WING-NACELLE ASSEMBLY MULTIDISCIPLINARY PERFORMANCE OPTIMIZATION

A Thesis
Presented to the Faculty of
California Polytechnic State University
San Luis Obispo

In Partial Fulfillment
of the Requirements for the Degree of
Master of Science in Aerospace Engineering

by
Yevgeniy Mikhaylovich Gisin
March 2006

Approval Page

TITLE: WING-NACELLE ASSEMBLY MULTIDISCPILINARY PERFORMANCE OPTIMIZATION

AUTHOR: Yevgeniy Mikhaylovich Gisin

DATE SUBMITTED: March 2006

Dr. David Marshall _____
Advisor, Committee Chairman Signature

Dr. Larry Erickson _____
Committee Member Signature

Dr. Lanny Griffin _____
Committee Member Signature

Dr. Jin Tso _____
Committee Member Signature

Abstract

WING-NACELLE ASSEMBLY MULTIDISCPILINARY PERFORMANCE OPTIMIZATION – Yevgeniy Mikhaylovich Gisin

The issue of decreasing cruise drag has always been at the forefront of aircraft technology development. In the current state of the art in subsonic airliner design, a below-wing podded installation of a turbojet engine is considered traditional, largely because of the difficulty involved with the aerodynamic integration of an above-wing installation. However, benefits of such an installation are theorized to be higher when applied to a design of a notional Business Jet Aircraft. A drag-reducing benefit from a variant of such an installation has been claimed in the US Patent #6,308,913. Other benefits include, but are not limited to: FOD (foreign object damage) alleviation, ease of landing-gear integration and a possible decrease in ground-perceived noise both in takeoff and cruise conditions. An investigation was conducted to verify the research described in the patent and to quantify the actual benefit derived from this drag reduction measure. The installation is designed to reduce wave drag during transonic cruise; therefore the wing-nacelle geometry was analyzed at a C_L value of 0.4 and Mach number of 0.75.

An aerodynamically ideal location for the nacelle was ascertained, and drag characteristics calculated. A multidisciplinary optimization study, considering the effects of drag, structural weight and aircraft mission was performed. This allowed the investigator to obtain the optimum nacelle location for an aircraft. This study does not, and was not intended to have the fidelity necessary for detailed design of an airframe-engine installation, but is instead intended to provide guidelines for use in the preliminary-design field.

Acknowledgements

The author wishes to thank a whole lot of people for providing help and encouragement during the time spent working on this thesis. Most importantly, Dr. Marshall's help and easy-going support were invaluable in helping shape the topic of this thesis and in work to its completion. Thanks to Dr. Erickson for his help in the beginning of this research, and for introducing me to TranAir and AGPS. Steve Kubik's help was valuable in many ways: equation derivation, AGPS code bug-checking, and his UNIX magic in getting TranAir to work.

Thanks to Ross Sheckler and Garvin Forrester of Calmar Research for answering my countless questions on TranAir and AGPS, and being an immense resource of information – without them this project wouldn't have been possible. I am also very grateful to Paul Dees and his friends at Boeing for solving so many TranAir issues with a single e-mail – that truly breathed life into this project. I want to acknowledge Professors Dave Hall and Mark Waters for their advice and support during the development stage of this thesis. Props to Jeff Nadel for keeping the TranAir server up and running for all the 3000+ runs I put it through.

Of course, lots of love and innumerable thanks to: My parents, for being patient with their prodigal son during all his years at Cal Poly and this summer-fall adventure. All my roommates, without whose help I would have had a much less fun time working on this thesis. Clint Besheer for his interest and excitement for this topic. And finally, good weather (or perhaps global warming), for letting me sleep and work outside for all of the summer and late into the fall.

Table of Contents

List of Figures

List of Tables

Nomenclature

Abbreviations	
AOA	Angle of attack
c#r#	Nacelle Location in %MAC and # of nacelle radii
CFD	Computational Fluid Dynamics
C_L	Coefficient of Lift
FOD	Foreign Object Damage
G	Gravity
L/D	Lift-to-Drag Ratio
LBO	Local Box of Interest
LE	Leading Edge
MAC	Mean Aerodynamic Chord
MDO	Multi-Disciplinary Optimization
GCI	Grid Convergence Index
GMRES	Generalized Minimal RESidual algorithm
Symbols	
C_d	Drag coefficient
C_f	Friction coefficient
f	Value of solution
H	Shape factor
H^*	Shape factor: Kinetic energy thickness/Momentum loss thickness
H^{**}	Shape factor: Density thickness/Momentum loss thickness
M	Mach number, Unit vector in network row direction
N	Unit vector in network column direction
$\hat{n}$	Normal vector
p	Observed order of convergence
Q	Source
r	Grid refinement ratio
S	AGPS parametric direction 1
T	AGPS parametric direction 2
u, v, w	Velocity in X, Y, Z directions
$\bar{W}$	Mass flux vector
X	Free-stream flow direction
ξ	Green's function
ρ	Density
θ	Momentum loss thickness
τ	Continuous far field (Prandtl-Glauert) operator
Φ	Total potential
∇	Gradient vector operator

Subscripts	
1,2,3	Solution grid density 1=highest
D	Drag
e	Edge of the boundary layer
h=0	Solution at infinite grid density
L	Lift
t	Time
u	Velocity in the x-direction

1. Introduction

A wide variety of air intake geometries have found use on modern turbojet-powered aircraft, with each type having its benefits and disadvantages. In all conventional engine installations the designer attempts to reduce the normally harmful effects of the aerodynamic interaction between the nacelle and the wing. Because of this consideration, most modern engines are installed either in nacelles separate from the aircraft body or use intakes located either below or to the side of the aircraft body.

When considering the distinct set of challenges facing the designer of a small passenger jet aircraft, an engine mounting location on the upper body of the aircraft gains a number of significant benefits. Since there is no need to create space for wing-mounted engines underneath the aircraft, a short main landing gear can be used. The reduction in aircraft height also makes it better suited for field operations such as servicing and passenger loading. Most popular variant of such an installation is seen on the majority of today's business jets – with the engines located on pylons at the rear of the fuselage. An example of such an installation on the Gulfstream V business jet can be seen in Figure 1.

source: www.spacemodel.com

Figure 1 – Gulfstream V

Although advantageous in some ways, the rear-mounted engine configuration has a number of disadvantages to go with the benefits. The engine mounting structure decreases the internal fuselage volume available to the passengers. With the engines located close to the fuselage, noise concerns can also become an issue. It becomes very difficult to locate the engines too far behind the wing's aerodynamic center, because of the increasing difficulty of ensuring a controllable aircraft center of gravity position.

Another engine mounting possibility exists in locating the engines on top of the wing surface. Being mounted above a wing re-directs engine noise up and away from the ground, significantly reducing noise emissions from the aircraft. The weight of the engines reduces the bending moment generated by the aerodynamic forces acting on the wing. A degree of structural synergy can be obtained by using wing structure to carry both the engine and main landing gear loads, while space in the rear of the cabin is opened for either passenger or cargo use. Concerns of foreign object damage (FOD) to the engine are also greatly alleviated.

The advantages mentioned earlier for fuselage-mounted engines remain in effect for this mounting scheme, with a number of additional benefits being present. There are, however a number of concerns, such as cabin noise, passenger ballistic protection from turbine failure and finally the aerodynamic interaction between the wing and nacelle, that must be considered instead. In a number of historical aircraft designs the trade-off performed in choosing an engine location has lead to such an engine installation location.

Fokker VFW-614 is one of the more well-known subsonic aircraft to employ an above-wing engine installation. In this case, the main reason for this installation choice was to allow easy operation from unprepared runways by limiting the incidence of

foreign object ingestion. The lack of engine efflux below the trailing edge of the wing allowed for use of the more efficient one-piece flaps, improving the aircraft's short-field performance.[1]

source: DaimlerChrysler Aerospace

Figure 2 – Fokker VFW-614

Although it possessed impressive flight characteristics and is now popularly acknowledged as an aircraft ahead of its time, the VFW-614 did not sell well, and the programme became an economic failure. It is only approximately 20 years later that the full promise of the regional jet, which the VFW-614 was the first of, was realized. A number of political and circumstantial reasons can be blamed for the failure – with the 1973 oil crisis being a major contributor to the low customer demand.[2]

Another aircraft that incorporates the above-wing engine mounting, albeit for very different reasons, is the HondaJet research jet. A three-view of this aircraft can be seen in Figure 3.

