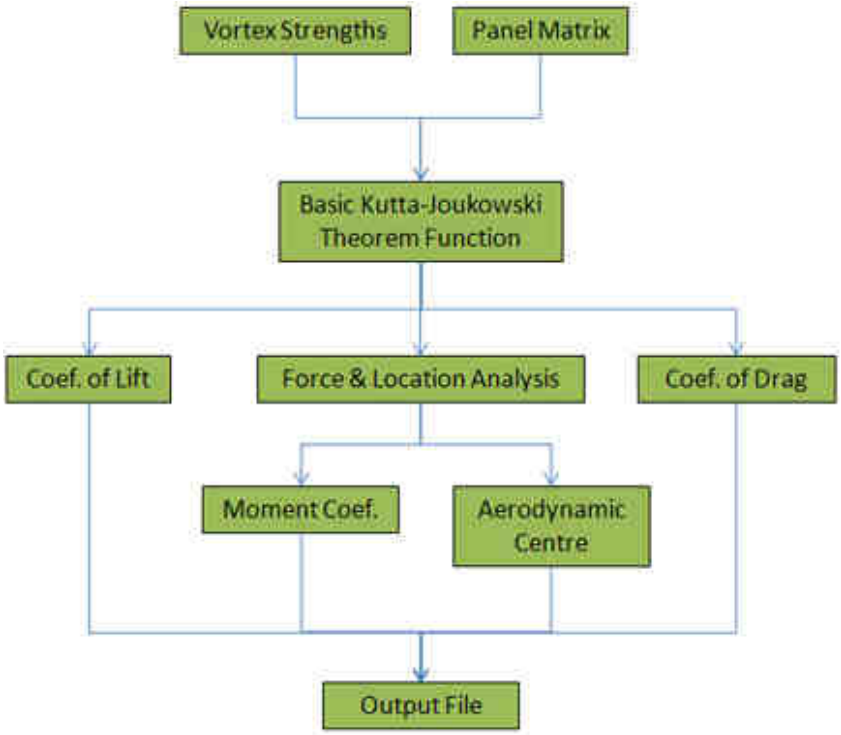


Figure 5-5: Post-Processor Flow

Using the Kutta-Joukowski Theorem (4.2.3), the values for vortex strength can be readily converted into a lift force. Furthermore these forces can be non-dimensionalised in the usual manner and the coefficients of lift and drag can be readily extracted. By performing a similar process to the induced downwash values collected during processing, the induced drag can also be quantified.

Additionally at this stage, the panel forces are tied up with their geometric locations and an analysis on the moments produced by the wing is carried out. From this it is possible to determine properties like the aerodynamic centre and stability derivatives.

SwanVLM also takes full advantage of MATLAB's capable visualisation tools. A good example of the sort of results available can be seen in Figure 5-6, which shows how the vortex strength can be worked back to calculate the ΔC_p over the wing surface and displayed as a 'surf' in MATLAB. It is also possible to produce a range of these surface figures for varying angles of attack, which SwanVLM can output as a GIF animation file.

The results of the alpha-sweep computations are stored back as an extra tab in the input Excel file. An example of a typical output is given in Appendix B.

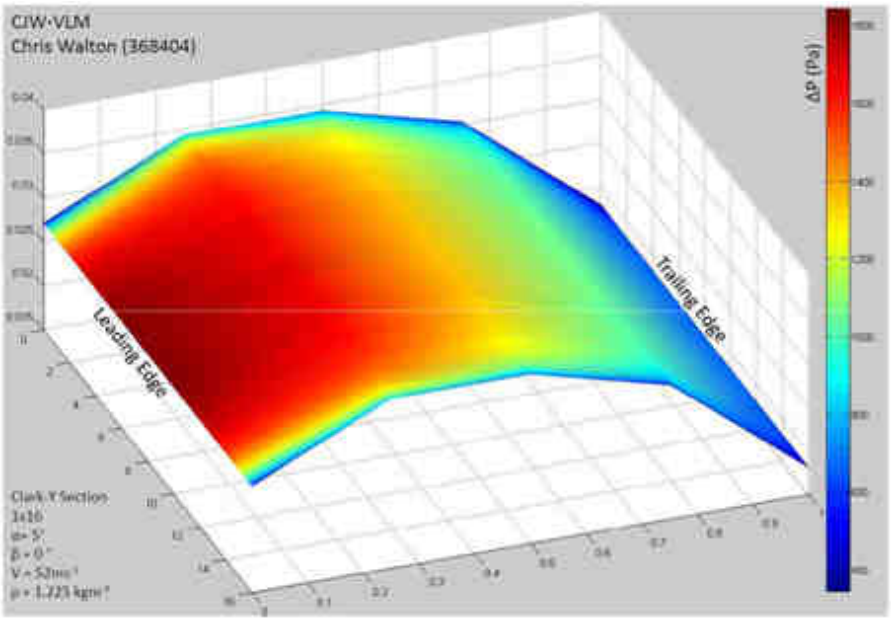


Figure 5-6: Example of SwanVLM's Graphical Output

5.5 Code Optimisation

One of the aims of SwanVLM was that it should be possible to run a problem in a reasonable time using desktop computing power. To this end, close attention was paid to ways in which the code could be optimised, both in terms of efficient programming, and catering for the best approaches to a problem within

MATALB. The bulk of computational effort is expended at the pre-processing, and processing stages of the program.

By using MATLAB's internal code profiler, M-LINT, it is possible to analyse the number of times a function is called, how much time is spent running that function and how it interacts with the rest of the program. This allows for a very rapid identification of any bottlenecks in the program.

5.5.1 Pre-Processor

A large amount of effort can be reduced in the meshing stage by taking advantage of any mirroring across a plane.

Typically most aircraft feature a symmetric wing configuration about the longitudinal axis. It is possible to generate co-ordinate points for one half of the aircraft only, and transpose these [x, y, z] coordinates as [x, -y, z].

As discussed in the previous section, it is also possible to greatly streamline the processing code if care is taking at the pre-processing stage to store everything with reference to panel numbering.

5.5.2 Processor

The single most computationally expensive function in SwanVLM is the Influence Coefficient Matrix generation during the Processor stage.

Given that the function calculates the velocity induced by all sources on each control panel, the process requires $O(N^2)$ operations, where N is the total number of panels. It can therefore be seen that even for a simple single wing with a single section, SwanVLM has to perform the vortex influence calculation 90,000 times (assuming use of the 5x60 panelling option).

There are a number of ways that this can be improved upon in a general programming sense, and specifically in MATLAB.

Great care must be taken to ensure that within the innermost scanning loop of the function, only the bare essentials are calculated. It can greatly improve the program's efficiency if any redundant lines of code are removed.

Another major bottleneck identified using M-LINT was the use of MATLAB's built in dot and cross product functionality. Both are absolutely essential to the numerical method, however close examination of MATLAB's internal functions showed that they included a number of lines of code to ensure that the vector input had the correct scaling and form. If care is taken to ensure a consistent vector input from upstream in the program, it is possible to use a more straightforward linear form to calculate the cross or dot product, and greatly reduce the number of computational cycles required.