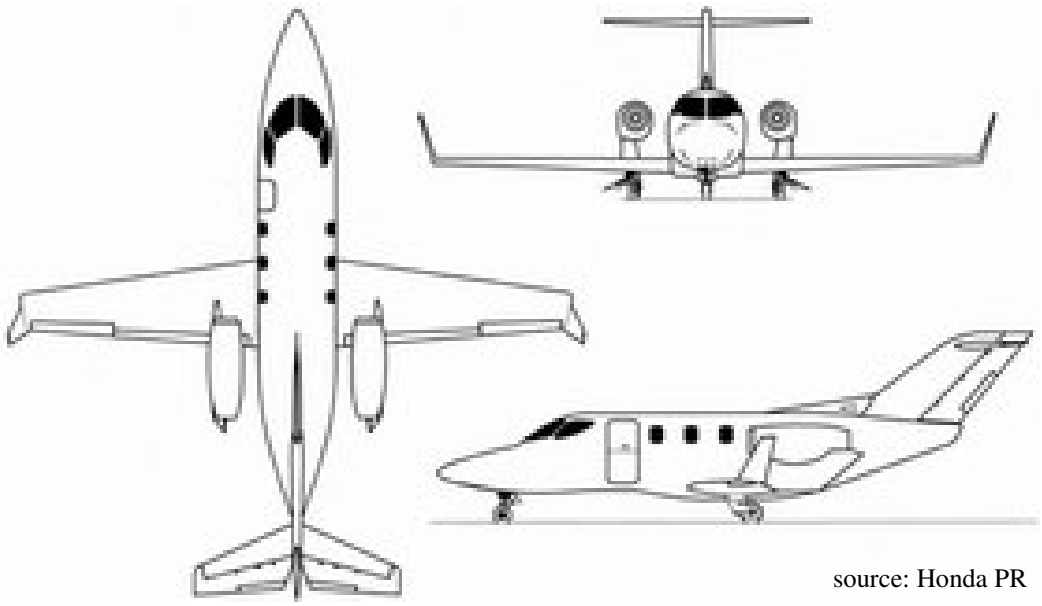


Figure 3 – HondaJet General Layout

This aircraft was designed to thoroughly test the invention described in the patent #6,308,913. As mentioned previously, the main reason for this engine installation is the reduction of trans-sonic wave drag on the top wing surface.[3] The flow field of the intake is superimposed on that of the wing, reducing the negative pressure peak and retarding the generation of a shock wave, thereby increasing the aircraft's divergence drag number. This engine installation brings a number of other advantages to the aircraft configuration – short and simple landing gear, lack of FOD ingestion issues, and a larger cabin volume afforded by the lack of an engine carrythrough.

2. Research scope and definition

Most modern jet-powered airliners cruise in the transonic regime of flight. Although the relative speed of the aircraft is below that of sound, pressure gradients result in localized areas of supersonic flow. These mainly occur above the wing surface. The supersonic shock associated with such flow results in a dramatic increase in drag for the aircraft, limiting the maximum flight Mach number. A number of approaches have been introduced to combat the occurrence of this phenomenon, with the above-wing mounted engine being one.

This research aims to explore an above-wing engine installation similar to that discussed in the patent #6,308,913 at a single Mach number. A set of aircraft wing geometries with an upper-surface mounted engine intake are evaluated in the Boeing TranAir potential flow CFD code. The research is conducted at a pre-assigned "cruise" condition for a notional aircraft, defined by Mach=0.75 and $C_L = 0.4$. These values were used in order to conduct this research in conditions similar to those experienced by a hypothetical business jet aircraft during the cruise portion of its mission. Although an engine intake location highly-separated from the wing surface may derive the highest benefit from the supersonic aerodynamics viewpoint, this study attempts to optimize the engine location by also evaluating a structural constraint, which naturally acts to prohibit an excessively complicated engine mounting scheme. Using these two criteria, an optimum intake location is defined.

There are two fundamental methods of obtaining the type of data required – using a wind tunnel and a computational fluid dynamics program. Because of a lack of availability of a wind tunnel capable of operating at the relatively high Mach number

required of this study, the second method had to be used. The research conducted limited itself to low-angle-of-attack aerodynamics and non-separated boundary layers – therefore a code which incorporated Integral Boundary Layer techniques was considered a method usable for this purpose.

The study to find optimum wing performance will focus on the effects of varying the X and Z coordinate location of the nacelle in respect to the wing surface. The Y coordinate will be kept constant throughout the study, and in order to simplify the amount of parameters affecting the trade study, no aircraft fuselage effects will be considered or included. Having specified a notional aircraft configuration, the cruise angle of attack, and wing area could both be determined. Because the cruise condition is where the aircraft's efficiency is most important, the engine location was optimized to reduce transonic drag at that point. In order to produce more generalized aerodynamic results, it was decided to not sweep either the leading or the trailing edge of the geometry analyzed in the TranAir.

The general process of the study undertaken can be explained by the flowchart presented in Figure 4.

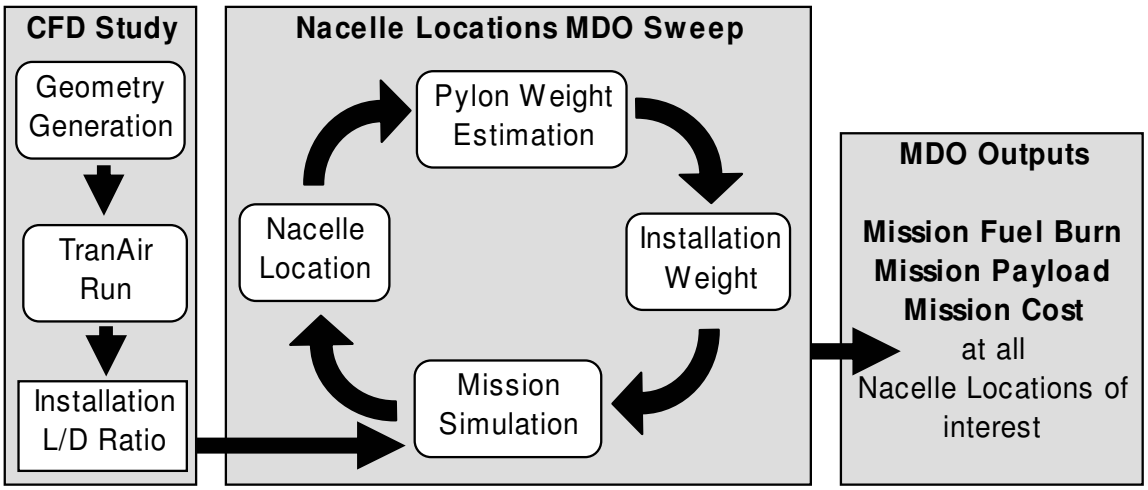


Figure 4 – Optimization Study Flowchart

During the CFD study a set of wing-nacelle geometries defined by a range of nacelle locations is evaluated at a C_L of 0.4 and a freestream Mach number of 0.75 to obtain L/D values at each nacelle location point. The operations shown in the MDO section of the flowchart above are performed within a user-specified nacelle position range. The optimization routine sweeps all the possible nacelle locations of interest, which are normally limited by the range of available CFD data. Using a mission simulation routine at each incremental nacelle position, the benefits to the notional aircraft are obtained in real-world terms such as change in mission range, payload or operations cost. From an evaluation of these aircraft performance metrics, an optimum overall nacelle location point can be determined.

3. CFD Tools Description

A wide variety of computational fluid dynamics codes could have potentially been used to solve this problem. Lacking a need to simulate fully-separated and highly turbulent flows, a full Navier-Stokes code was judged excessive for this study. While Navier-Stokes solutions would be desired for many flow problems, unfortunately such computation is rarely practical for a full 3D complex configuration, due to the fine grid required, parabolic nature of the equations in a shear layer, and the dependence of the solution on the turbulence model.[4]

The ready availability of the Boeing TranAir (A633) code on CalPoly campus, and the number of advanced features incorporated in this code made it an attractive choice. Automatic grid refinement, integral calculation of moderately turbulent boundary layers and the fast run-times offered by this code were all considered beneficial to conducting an accurate and efficient study. One of the main distinguishing features of TranAir is its use of a rectangular configuration grid superimposed over the configuration surface. This allows for a simple grid generation procedure, thus simplifying and speeding up the code execution.

The TranAir code requires a relatively powerful UNIX-based system to operate, for this reason it was run remotely on a dedicated server. TranAir solves the nonlinear, full potential equation with the viscous effects of a coupled boundary layer being modeled using either the A411 or ISES codes. The output of TranAir code is presented in a large formatted data file with a variety of flow characteristics calculated at the corner points of panels that make up the geometry. Both the geometry generation and the results

post-processing for this study were accomplished with the use of the Boeing Aero Grid and Paneling System[5] (AGPS) tool.

AGPS is an interactive, three-dimensional programming language which incorporates a parametric surface geometry system. It features geometry, grid, and graphics capabilities necessary for effective and rapid CFD code use, including several curve and surface generation options, curve and surface intersections, raster imaging, structured grid extraction, several supported input formats and flexible grid output capability for interfacing with engineering analysis codes.[6] Because it is at its core a programming language, AGPS users have been writing command files for more than 15 years, and a wide array of useful tools has been generated. Unfortunately, few of these tools are available for public use. For this reason, a customized "geometry generation module" had to be created from scratch in order to be able to perform trade studies on the wing-nacelle geometry.

One of the most attractive characteristics of the AGPS/TranAir combination is their particular suitability to use in a rapid preliminary-design environment. Unlike many other available mesh generators, AGPS allows for high amount of scripting of both the surface generation and meshing processes – thereby practically automating the geometry generation phase of the study. The TranAir code was designed to provide fast results in a design environment when evaluating transonic aircraft at cruise condition - a task very similar to that this thesis aimed to perform. The combination of these two programs was natively designed to work together, and was chosen because it appeared to be an ideal set of tools with which to perform this study.

4. Solver Fundamentals

4.1. CFD Analysis Basics

CFD is a technology which is gaining more and more acceptance as an important tool for the modern aerodynamicist. An increase in computing power has made it possible to use flow models to analyze fluid flow around geometries sufficiently complicated to be of use in aircraft design. Most commonly used CFD codes divide the flow domain around the geometry into a volume mesh, and then use a chosen algorithm to solve the equations in each individual mesh cell.

4.2. Flow Solution

The nonlinear, full-potential flow equation is the basis of the TranAir code, with the flow outside the boundary layer therefore being assumed to be inviscid, and irrotational. In potential flow, the flow velocity is assumed to be derivable from a scalar velocity potential Φ, which is in turn defined as the gradient of the local velocity.

$$\vec{U} = \vec{\nabla}\Phi \qquad [1]$$

The full-potential equation, is then defined in terms of the unknown scalar velocity potential Φ in the following manner[7].

$$\frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} + \frac{\partial(\rho w)}{\partial z} = 0 \qquad [2]$$

Because of the inability to achieve an analytical solution to the full-potential flow problem with a set of reasonable boundary conditions, TranAir, as all other CFD codes, discretizes the solution field by a set of nonlinear equations, and solves those. The solution field is split into a *global grid* of rectangular boxes (cells), each of which is then refined hierarchically to approach a solution with an error level specified by the user.

TRANAIR solves a set of nonlinear equations which approximate the full-potential equation using the Newton method at each corner point of every cell of the grid. In applying the Newton method, each problem is linearized and the linear problem is solved during each iterative step. The linear problem is also solved iteratively using the GMRES method.[8]

4.3. Finite Computational Domain

It is important to note that unlike many other codes, the rectangular grid elements penetrate the boundary of the geometry – making initial global grid definition independent of the configuration surface. During the solution process, the global grid is refined by subdividing a single given cell into eight equal sub-cells. Global grid refinement can be controlled via a variety of methods, allowing for arbitrary local hierarchical refinement.[9] The computational domain is restricted to the global grid, which can be terminated at a very near distance from the computational geometry (as near as one unrefined global grid cell). The TranAir Theory Document[10] justifies and explains this approach in great detail.

4.4. Boundary Layer Solution

In the TranAir implementation, the boundary layer solver is coupled to the inviscid solver, and the solution information is shared between the two codes. Two boundary layer codes, A411 and ISES, are incorporated in TranAir, the first a "loosely coupled", and the second a "closely coupled" boundary layer code. A411 passes information back and forth between the viscous and inviscid programs at each Newton step in the solution process, while the ISES code actually combines the equations for viscous and inviscid flows, and solves them all together. This coupling method provides

the ISES code with improved solution convergence and allows it to achieve solutions for mildly separated flows, in addition to attached flows. Because ISES code was chosen to be used for boundary layer analysis in this study, the description will focus on this code.

The ISES 2-d boundary layer code was developed by Mark Drela and Michael B. Giles[11]; the newest version of it is marketed under the name MSES. A streamline-based Euler discretization and a two-equation boundary layer formulation are coupled through the displacement thickness and solved simultaneously by a full Newton method.[12] ISES implements advanced transition and separation calculations with the capability to model laminar separation bubbles and limited zones of turbulent separation.

At its core, ISES uses a simplified form of the Prandtl boundary layer equations of continuity and conservation of momentum and energy.

Continuity: $$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho u) + \frac{\partial}{\partial y}(\rho v) = 0 \quad [3]$$

Momentum: $$\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) = -\frac{\partial p_e}{\partial x} + \frac{\partial}{dy}\left(\mu\frac{\partial u}{\partial y}\right) \quad [4]$$

Energy: $$\rho\left(\frac{\partial h}{\partial t} + u\frac{\partial h}{\partial x} + v\frac{\partial h}{\partial y}\right) = \frac{\partial p_e}{\partial t} + u\frac{\partial p_e}{\partial x} + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + \mu\left(\frac{\partial u}{\partial y}\right)^2 \quad [5]$$

Assuming a perfect gas in steady flow and eliminating the velocity terms in the equations, integration can be performed outward from the wall to infinity. Finally, neglecting cross-stream pressure variation and considering only adiabatic freestreams, the final form of equations used in the program are achieved.[13]

$$\frac{d\theta}{d\xi} + (H + 2 - M_e^{\,2})\frac{\theta}{u_e}\frac{du_e}{d\xi} = \frac{Cf}{2} \quad [6]$$

$$\theta\frac{dH^*}{d\xi} + (2H^{**} + H^*(1-H))\frac{\theta}{u_e}\frac{du_e}{d\xi} = 2C_D - H^*\frac{Cf}{2} \quad [7]$$

The boundary layer is described with a two-equation [6,7] lagged dissipation integral BL formulation and an envelope e^n transition criterion. The alternative displacement body model is used[14] – limiting the effects of the boundary layer to displacing the inviscid flow away from the physical body constraints. The boundary layer solution is calculated for a 2-d "rib" that is user-defined on top of the configuration geometry. In the TranAir application, the 3-d solution is then interpolated from the 2-d ribs onto either an infinite swept wing with taper, or a body of revolution.

4.5. Network Types

TranAir uses a rectangular gridding system, with the surface network data being input in a set of ordered arrays of surface corner points known as *networks*. Although the networks, being composed of rectangular *panels*, are locally flat, the resultant configuration boundary is a good approximation to that present in the original geometry. The panels serve the purpose of limiting the region of integration for the Bateman variational functional. No fundamental unknowns (such as the doublets or sources in linear panel networks) are associated with the panels except for wakes, where a discrete set of doublet unknowns, μ, are defined at various corner points of the wakes.[15]

A number of boundary conditions can be specified for each network that composes the geometry. All of the geometry in this study was defined using Impermeable Surface Networks (kt=1) subject to the following boundary conditions. [8,9]

$$\vec{W}_u \cdot \hat{n} = 0. \quad [8]$$

$$\varphi_l = 0. \quad [9]$$

Data is gathered at the plane of interest within the intake using a Sample Network (kt=6), a type that is used to generate flow field data on a set of points other than on the configuration surface. This type of a network only evaluates the flow solution, but does not affect it.

Wakes must extend downstream from lifting bodies. These surfaces allow non-zero circulation in potential flow and can be thought of as thin sheets of concentrated vorticity.[16] Unlike the surface boundary networks, the wake networks have a discrete set of doublet unknowns defined at their corner points. Viscous wakes directly behind all the geometry components were simulated using Viscous Wake Networks (kt=14), which allow for the simulation of thickness through transpiration. The value of transpiration is automatically calculated by the code, while the leading edge variation of the doublet strength is determined from a Kutta condition (or doublet matching condition to an abutting wake network).[17] The following boundary conditions are imposed at all wake points.

$$\vec{W}_u \cdot \hat{n} - \vec{W}_t \cdot \hat{n} = bet \qquad [10]$$

bet = Non-dimensional mass flow (ρ*V)

$$\varphi_u - \varphi_l - \mu = 0. \quad [11]$$

The Standard Wake networks (kt=18) are defined in a similar manner, with the only difference in the boundary conditions being the non-dimensional mass flow in equation [10] set to zero. A standard wake network connects to a viscous wake network and must exit the computational grid in the downstream direction.

5. Geometry Generation

The full-scale geometry is designed to represent an engine installation on a notional business jet, with all the relevant scale parameters being derived from this aircraft. The aircraft characteristics are obtained from an evaluation of a range[18] of Light Business Jet Aircraft, and allow for a real-world demonstration of effects of engine installation on an aircraft and its mission. An isometric view and some important characteristics of this notional aircraft are both presented in Table 1 and Figure 5.

Table 1 – Notional Aircraft Parameters

Wing Span	41.7	ft		Takeoff Gross Weight	15,000	lb
Wing Area	217.4	ft^2		Wing Loading	69	lb/ft^2
C_L Cruise	0.4	-		Limit Load	4.5	G
Cruise Speed	0.75	Mach		Cruise Altitude	35,000	ft
Cruise Reynolds #	9,400,000	-		Baseline L/D Ratio	14.0	-
Wing/Total Drag	50	%				

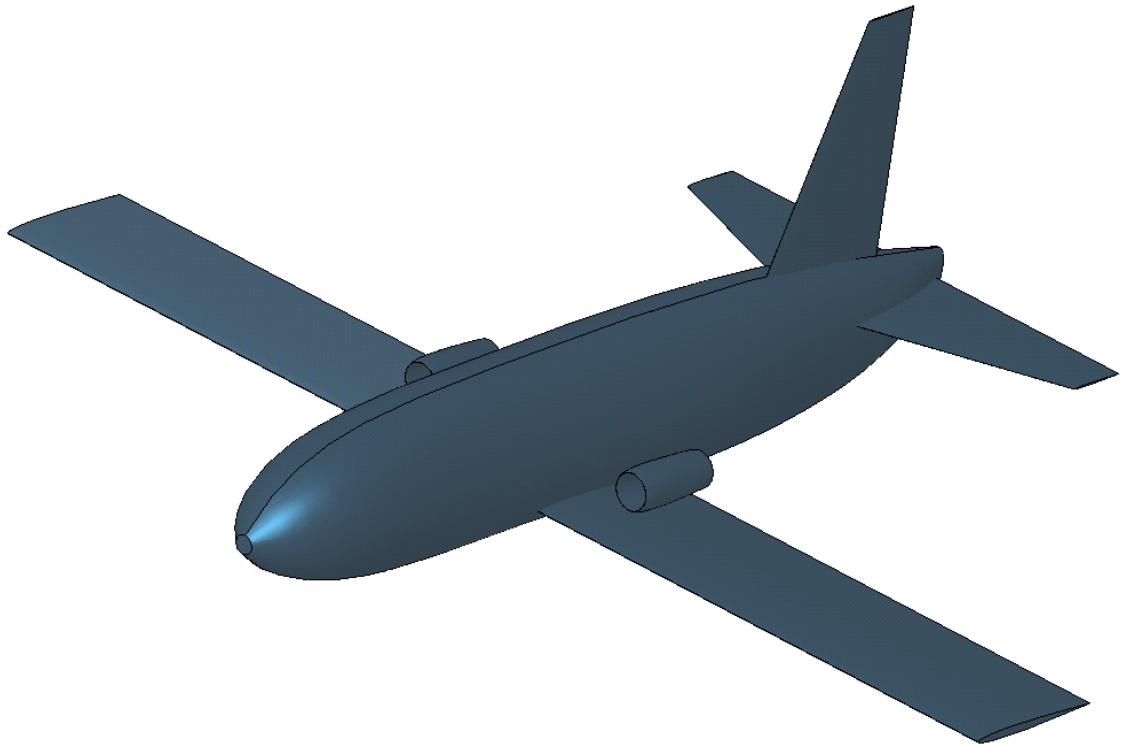


Figure 5 – Notional Aircraft Configuration

With the exception of the engine installation, the aircraft shown has a configuration similar to that of many current business jets. Although not considered during the aerodynamic study, the leading and trailing edges of the actual aircraft would likely feature moderate sweep, mainly for structural reasons.