Chapter 6

Results (Validation)

This chapter present the results of the validation runs performed with SwanVLM. It comprises of two distinct studies, the first was on mesh grid solution convergence, and the other was concerned with validating the result from the program against reference data.

6.1 Grid Convergence

As with any numerical solution, SwanVLM's results are highly dependent on how fine the discretisation scheme is when modelling the problem. The purpose of this area of study was to determine the optimal panelling options required (in terms of chordwise and spanwise panels per section) to achieve a suitable result.

Katz & Plotkin [2] reproduce a graph (Pg 348, Fig 12.16) from Jones and Cohen's, "High Speed Wing Theory", which details the expected $C_{L\alpha}$ against Aspect Ratio for an untapered planar wing. Using a fixed Aspect Ratio of 3.33 and evenly spaced panelling, the following results were obtained;

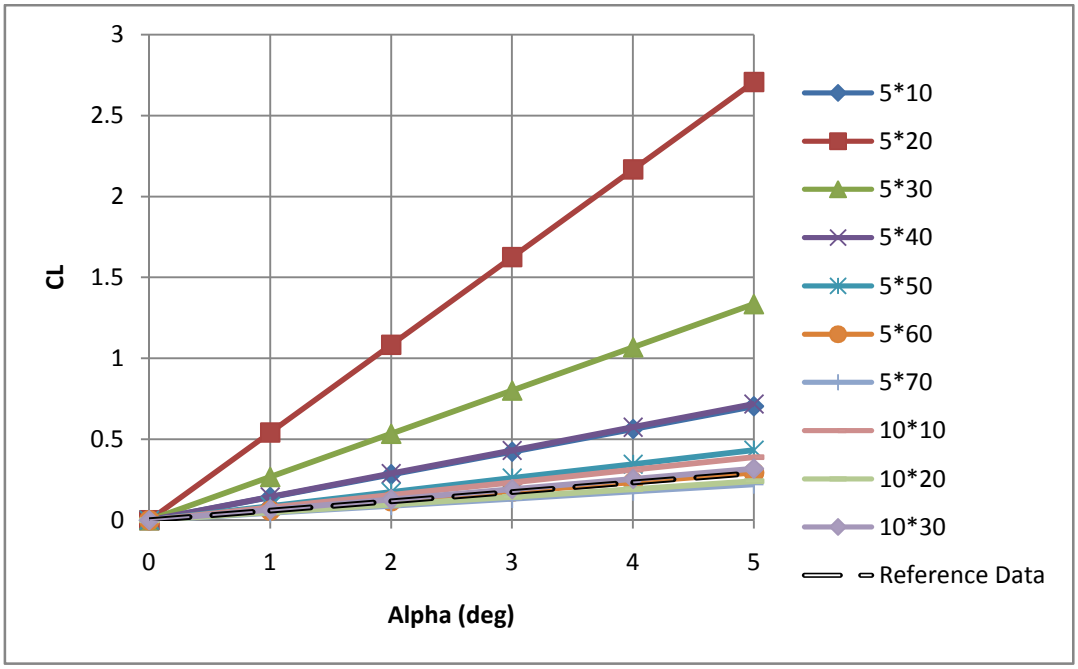


Figure 6-1: CL vs Alpha for Different Panelling Options

The closest result to the Reference Data was for the 5 chordwise by 60 spanwise panelling option. The detailed results are tabulated as follows;

Alpha (Deg)	Reference CL	SwanVLM CL (5x60 Panels)	Percentage Error
0	0	0	1.00
1	0.058119464	0.058803173	98.84
2	0.116238928	0.117588434	98.85
3	0.174358392	0.176337877	98.88
4	0.232477856	0.235033605	98.91
5	0.29059732	0.29365774	98.96

Table 1: Results for 5x60 Panelling

The run time for each problem was also recorded, to keep track of the requirement that the problem should be accessible with average computing power. The following graph and tabulated results detail the run time for a 1.6GHz x86 CPU, 1Gb RAM, desktop computer running MATLAB R2008a,

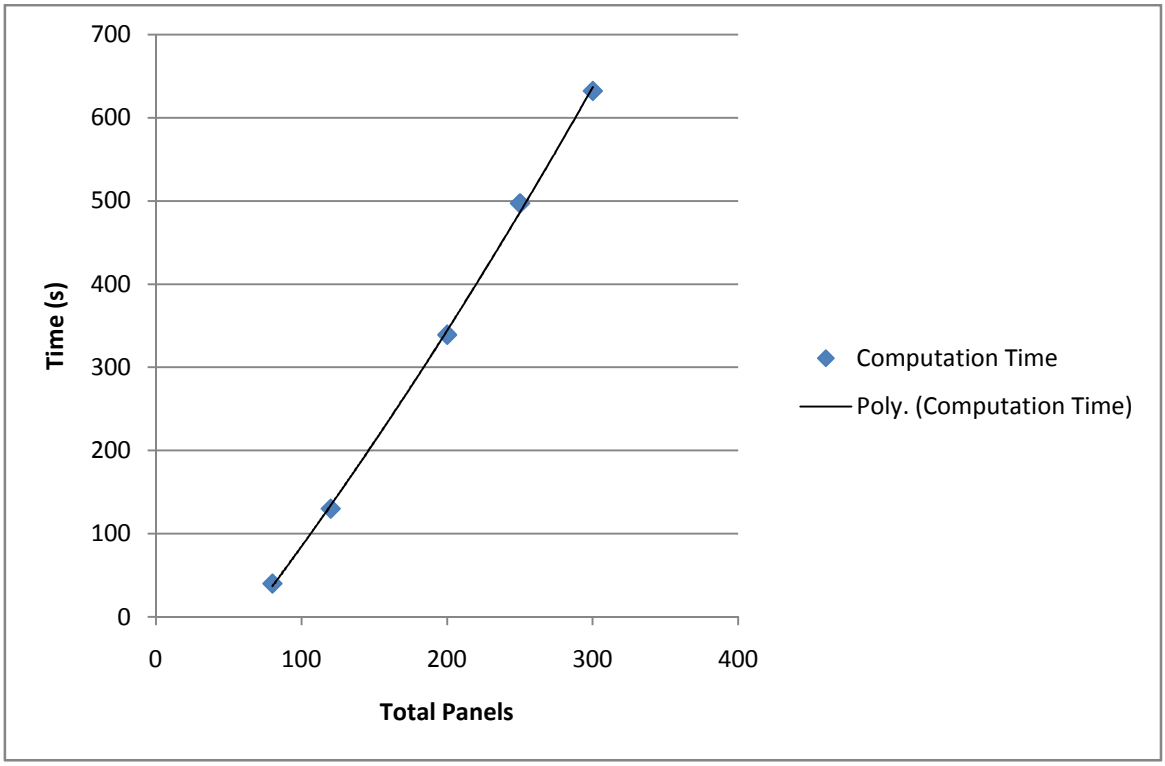


Figure 6-2: SwanVLM Grid Study Running Time

No. Of Panels	Run Time (sec)
80	40
120	130
200	339
250	497
300	632

Table 2: SwanVLM Grid Study Running Time

6.2 Validation against Tornado

As discussed in the literature review (Chapter 2), Melin [4] developed a VLM implementation called Tornado as part of his MSc research project. The accompanying thesis carries out a multitude of validation cases on the program, ranging from academic reference sources such as Bertin and Smith's reference planform, Prandtl's Lifting Line and Jones' Small Aspect Ratio Method.