The engine intake is of a simple "pitot inlet" configuration. This type of an inlet is simple, lightweight and inexpensive. The pitot inlet/nacelle arrangement approaches a pressure recovery ratio of unity at transonic speeds, and is the most common method of housing a high-bypass turbofan. Because the notional jet design will be limited to transonic speeds, this type of an inlet appears to be an ideal compromise in terms of performance, price and simplicity.[19]

5.1. Geometry Description

The baseline geometry created in AGPS is comprised of an axisymmetric nacelle separated from an unswept wing surface. This geometry is comprised of 5 surface networks and 3 viscous wake networks, which can all be seen in Figure 6.

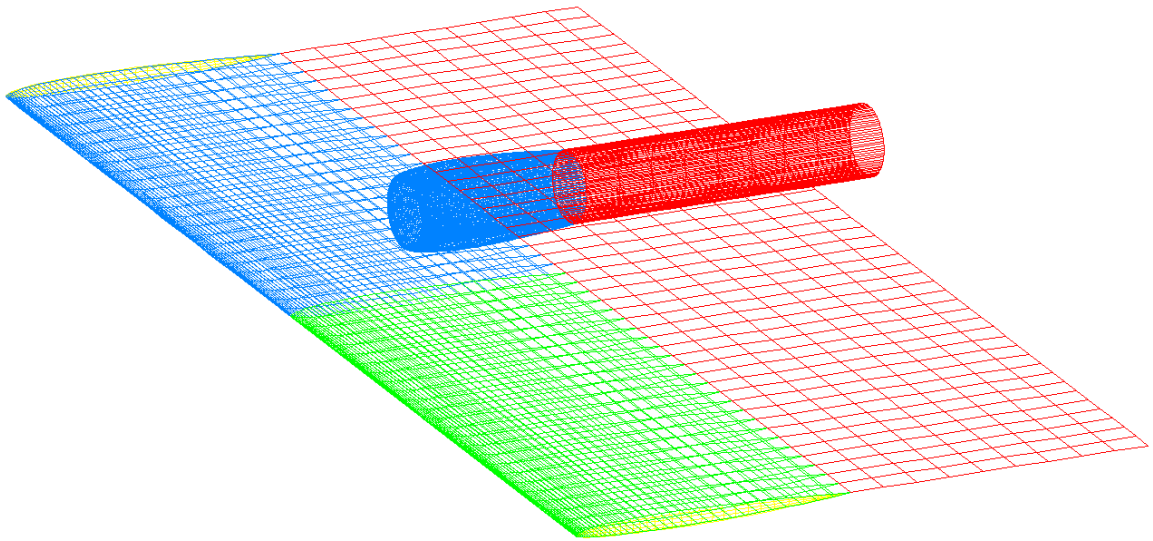


Figure 6 – Sample Geometry

A variety of different types of networks are available for use in order to allow for the representation of the different required types of boundary conditions. In the Figure 6 above, the viscous wake networks are shown in red. The single type-1 wraparound network making up the nacelle is outlined in blue. The two rounded wing cap networks are outlined in yellow. The two networks making up the left and right wing sections are shown in green and blue, respectively.

To obtain a clearer view of the wake networks, they are displayed individually in Figure 7 below.

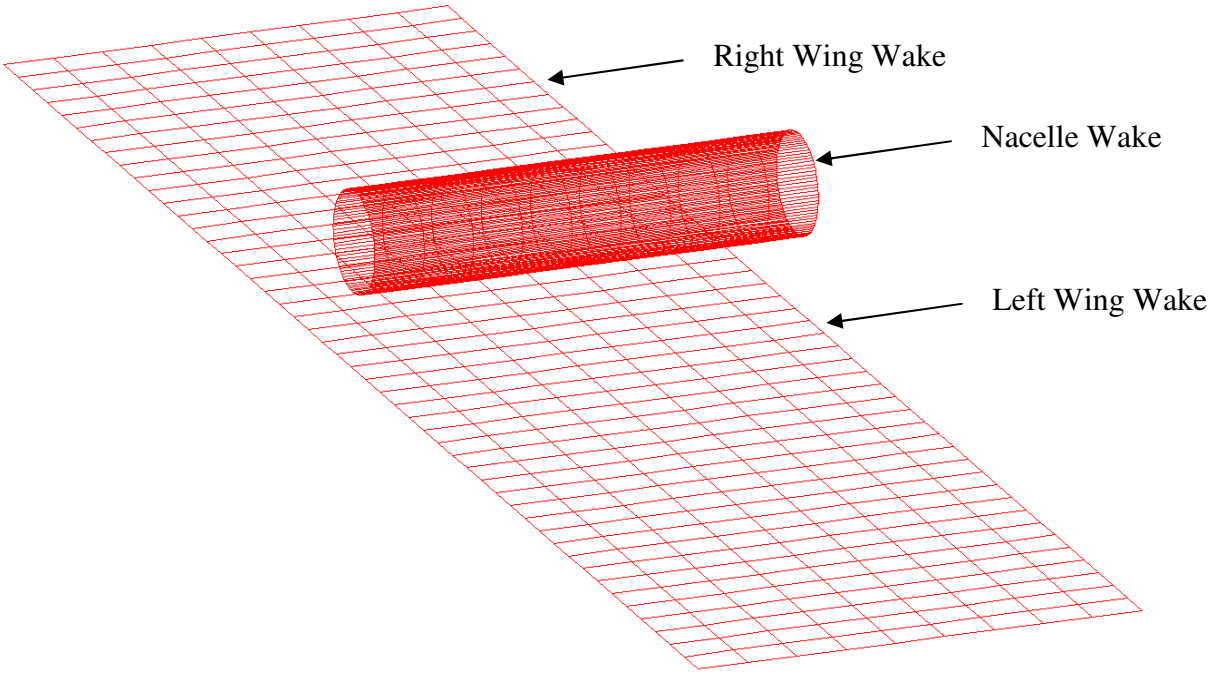


Figure 7 – User-specified wakes

The two airfoils used to create the nacelle and the wing are shown in Figure 8 below. Both are 5.5% thick, with a modified Grumman/Gulfstream GIII Transonic Airfoil used

on the wing, and the nacelle being created using a modified airfoil generated to minimize the convergent-divergent characteristics of the nacelle throat.

Wing Airfoil

Nacelle Airfoil

Figure 8 – Airfoils Used

When executing a run with this geometry, TranAir code automatically generates 3 additional networks in order to represent the non-viscous trailing wakes. These 3 additional wakes are automatically attached to the downstream edges of the viscous wakes mentioned previously. The user-generated surface and wake networks are specified in the .poi input file. The networks used in the geometry, their names and gridding type implemented are all listed in Table 2.

Table 2 – Networks Used

	Grid Size (M:N)	Network Names	Gridding type (Row:Column)
Nacelle	101:100	ng	uniform:custom
Wing Surfaces	81:25	wrg, wlg	custom:uniform
Wing Caps	41:7	rwcr, rwcl	custom:uniform
Viscous Wakes	10:100, 10:25, 10:25	ngvw, wrvw, wlvw	uniform:uniform

The relevant dimensional parameters of the geometry used in the TranAir study are presented in Figure 9 below. To conduct the study at the full-scale wing Reynolds number of 10,000,000, a Reynolds number of 4,000,000 per unit length of TranAir geometry is specified in the boundary layer input section of the input file. The scaling between the CFD and real-life geometry is accomplished using the ratio of 2.085 ft/unit.

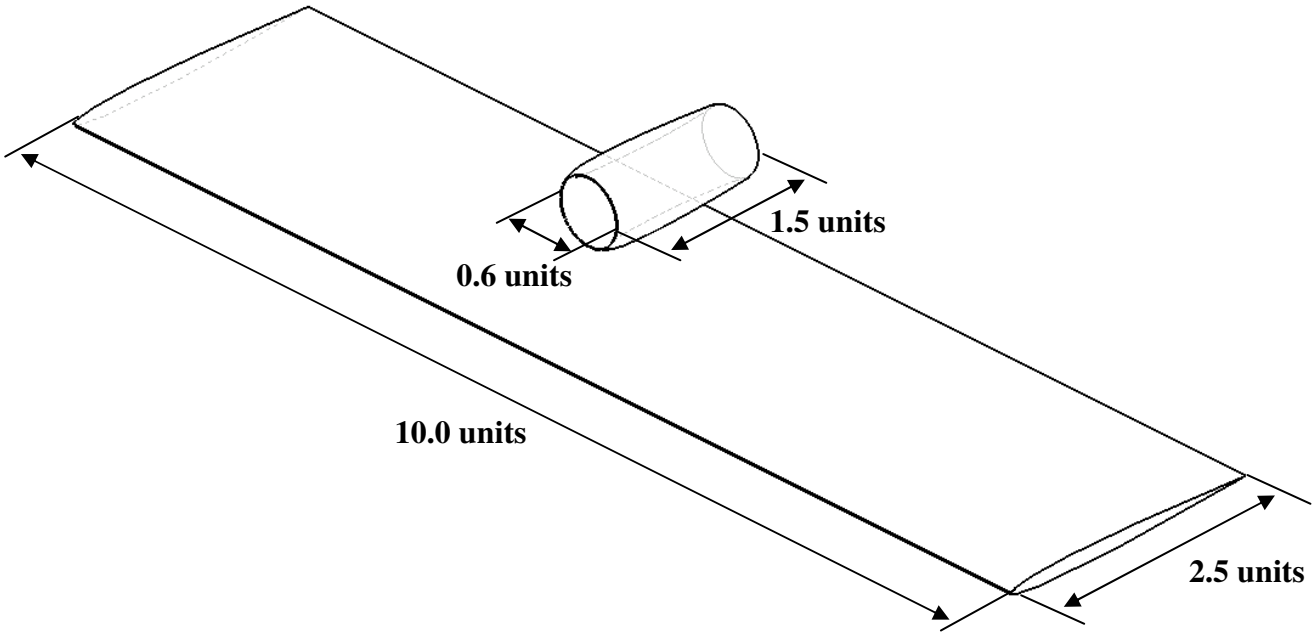


Figure 9 - Geometry Analyzed in TranAir

The set of issues encountered in analyzing "three-component geometry", consisting of a wing, nacelle and a pylon is described in full detail in section 9.2. A comparison of the results obtained from two possible geometry configurations at one pylon case is presented in the same section. As a result of these issues and in order to isolate the effects of nacelle placement, the geometry used in the majority of this study does not use a pylon. Instead, the nacelle and the wing are evaluated as separate geometric entities "flying" in close proximity. The decision to revert to a pylon-less version of the geometry was taken relatively late in the study, and as a result the Geometry Generation code has a number of capabilities that ended up not being used. Although not being necessary in this particular study, the capability to successfully create more complicated geometries is present and is discussed in the description of code operation below. It is hoped that this information will fully describe the work performed, and help to provide some guidelines for further usability of this AGPS code.

5.2. Geometry Generation Code Operation – 3-component Geometry

In order to create this geometry automatically, a "journal file" – which is basically a script, had to be created. This script is presented in Appendix D. The script is extensively annotated, and the reader should refer to that section of the report for a detailed description of the code functionality.

Initially a set of airfoils is read in from the data directory. From these, the basic shapes of the wing, the pylon and the revolved nacelle are generated. The surfaces making up these geometrical entities are intersected in order to remove areas of mutual interference. After the surfaces are all defined, each is gridded. Next, the geometry creation code creates the wake networks that trail the geometry and the data gather networks located at points of interest in the flow. At the end of the code operation, the resultant geometry is displayed, and upon user confirmation, all of the networks are output in a formatted .poi file.

The code requires that the abutting network edges have exact panel edge points which match along the common network edge, or panel edge points which are on the straight line between the exact points.[20] A relatively complicated process is involved in assuring this, and programming the logic to consistently produce the geometries required for this study necessitated a significant amount of time and effort. Panel abutment is accomplished by modifying an edge of network A by "borrowing" edge points from neighboring network B. This operation becomes more difficult in places of multiple networks abutting a single network, such as at the wing surface-pylon joint. This method is used, for example, in creating the rounded wing caps which are discussed in further detail in section "CFD Operation - Rounded Wingtips".

To “trim off” the interfering surface edges, the parametric features of AGPS become extremely useful. For example, the initial nacelle surface created by the 360° rotation of the nacelle airfoil interferes with the top of the pylon. A “trim map” is created piece-by-piece in AGPS using traces of intersection curves of the two surfaces. This trim map, which can be seen in Figure 10, represents a parameterized view at the nacelle surface. After surface trimming is performed using this map, the interfering nacelle edges are removed.

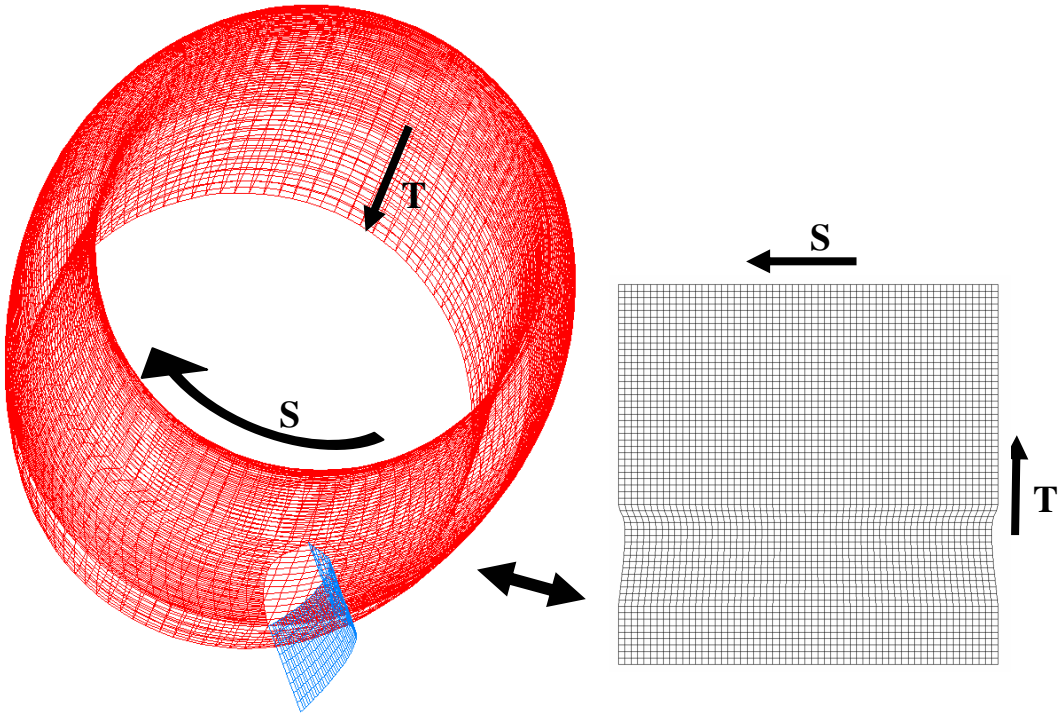


Figure 10 – Nacelle-pylon Intersection Trimming

In the process of creating the geometry for this project, the gridding capabilities present in the AGPS code were not sufficient, and some additional capability development had to take place. Although AGPS offers a number of pre-programmed “gridding rules”, custom ones had to be created for some of the components. A gridding rule is an ordered sequence of points which is parametrically applied to the target surface

during the gridding procedure, and which guides the creation of a mesh network on that surface. Because in AGPS each surface is parameterized in two directions (S and T), a maximum of two gridding rules can be used during the meshing of a given surface. For example, it was required that the corner point of the leading edge of the pylon was coincident with a point on both of the top wing surface networks. This was successfully accomplished by programming a custom gridding rule, an example of application of which on the left-top wing network can be seen in Figure 11.