It further extends the validation to compare Tornado against two other industry VLM codes and a panel method code for a Cessna 172 test case.

Generally speaking, as found in the thesis, Tornado compares very well to the other VLM codes, and the data sourced from actual flight testing.

Given that Tornado can therefore be considered validated and generates acceptable results for the scope of this project's problems, a range of test cases were put through SwanVLM and Tornado for validation purposes.

(N.B. The following graphs are annotated with reference to CJW_VLM, the development name of SwanVLM. Tornado is abbreviated as Tdo also.)

For the following series of results, the lift force has been non-dimensionalised in the usual manner;

$$C_L = \frac{L}{\frac{1}{2}\rho_\infty V_\infty^2 A} \tag{6.1}$$

Where; L - Lift Force (N)

ρ_∞ - Freestream Density (kgm^{-3})

V_∞ - Freestream Velocity (ms^{-1})

A – Reference Area

6.2.1 Clark-Y Series

A Clark-Y aerofoil was chosen and a range of runs were made using 5x60 panelling options and the same dimensions in both SwanVLM and Tornado;

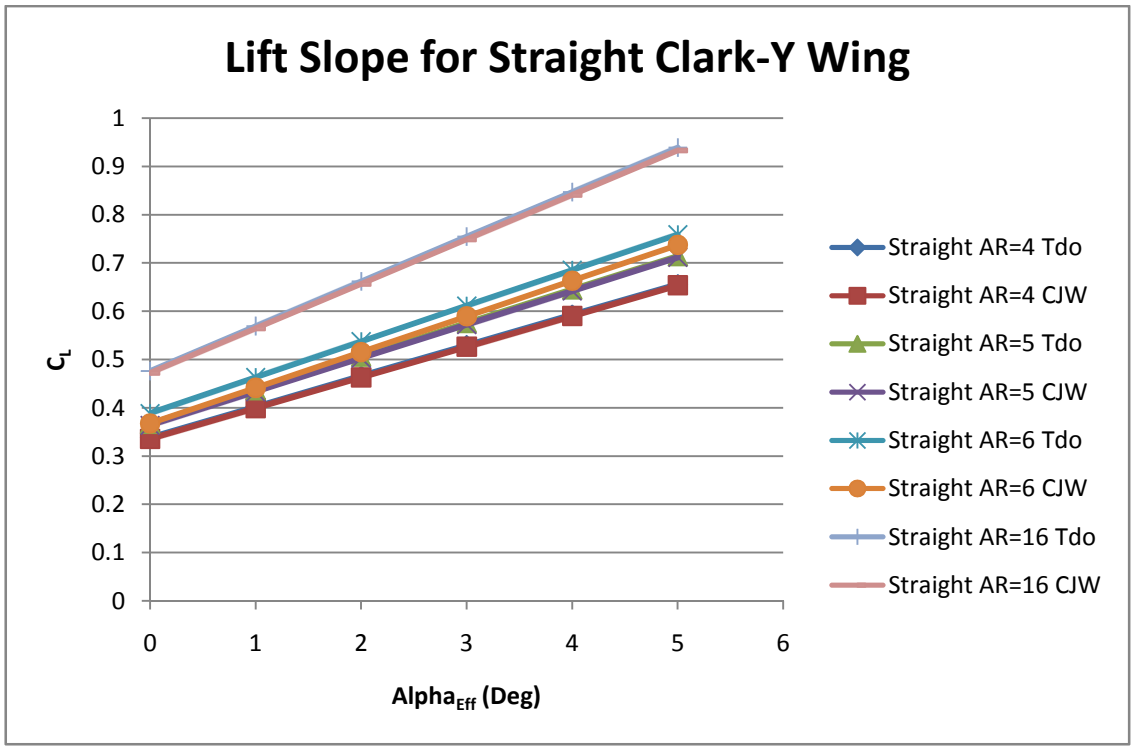


Figure 6-3: Lift Slope for Straight Clark-Y Wing

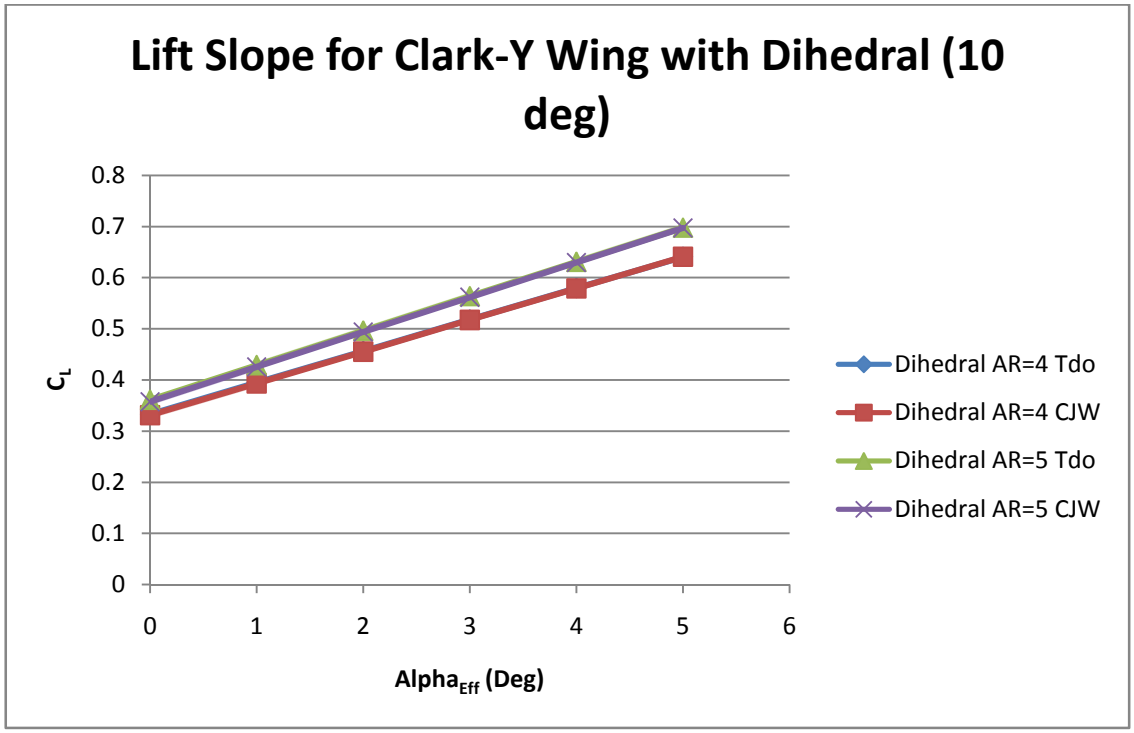


Figure 6-4: Lift Slope for Clark-Y Wing with Dihedral

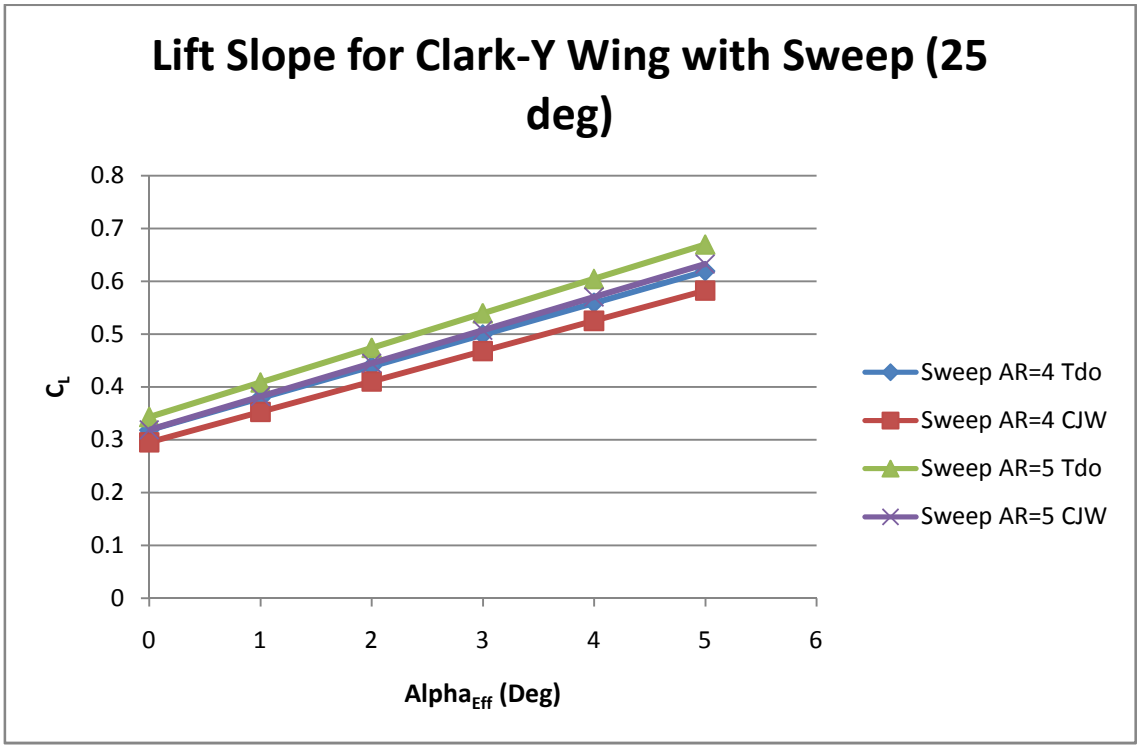


Figure 6-5: Lift Slope for Clark-Y Wing with Sweep

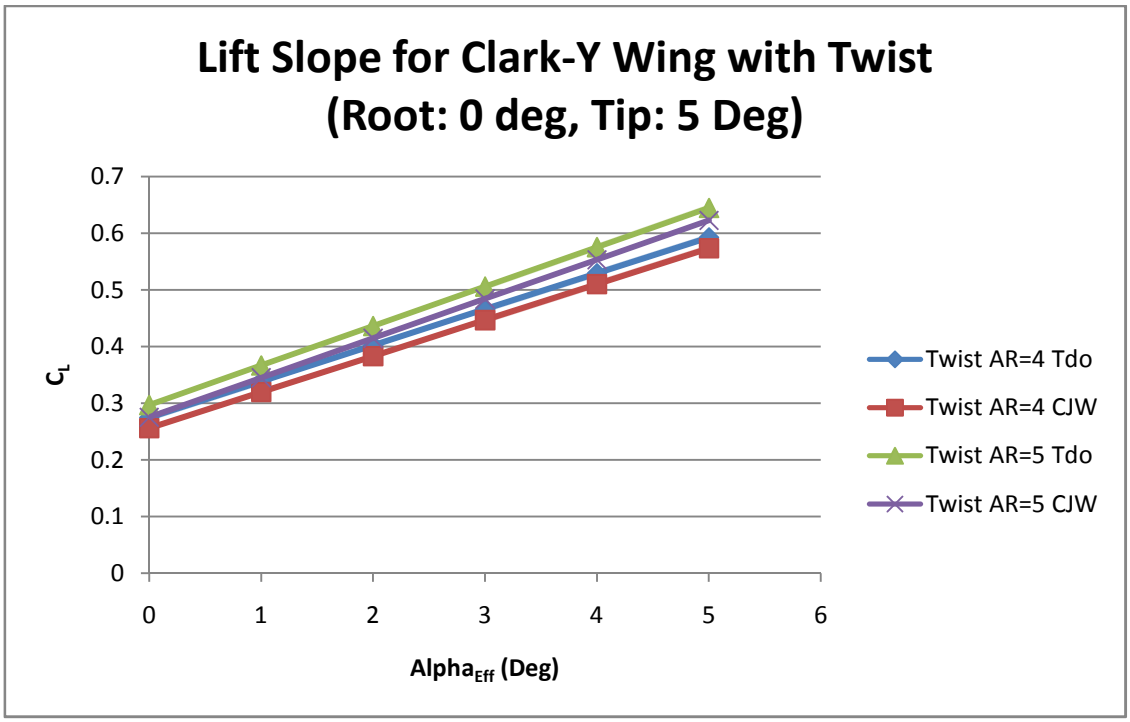


Figure 6-6: Lift Slope for Clark-Y Wing with Twist

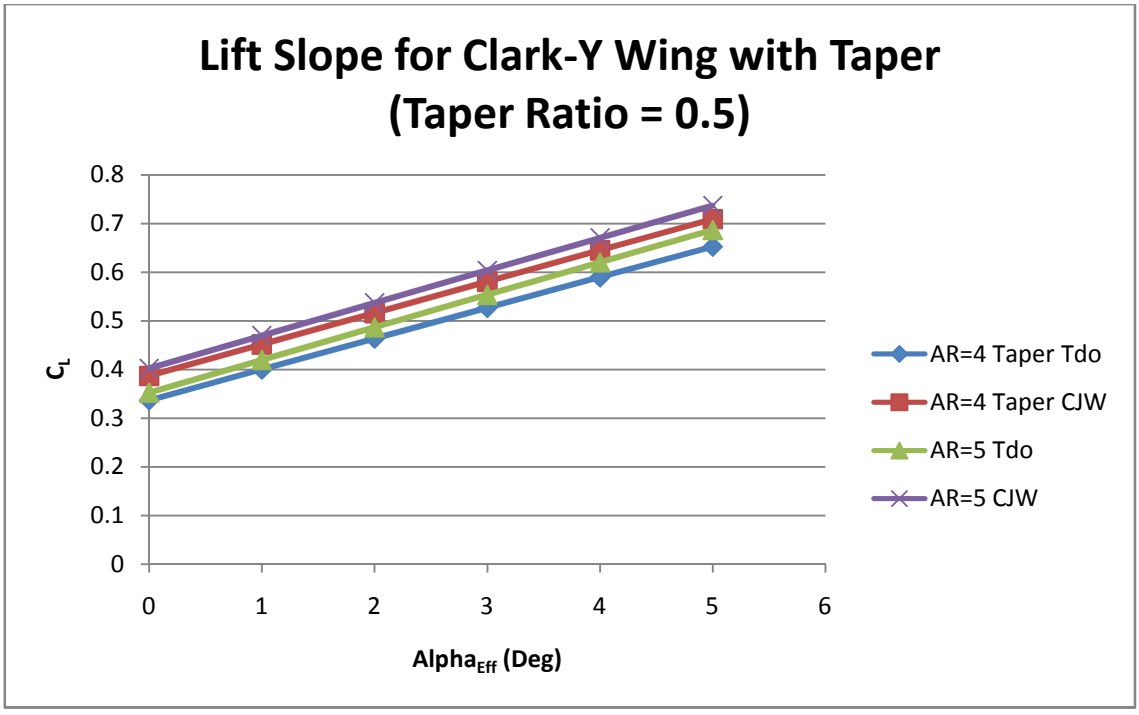


Figure 6-7: Lift Slope for Clark-Y Wing with Taper

6.2.2 Other Aerofoils

Additionally other aerofoils in a basic configuration (i.e. straight only) were examined.