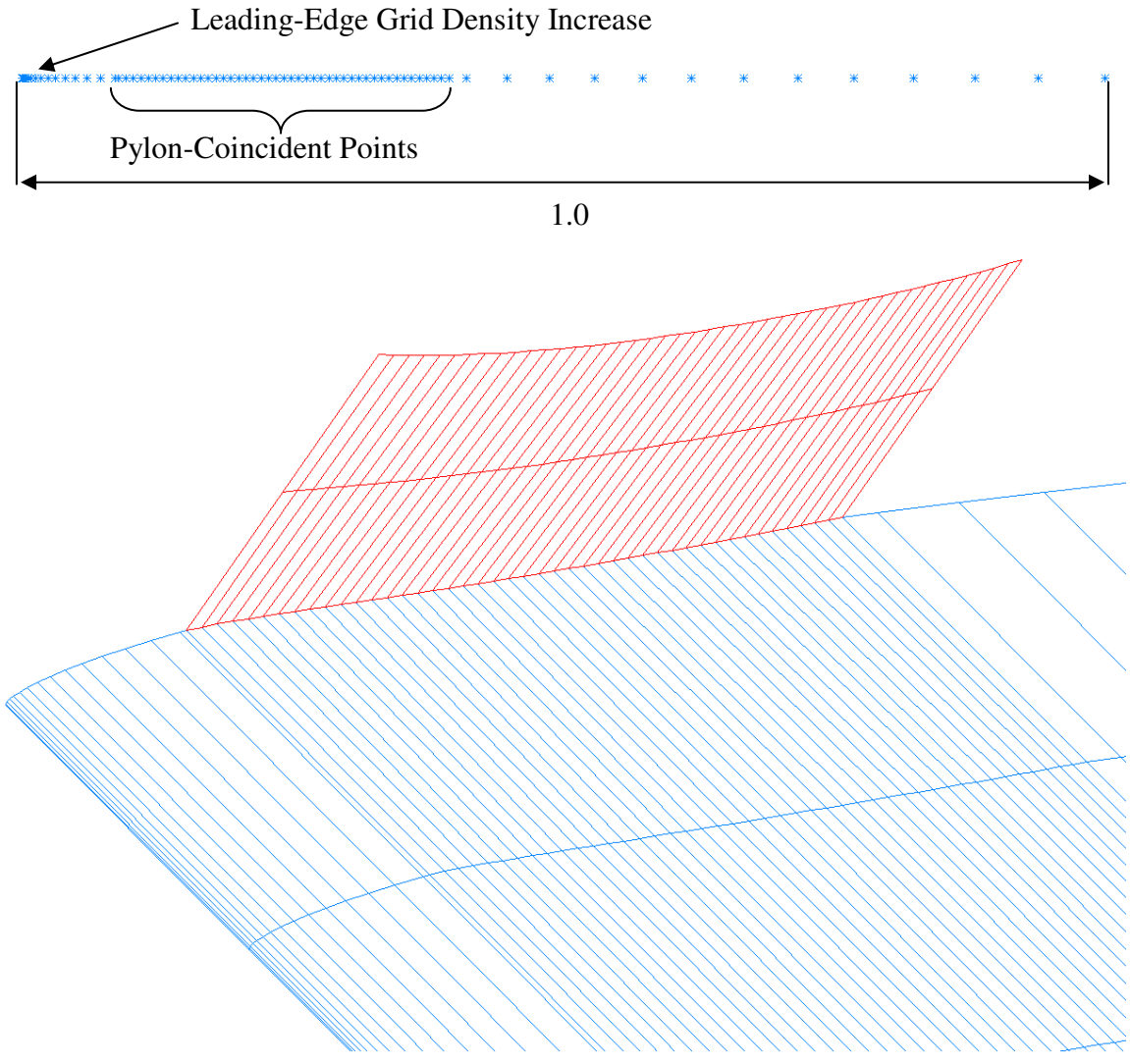


Figure 11 – Network Coincidence

By modifying the parametric gridding rule applied to the wing top surface during the gridding procedure, it was possible to create coincident mesh points at the intersection of the wing and pylon networks, as shown in the picture above. Obviously, with the pylon moving with the relation to the wing from case to case, a single gridding rule would not be sufficient, and a new one had to be automatically created during each run. A similar gridding rule was created for the nacelle geometry.

6. CFD Operation

6.1. Global Grid, Refinement and Convergence

The case geometry is contained within an initial "global grid" box with the dimensions listed in Table 3.

Table 3 – Global Grid Initial Dimensions

Dimension	Start, end	Number of cells	Cell Dimension
X	-2.5 9.0	28	0.4107
Y	-2.0 13.0	21	0.7143
Z	-2.5 7.5	16	0.625

The initial global grid box superimposed on the solution geometry can be seen in Appendix C. It should be noted that the limits of the global grid are much further away from the geometry than 1 unrefined grid cell limitation of the code mentioned previously. The reason for this is that although the global grid can be very close to the geometry in sub-sonic flow studies, in the transonic regime it must also capture all the shocks that occur around the geometry.

6.2. Adaptive Refinement

A process known as Adaptive Grid Refinement modifies the initial evenly-dense global grid to achieve higher grid density in all regions of flow discontinuities. This process is extremely important, affecting both the eventual solution convergence and speed of solution. The process of adaptive grid refinement splits each grid cell selected for refinement into 8 equivalent sub-cells. Because the cell is split evenly, the "aspect ratio" of the resultant sub-cells stays constant. For this reason the code requires that the

individual cell dimensions of the initial grid in the Y and Z axis be equal and approximately 1.5 times the dimension in the X axis.

Adaptive Gridding Procedures used in the TranAir code refines and de-refines user-specified fractions of grid cells. Cells are chosen to be refined and de-refined according to the error level criteria specified by the user in the input file. These error criteria are compared to the weighted local error indicators generated by the program at each grid box.

Ten cycles of adaptive refinement are used during the run, with a variety of parameters specified for each cycle. These are shown below in Table 4.

Table 4 – Adaptive Grid Refinement Parameters

Cycle Number	Grid Box Upper Limit	Volume LBO Upper/Lower Limit	Surface LBO Upper/Lower Limit
1	9,000	none	none
2	10,000	none	none
3	22,000	1.05/0.15	89/0.15
4	24,200	1.05/0.15	89/0.15
5	50,000	0.5/0.077	39/0.077
6	55,000	0.5/0.077	39/0.077
7	110,000	0.5/0.038	20/0.038
8	120,000	0.5/0.038	20/0.038
9	240,000	0.5/0.0193	20/0.0193
10	270,000	0.5/0.0193	20/0.0193

Two “Local Boxes of Interest” or LBOs are specified around the geometry in order to achieve higher local grid density at the wing surface and the areas of complicated flow behaviors. The LBO is refined at every other step of the adaptive refinement procedure.

It is important to refine the global grid to a sufficient degree, and in correct increments. The minimum cell size used in the final refinement step is 0.0193, which, as suggested by the TranAir manual, is less than 1% of wing chord. Low levels of grid

refinement can contribute to two undesirable results. Firstly, the solution may have difficulty converging to the level specified by the user. Secondly, the results may not be as precise as those possible with higher level of convergence because some complex flow behaviors, such as supersonic shocks may not be captured fully. The adaptive refinement process implemented in this case is robust enough to provide convergence in all run cases, while ensuring quick run times and accurate results.

6.3. TranAir Usability

TranAir has just recently transitioned from a company-proprietary to a commercially-available CFD tool and therefore there is little public domain information regarding its use. Like many other CFD tools, the interface and operation of the code require an extremely high degree of familiarity from the user. Therefore, the process of evaluation of the subject geometry in the TranAir CFD code is one that required a considerable amount of mastery of the code. A variety of issues not mentioned in the software manual had to be resolved with help from supplementary sources and experimentation. In the process of learning the code it became the desire of the author to dedicate a portion of this thesis to providing public-domain guidance to future TranAir users. A variety of lessons learned in the process of working on this thesis and suggestions for future users are included in Appendix B.

7. Structural Constraint

Data gathered during the initial CFD runs showed that increased nacelle separation from the wing surface increase overall efficiency. However, it was also obvious that the structure used in the mounting of the engine will be the lightest at the lowest separation of the nacelle from the wing surface. A balance had to be found between these two drivers, so for this reason a structural constraint that accurately estimated the weight of structure used to mount an engine in a given location had to be developed and implemented.

An important concern was to make the constraint be able to represent the applicable structural trends accurately, but to not complicate it to the point where it would end up being limited in use. It was assumed that the major structural difference between different nacelle locations would be limited to the pylon. Because a pylon is extremely structurally complicated, it would not be feasible to perform a complete structural design at each point of the MDO iteration. Instead, a number of simplified approaches of quantifying pylon weight, both analytical and empirical, were attempted.

7.1. Analytical Method

Having defined the chord-wise locations of the two wing spars, and represented the structural makeup of the pylon by two hollow beams of circular cross-section, weight trendlines for the pylon structure could be derived. A plane view showing a section of the pylon and nacelle can be seen in Figure 12.

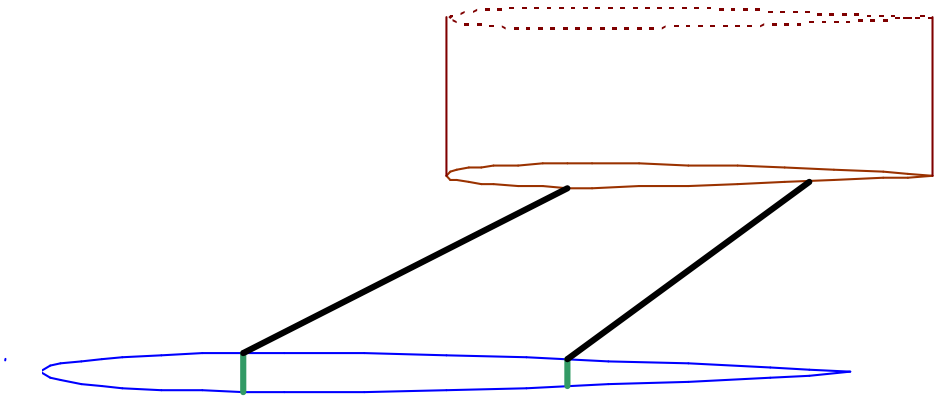


Figure 12 – Nacelle Mounting

The structure of the pylon is designed to resist two of the major loads generated by an engine during flight. The first load is the thrust load generated by the engine during maximum-thrust operation. The second load is the downward force of acceleration generated by the engine during a limit G-load maneuver. With the pylon attachment points defined at 25% and 75% points on the nacelle, it is approximated that the g-load force is equally split between the two pylons, acting in the vertical direction at the mounting point. The engine thrust force arm is in turn defined as the distance from the nacelle centerline to the wing chord line. From the beam bending moments thus generated, two tubes of a wall/radius ratio of 15 are sized, and their weight calculated. The entire procedure is automated using Microsoft Visual Basic, and is performed at every point of the MDO run. An example calculation for the pylon structure shown in Figure 12 is presented in Table 5.

Table 5 – Structural Sizing

Limit Load	4.5	G	Design Stress	60000	psi
Engine Weight	700	lbf			
Gravity X1	4.69	ft	Mom. Gravity 1	88670.75	lb-in
Gravity X2	4.17	ft	Mom. Gravity 2	78818.45	lb-in
Thrust Arm	4.07	ft	Mom. Thrust	60929.48	lb-in
Front Inner Rad.	1.87	in	Rear Inner Rad.	1.77	in
Front Outer Rad.	2.00	in	Rear Outer Rad.	1.90	in
Mom. Of Inertia	3.0	in^4	Mom. Of Inertia	2.5	in^4
Stress Front Gravity	56,567	psi	volume	203.40	in3
Stress Front Thrust	38,870	psi	weight	57.56	lb
Stress Rear Gravity	58,646	psi			
Stress Rear Thrust	45,336	psi			

The pylon structure designed for a production vehicle will obviously be much more complex than that modeled in this study. However, this relatively simple model allows for accurate simulation of the weight-to-strength relationship trends, thus permitting evaluation of the structural penalties of different engine installation locations. Although relatively simple, an issue is inherent to this method. Because the moment generated by a landing G-load is the sizing driver, the structure was significantly undersized in nacelle locations with highly vertical pylons, since these reduced the engine moment arm to practically zero.