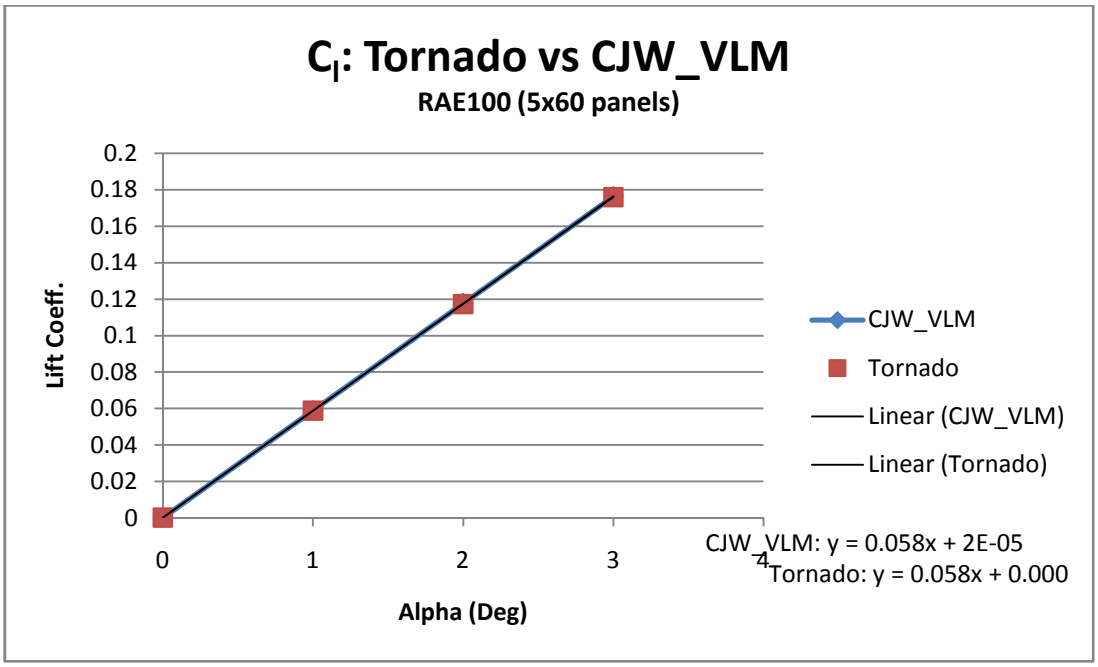


Figure 6-8: Lift Slope for RAE100 Wing

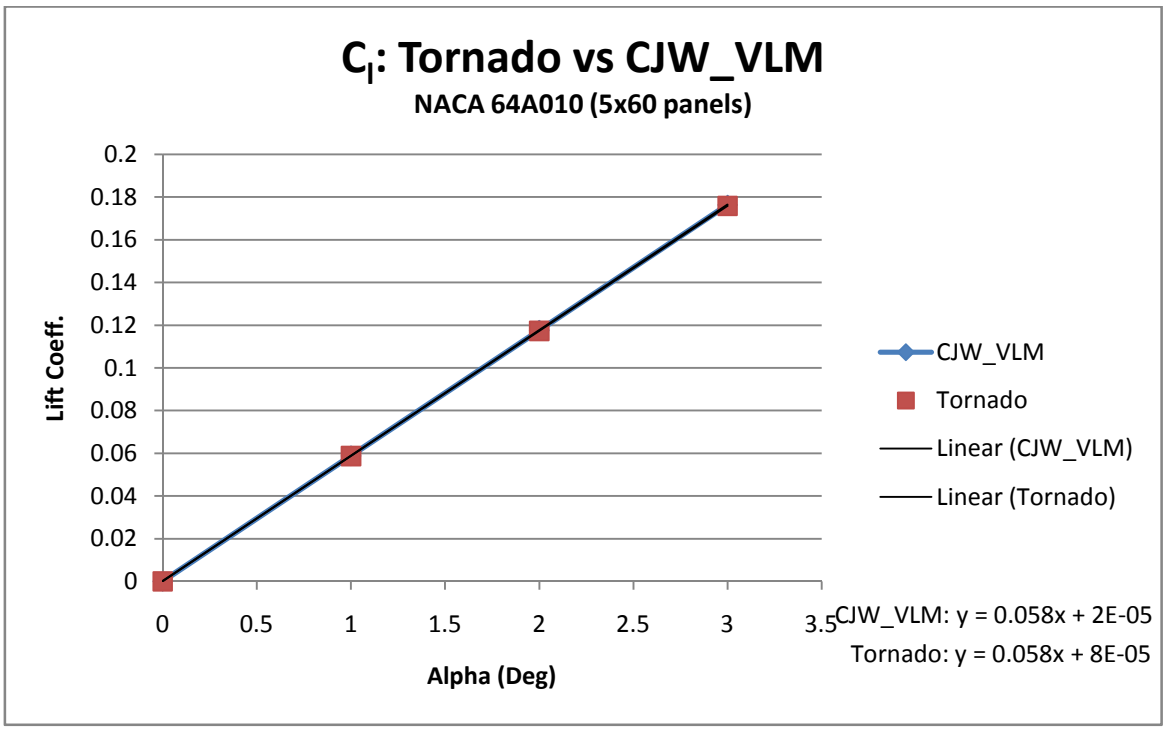


Figure 6-9: Lift Slope for NACA 64 A010 Wing

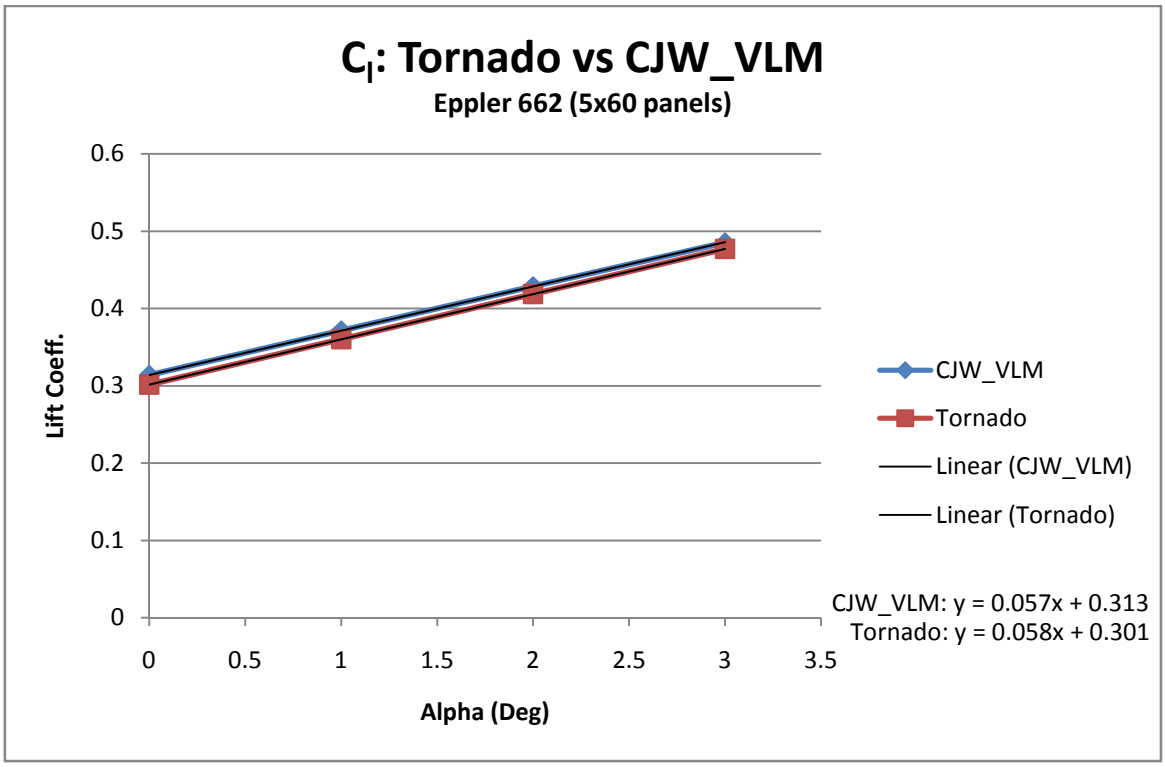


Figure 6-10: Lift Slope for Eppler 662 Wing

Chapter 7

Discussion

7.1 Analysis of Clark-Y Series

7.1.1 Straight

For all aspect ratios it was found that SwanVLM returned results that were extremely close to Tornado's output.

Of particular note is that the y-axis intercept ($C_{L\alpha=0}$) and gradient ($C_{L\alpha}$) were extremely close, with minor deviations seen in the intercept only.

This result provides a fundamental validation of the basic geometry generation, processing and post-processing in SwanVLM.

7.1.2 Dihedral

The two aspect ratios tested show an exact match between SwanVLM and Tornado.

This test provides a further validation of the geometry and boundary condition generation. It should be noted that in terms of lift generation only, it can be expected that,

$$C_{L_{Dihedral}} = C_{L_{Straight}} \times \cos\theta_{Dihedral} \tag{7.1}$$

It can be seen that the result from Equation (7.1) is directly found in SwanVLM's results.

7.1.3 Sweep

The sweep test run provides a potentially more insightful result, with a good match between the gradients, but a distinct difference between the y-intercept.

Examination of the results from the run shows that the difference between the y-intercepts for both runs is approximately equal. This suggests that the geometry is likely correct, but that the freestream component of the boundary conditions may differ slightly, in a consistent manner.

Given that the scale of the difference is small, and that the gradients are equal, the difference is acceptable.

7.1.4 Twist

The twisted wing results largely mirror the results found in the sweep test.

This further suggests that the underlying geometry is correct, but that the freestream boundary conditions are subtly different between SwanVLM and Tornado.

The relative order of the difference between results is again small, and acceptable.

7.1.5 Taper

The tapered wing case shows a difference of results consistent with those previously; that there is a difference in the y-axis intercept, but the gradients match.

The run offers a similar narrative to the previous results. The gradient match suggests that the geometry is correct, however the y-axis intercept difference is most likely due to a subtle difference in the freestream component of the boundary condition.

7.2 Analysis of Other Aerofoils

The RAE100 and NACA 64A010 runs are consistent with the result found for the straight Clark-Y wing section. They show a direct match between Tornado and CJW-VLM, providing further validation of the geometry capabilities in SwanVLM.

The Eppler 662 run shows a difference in the y-axis intercepts between SwanVLM and Tornado. In this case, closer examination of the profile co-ordinate definitions used by the two programs showed that there was a slight difference between the two, rendering the difference in the final result.

7.3 Computational Time

As expected (as discussed in Section 5.5.2), the computational time (Figure 6-2) required to complete a problem exhibits a slight quadratic trend in relation to the number of panels.

However the result has a predominantly linear relationship which suggests that a computational overhead may exist outside of the most computationally intensive routine (the influence matrix generation).

Additional analysis with M-LINT showed that MATLAB carries a consistent amount of overhead when compiling the functions at run-time. It also showed that the function which reads the input Excel file is relatively slow compared to SwanVLM's internal functions.

With pre-compilation of SwanVLM's component functions, and the input file previously read into MATLAB's workspace, a significant reduction of the computational overheads may be found.

7.4 Summary of Analysis

SwanVLM provides results very closely matched to Tornado, which serves as a validation of the entire process.

The small deviations observed in $C_{L\alpha=0}$ values between runs suggests a minor difference in the boundary conditions generated from the meshes generated by the two programs. While Tornado's approach to meshing is not extensively discussed in Melin's thesis [4], examination of the source code suggests that Tornado adopts a different approach to SwanVLM. Therefore a small difference in the end results can be expected, given the number and range of calculations performed based on the mesh.

Chapter 8

Conclusion

8.1 Summary

As set out in the project plan written at the commencement of the research project, this study aimed to;

-Research key principles and applications of Vortex Lattice Methods

-Develop a MATLAB program capable of implementing these methods

-Tailor the program towards being able to handle practical problems

Reviewing the work carried out over the duration of the project, it would be very fair to say that these tasks were carried out to the fullest possible extent within the time available.

The final result of the efforts was the MATLAB implementation of a Vortex Lattice Method, the program SwanVLM. This program offers a validated, original approach to the traditional VLM method, with an easy and intuitive interface and output.

8.2 Conclusions

8.2.1 Research

The VLM is well documented, although the minor nuances of its implementations are not well covered by any single text. Furthermore there is currently only one widely available resource which offers a clear guide on the routes to implementing the VLM as a computer code.

The research conducted and presented in this thesis offers a concise guide, with a clear computational/programming relevant emphasis in the content. Should a student commence a similar project in the future, the work presented in this thesis offers a strong route-in to the *ab initio* student.

8.2.2 Program Development

The main challenge in this work was in writing the MATLAB code to implement the VLM.

Whilst the concept is both algorithmically and numerically straightforward once decomposed, the arbitrary meshing presents a great challenge and represents ~50% of SwanVLM's total source-code content.

SwanVLM in its final form offers a competent meshing algorithm which is very user friendly, and is also open to extensive customisation.

The validation runs, as commented on in Chapter 7, demonstrate that SwanVLM is capable of matching current well-developed VLM implementations. Whilst SwanVLM can be compared to the other VLM code used in this project (Tornado), the actual employment of the two differs greatly.