7.2. Empirical Method

This method was developed in an attempt to use the weight data available for aircraft with similar pylon engine installations. The underlying hypothesis in creating this method was that the weight of a pylon installation was mainly driven by the weight of the engine it supports, and by the length of the pylon. By correlating the structural weight of

the “nacelle group” from a MIL-1374 weight breakdown form[21] to the two pylon weight drivers mentioned above, Figure 13 was obtained.

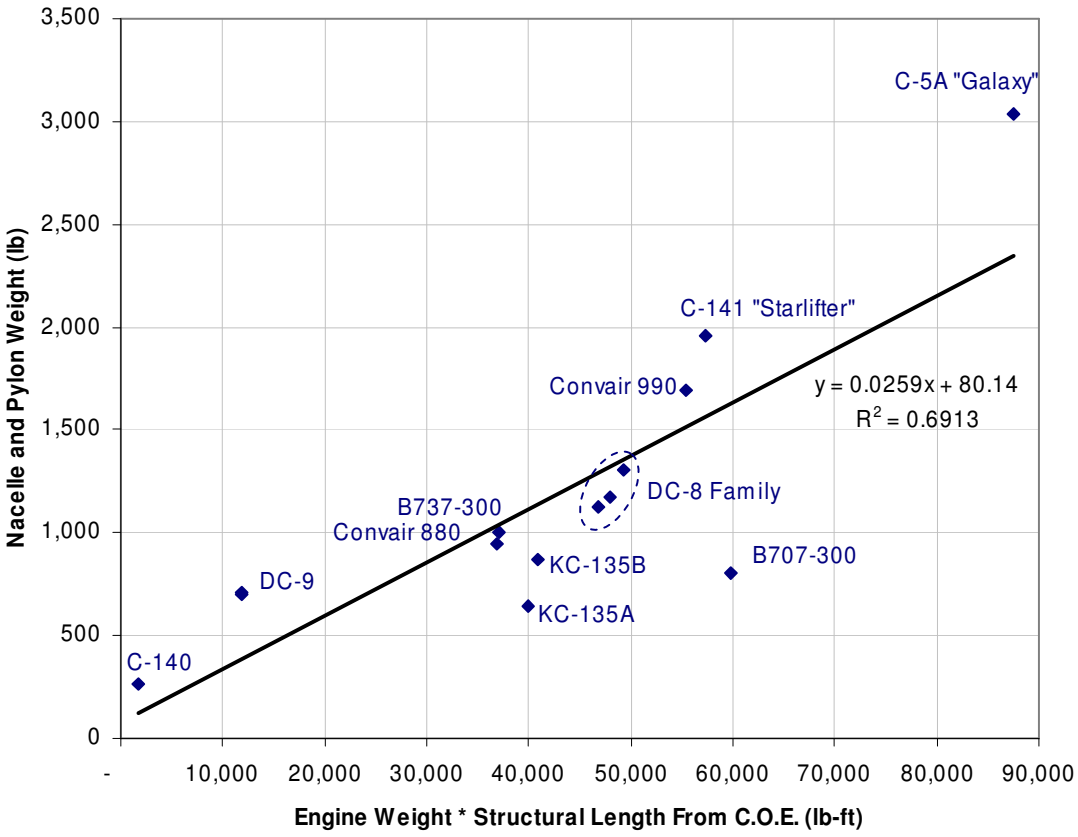


Figure 13 – Pylon Weight Trendline

This trendline gives an added benefit of accounting for nacelle structure and other components necessary. The aircraft included in the study all have similar cruise speeds and mission requirements as the hypothetical aircraft. Aircraft with special mission characteristics that may impact the weight of the pylon installation, such as, for example, the A-10 ground attack aircraft, have not been used. The linear trendline drawn through the data points was used at each step of the MDO process to establish the weight of the nacelle + pylon combination on the hypothetical aircraft configuration.

8. Multi Disciplinary Optimization Tool

The Multi Disciplinary Optimization (MDO) tool allowed for an optimizational synthesis of the results obtained during the CFD and structural studies, and showed the impact of various nacelle mounting options on an aircraft's performance during a given mission. A relatively simple flight simulation that limited its scope to the long-distance cruise component of an aircraft's mission was conducted, allowing the user to evaluate the amount of fuel burned by the aircraft. The tool was created in Microsoft Excel, and the important assumptions used in the construction of this tool are listed below.

The particular mission the hypothetical aircraft is being optimized for is 2,500nmi long, with the in-cruise specific fuel consumption of 0.5 lb/lb/hour being assumed. The empty weight of the aircraft is calculated as a combination of structural empty weight of 7,500lbs, added to the weight of the engine and installation. The Empirical Method described in section 7.2 was selected to calculate the weight of the pylon. An engine similar in size to the Williams FJ-44 is assumed, with a dry weight of 445lbf used.

The CFD results map obtained earlier is input in a tabulated format into the MDO spreadsheet, where 3-d parabolic interpolation enabled by the XlXtrFun[22] Microsoft Excel plug-in allows the L/D value to be obtained at any hypothetical nacelle location. With the aircraft empty weight and L/D ratio determined, thrust required and fuel burn during cruise can be calculated. By subtracting total fuel used during the mission and aircraft empty weight from the takeoff gross weight, the maximum possible payload weight is derived. This payload weight is used as a metric of the aircraft's performance, since it can be used to either extend the flight range or to carry extra income-generating payload.

The L/D ratio values are compared without taking into account the possible changes to the aircraft's fuselage and empennage. It is be assumed that the influence of these components will not vary between different nacelle geometries evaluated, and as such will decrease the overall L/D ratio of the all geometries by the same amount, therefore not affecting the result of the trade-off performed. The additional drag somewhat "dampens" the wing's aerodynamic benefit to the aircraft, and is compensated for by the assumption that wing drag constitutes only 50% of overall aircraft drag.

It should be mentioned that the MDO tool does not attempt to resize the aircraft in order to take the maximum advantage of a reduced fuel burn. Instead, with all aircraft configurations flown at the same overall gross takeoff weight, an assumption is made that a lower-fuel burn aircraft will carry extra payload in the space afforded to it by its lower structure weight. A full aircraft sizing process would require for an established relationship between engine thrust and weight in order to allow for weight estimation of engines of varied required thrust level. This relationship would introduce another level of uncertainty to the results of the study, and was judged to be excessive. In the current MDO implementation the gross takeoff weight of the aircraft stays constant, and therefore the mission chosen will not affect the optimum nacelle location.

By iterating through the applicable study range, it is possible to obtain the empty weight, takeoff gross, and in-mission fuel consumption for all aircraft configurations of interest. This, in turn allows finding an aircraft configuration in which the benefit of an aerodynamically efficient engine installation and the drawback of the higher-weight structure required to achieve such an installation are balanced, providing the aircraft with the optimum in-mission performance.

9. Results and discussion

9.1. Grid Independence Study

A grid independence study was performed in order to establish the correct degree of precision to use for the surface mesh networks. With the increase of surface mesh density the value of the flow solution approaches a limit – the goal of this study was to arrive at the lowest acceptable mesh density value, one producing results sufficiently close to the solution limit. This study was performed with a single nacelle position case – at height of 3.5 radii and 20% of the wing's chord.

The convergence limit of the data acquired was determined using a Richardson's extrapolation of the individually sampled data points. Because Richardson's extrapolation is most easily used with equally-spaced data points, the extrapolation was based on three solutions obtained at grid density levels of 25%, 50% and 100% of final density used.

Table 6 – Solution Convergence

Density	C_L	C_D	L/D
25%	0.4	0.022802	17.542
50%	0.4	0.024356	16.423
75%	0.4	0.024862	16.089
100%	0.4	0.025075	15.952
125%	0.4	0.025071	15.955

The observed order of convergence of the solutions was obtained using the equation [12].

$$p = \ln\left(\frac{f_3 - f_2}{f_2 - f_1}\right) / \ln(r) \qquad [12]$$

r – grid refinement ratio

The extrapolation was then performed using the two finest grids using equation [13], and an estimate of the L/D ratio at zero grid spacing obtained.

$$f_{h=0} \cong f_2 + \frac{(f_1 - f_2)r^p}{r^p - 1} \qquad [13]$$

The Grid Convergence Index (GCI) is a measure of the percentage the computed value is away from the value of the asymptotic numerical value. The GCI was calculated using a safety factor of 1.25 according to the methodology provided in Roache[23] and is listed alongside with the other results obtained in this study in Table 7 below.

Table 7 – Convergence Study Results

Result	Value
Order of Data Convergence	1.249
Value of L/D at zero grid spacing	**15.963**
Grid Convergence Index 1 (25%-50%)	6.188%
Grid Convergence Index 2 (50%-100%)	**2.503%**

The convergence of the solution, shown above to be sufficient by the GCI of 2.503% at the grid density used (100% density) can also be observed in Figure 14 below.