Tornado offers a mature design with extensive capability; however SwanVLM is still in a very modular form, allowing it to be easily modified by other students for their own purposes. SwanVLM is also straightforward in its employment, making for ease of use in both the academic setting, and for first stage designs.

8.2.3 Applicability

The two main avenues of program applicability set out were for use in undergraduate study, and for first stage design of suitable vehicles like UAVs and light aircraft.

SwanVLM offers a range of output entirely suitable for studying Thin Aerofoil Theory approximations to forces and moments generated by a 3D wing. Furthermore, it's graphical capability, and in particular the animated pressure distribution output, allow for interactive learning within the confines of Thin Aerofoil Theory.

With regards to the latter aims, the educational institution at which SwanVLM was developed features a 6 Degree-of-Freedom flight simulator with extensive parameter input options available. SwanVLM was successfully used to finalise the design of, and generate baseline data for, a basic UAV geometry. Whilst it only covers a narrow range of the work required in total aircraft design, it can significantly reduce the early groundwork leading towards a refined design. In this regard, SwanVLM is very useful.

8.2.4 Further Work

There are a number of extensions which could be made to this study, both in pure theoretical terms, and in the computational implementation.

In the theoretical sense, there have been a number of extensions to the VLM suggested in the works referenced in writing this thesis. One of potential note is a modification of the pressure relation (4.1.2) to allow the treatment of compressibility effects, i.e. to handle higher Mach numbers than otherwise possible.

The computational element to this project potentially offers more scope. SwanVLM was written in a highly modular manner, which would allow for component parts of the program to be put to use for other projects, or modified to offer more flexibility.

It would certainly be feasible to study a more suitable meshing approach, possibly using a non-linear grid distribution. The processor is fairly mature in its design, however work could be carried out towards exploiting properties of the Influence Coefficient Matrix for more efficient computation.

The post-processing of the vortex strengths offers many potential improvements relating to moment and stability calculation, aspects which were omitted from SwanVLM due to time constraints.

In summary, SwanVLM offers a good starting point from which a more capable program can be developed.

References

[1]. **Anderson, John D. Jr.** *Fundamentals of Aerodynamics.* s.l. : McGrawHill Higher Education, 2005 (Fourth Edition). 007-125408-0.

[2]. **Katz, Joseph and Plotkin, Allen.** *Low-Speed Aerodynamics.* s.l. : Cambridge University Press, 2001. 978-0-521-66552-0.

[3]. **Mason, WH.** *Class notes for AOE 4114, Applied Computational Aerodynamics.* Dept of Aerospace and Ocean Engineering, Virginia Polytechnic Institute and State University. s.l. : Author - Internet, 1998. http://www.aoe.vt.edu/~mason/Mason_f/CAtxtTop.html.

[4]. **Melin, Tomas.** *A Vortex Lattice MATALAB Implementation for Linear Aerodynamic Wing Applications.* Royal Institute of Technology (KTH). s.l. : Author, 2000. MSc Thesis (http://www.redhammer.se/tornado/tornado_files/thesis.pdf).

[5]. **NASA.** *Vortex-lattice utilization.* Langley : NASA, 1976. SP-405.

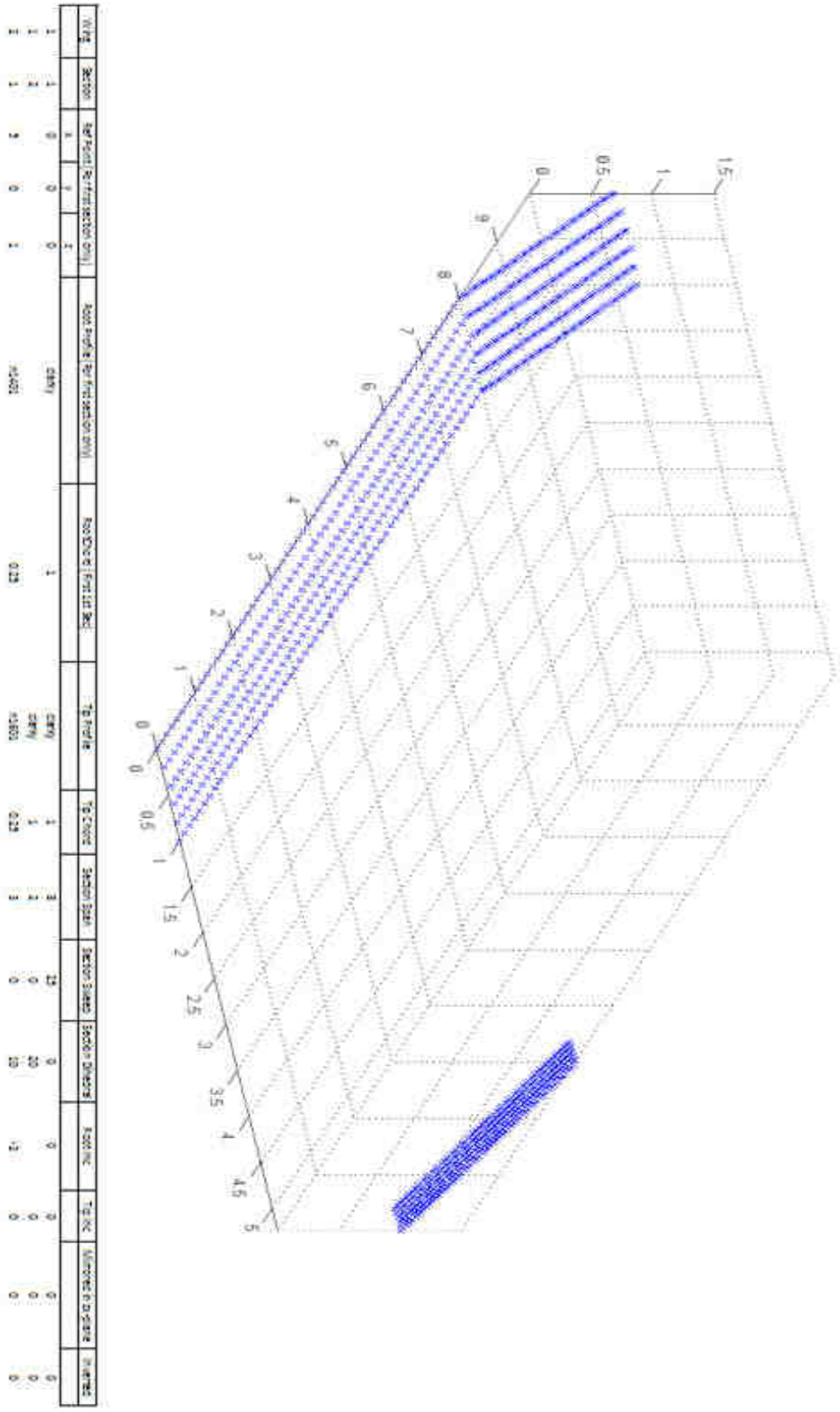
[6]. **Prandtl, L.** *Applications of modern hydrodynamics to aeronautics.* s.l. : NASA, 1923. NACA-TR-116.

[7]. **Falkner, V.M.** *The Accuracy of Calculations Based on Vortex Lattice Theory.* s.l. : British ARC, 1946. Rep No. 9621.

[8]. **Werner, Sofia.** *Application of the Vortex Lattice Method to Yacht Sails.* Naval Architecture and Ocean Engineering, Chalmers University of Technology. Gothenburg, Sweden : s.n., 2001. http://www.na.chalmers.se/~sw/sofiawernerMscreport.pdf.

Appendix A

Example Input File and Mesh



Appendix B

Example of Output Data File

Results for run completed on:17-Mar-2009 19:55:27			
Alpha_Geo (Deg)	Alpha_Effective (Deg)	CL	CD
0	-2.045828091	0.237956589	0.002748938
1	-1.045828091	0.318547171	0.004670332
2	-0.045828091	0.399040719	0.007166107
3	0.954171909	0.479412716	0.010233223
4	1.954171909	0.55963868	0.013867942
5	2.954171909	0.639694172	0.018065836
CL_Alpha=0	0.23816731		
CL_Alpha	0.080352412		
K	0.043459827		
Aero. Centre	0.34157, 0, 0		
CM_Alpha	-0.027445914		
Static Margin	34.15692653	%	
Ref. Chord	1		
Ref. Area	8		

Appendix C

Attached SwanVLM DVD

A DVD containing SwanVLM's source code and a simple user guide has been attached to the end of this thesis.

School of Engineering

EG-353 Research Project

Session 2008/09

Project Planning Statement

The Aerodynamics of 3D Lifting Surfaces Using Vortex Lattice Methods

Chris Walton
368404

Professor K. Morgan

Aims

- Research key principles and applications of Vortex Lattice Methods
- Develop a MATLAB program capable of implementing these methods
- Tailor the program towards being able to handle practical problems

Project Planning Statement

Description

Vortex lattice methods (VLM) are an early computational technique used for basic lift and drag estimation of a surface. By extension they can be used to provide additional data such as pitch, roll and yaw moments, and stability coefficients.

VLMs are essentially methods based upon Thin Aerofoil Theory approaches to solutions of the Laplace equation.

VLM computer codes have been widely written from approximately the mid-1980s and onwards, with many notable versions produced by professionals working for bodies such as NASA. Additionally during the course of this work, many improvements, both in computational and analytical terms, have been produced or suggested.

The main purpose of this project is;

-To research the theory behind VLMs in theoretical terms.

-Study previous computational implementations and extensions to the theory.

-Implement the method into a useful MATLAB program and where possible extend the code based upon previous VLM work.

-Once a working core software engine has been developed which can render consistent aerodynamic estimations, I intend to develop the software to a level which would prove useful as a first order tool for an aircraft designer or for educational purposes within the university.

Main tasks

1. Literature search and review
2. Project planning
3. Design basic software architecture
4. Write prototype software
5. Test prototype software
6. Freeze core software engine design (on basis of prototype results)
7. Develop user interface and results display towards being a practical design tool
8. Test completed software
9. Write thesis

Milestones

1. Project Planning Statement
   -This document.
   - Deadline: 24th Oct 2008
2. Progress Report
   -Two page summary outlining progress to date.
   -Deadline: 12th December
3. Prototype software design
   -Begin writing trial implementations of different VLM methodologies.
   -Aim for a basic level in input/output functionality (i.e. lift and drag estimations)
4. Prototype testing
   -Test all developed prototypes with a range of standard test case.
5. Core software engine design freeze
   - On basis of testing, choose one software approach.
6. Interface development
   - Develop a user interface for the software engine.
   - Work towards developing the software as a design tool.
7. Testing
   - Trial software against standard cases and compare results.
8. Summary Paper
9. Thesis
10. Presentation

Gantt Chart

Task ID	Name	Start Date	Finish Date
1	Project Planning Statement	13/10/2008	24/10/2008
2	Progress Report	13/10/2008	24/10/2008
3	Prototype software high-level design	13/10/2008	24/10/2008
4	Prototype software coding	24/10/2008	20/11/2008
5	Prototype testing	20/11/2008	04/12/2008
6	Core software engine design freeze	04/12/2008	04/12/2008
7	Interface development	04/12/2008	26/01/2009
8	Testing	26/01/2009	09/01/2009
9	Summary Paper	09/01/2009	16/01/2009
10	Thesis	16/01/2009	30/03/2009
11	Presentation	30/03/2009	13/04/2008

Month	Oct			Nov				Dec					Jan				Feb				Mar					Apr			
Week Beginning Monday	13	20	27	3	10	17	24	1	8	15	22	29	5	12	19	26	2	9	16	23	2	9	16	23	30	6	13	20	27
		X																											
		X																											
									X																				
											Holidays			Exams															
																													X

Thesis Deadline

Progress Summary Report

Legend	
Working week	
Deadline due this week	
Deliverable	X

School of Engineering

EG-353 Research Project Progress Report

The 3D Aerodynamics of Lifting Surfaces Using Vortex Lattice Methods

Christopher Walton (368404)

Professor K. Morgan

Achievements to Date

To date the project has proceeded at a good pace, with many challenges met and suitably dealt with. In the main the date structure for major project items, as detailed in the Project Plan Gantt chart, has been adhered to.

Initial progress was very rapid; I was able to write a functional and valid Vortex Lattice Method (VLM) for flat plates using Horseshoe Vortices by the close of the second week of the academic term. The route to this outcome allowed me to study the majority of the currently available and relevant literature for this project. It also gave me a sound understanding of the principles at work at the heart of the VLM approach.

While this early advance was relatively smooth, there then followed a period of challenges which in some cases required a fairly significant amount of time to resolve. One of the key aspects to making the intended VLM program relevant and useful is for it to be able to handle cambered wing surfaces. On the basis of research from relevant textbooks, I set off down the path of using a Vortex Ring element approach to allow the handling of cambered surfaces.

After approximately three weeks of determined efforts, I found the Vortex Ring method largely unworkable. My implementation of the method could not consistently return valid, useful results for basic geometries.

Following this set back I started to look towards developing flat plate methodologies to provide an approximation to cambered surfaces. It was at this stage, drawing in part from an MSc thesis on a similar program, and an element of experimentation, I developed an approach which placed cambered wing surface boundary conditions on to a flat plate method using Horseshoe Vortices. I found that the results produced by this approach were both consistent and valid.

While the three weeks spent researching the Vortex Ring approach was productive in terms of researching alternative methods and refining my computer programming techniques, it has ultimately made for a deviation from the Gantt chart, seeing an almost direct three week shift backwards.

However in conclusion I feel it is possible to absorb this developmental delay into the overall structure outlined in my project plan, and deliver the intended program within a sensible time-frame.

Planned Activities

At this stage the bulk of the core software engine is complete. What is now required is refinement in terms of reducing the number of computational operations required for each run and generally streamlining the code.

From that point a sound validation will need to be devised and carried out. This will nominally be a series of runs of a defined geometry with known results to compare against. The main intent of this process is to ensure that consistent trends can be predicted for 3D geometries (this being most critical to later stability calculations).

With these areas complete, a period of interface and data post-processing development will be required, essentially designing software to effectively employ the core software in a relevant sense for an aircraft designer.

Chris Walton

December 2008