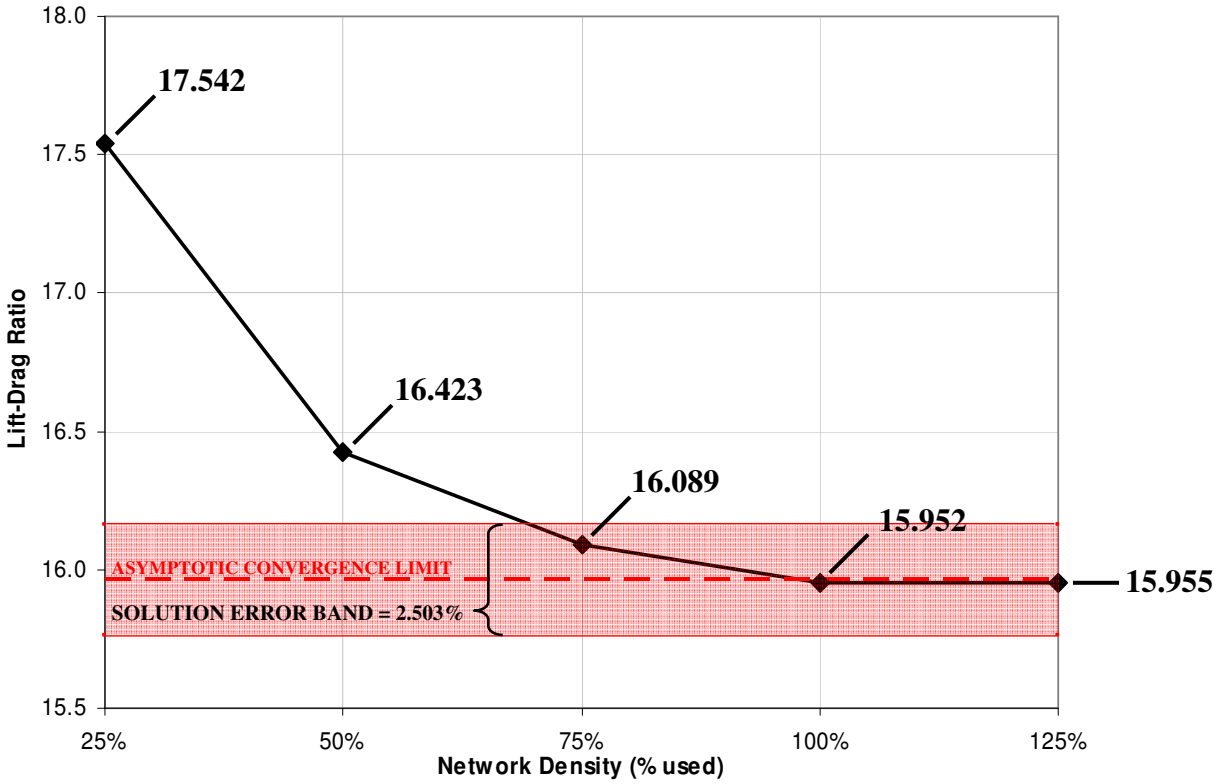


Figure 14 – Grid independence study

9.2. Geometry Issues

The initial geometry used in the study contained a pylon, which connected the nacelle and the wing. This geometry took a significant amount of work to create, with the main issues being the surface trimming, gridding and network coincidence discussed earlier. The 3-component (wing, pylon, nacelle) can be seen below in Figure 15.

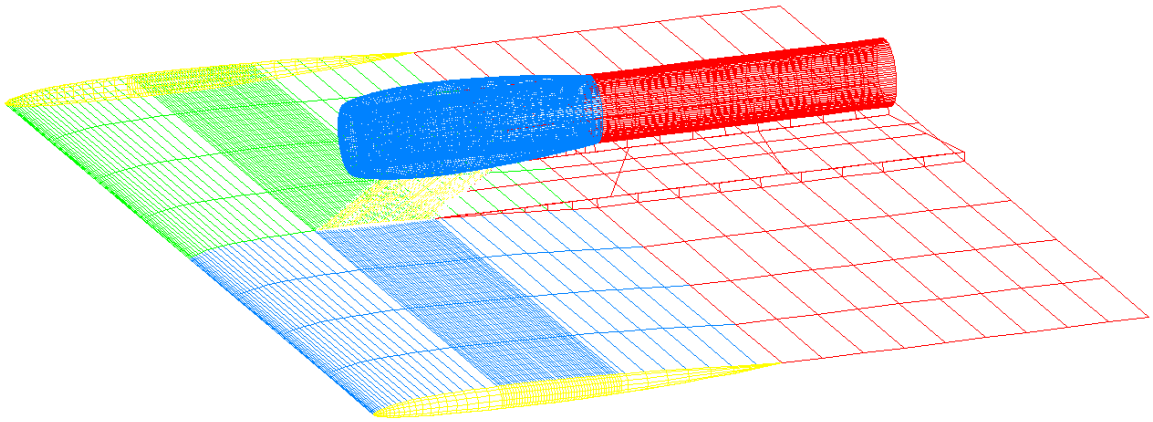


Figure 15 – Generalized geometry (3-component version)

A number of issues resulted when this geometry was studied in TranAir. Although the general trends portrayed in the L/D curves generated were similar to those obtained with the two-component geometry, a number of anomalies made it difficult to have any confidence in the results. The results achieved with the two geometries compared at one nacelle-height case (5.5 rad.) are compared in Figure 16 below.

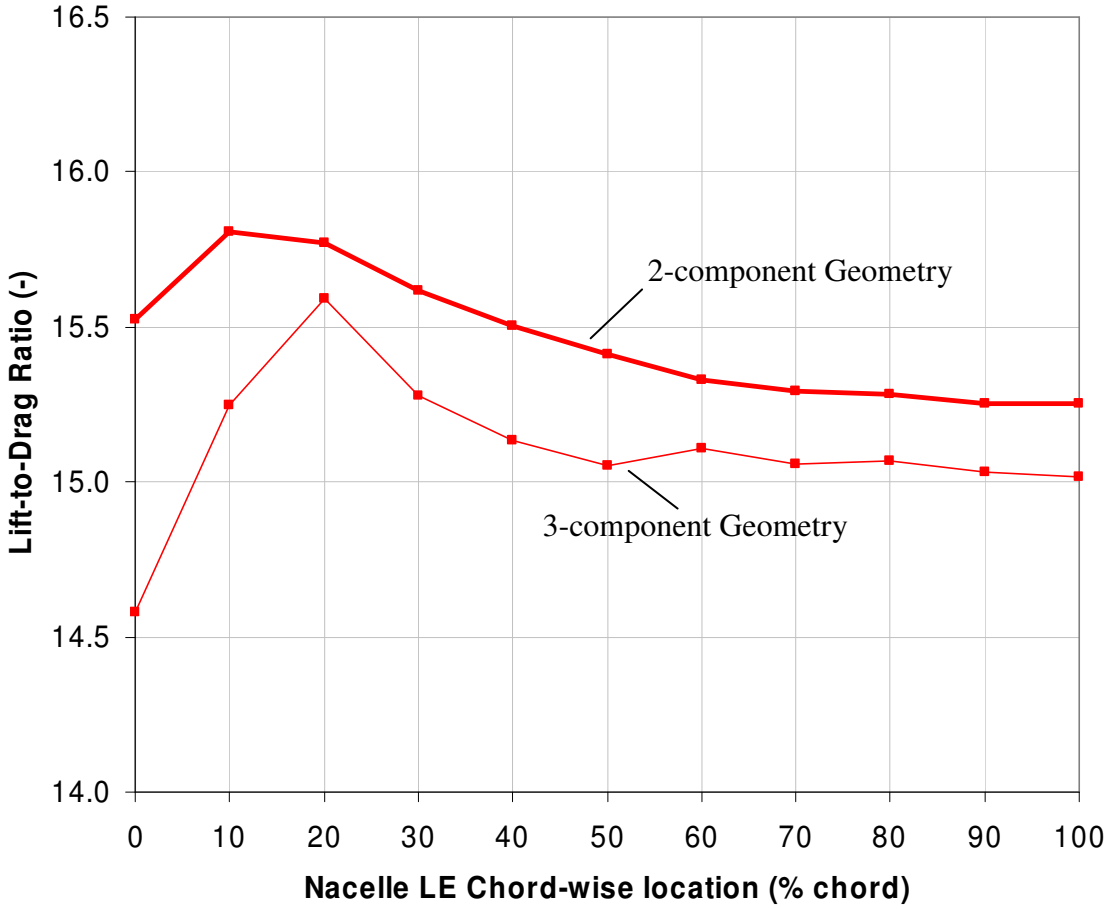


Figure 16 – Comparison between 2-component and 3-component geometries

It can be seen that the drag curves are showing a similar trendline - they both have an L/D peak at approximately 20% of the wing's chord, and come to a gentle decrease as the nacelle nears the trailing edge of the wing. The two curves can be seen to be vertically spaced, a difference that can be attributed to the drag-generating influence of the pylon. It is, however, obvious that the trends obtained with the three-component geometry are less gradual and consistent. Even more specious results were encountered in other runs, while the two-component geometry produced consistently smooth results. Because the purpose of the study was to address the effects generated by the nacelle, and it appeared that the pylon would only reduce the clarity of the results obtained, it was decided to remove the

pylon from the study until a reason for this faulty operation could be found.

Another benefit of reverting to the pylon-less geometry was that the gridding on the nacelle and the wing could now be kept constant across the range of geometries used in this study, instead of having zones of denser gridding corresponding to that used on the pylon. This allowed for much faster model generation, as the same boundary layer definitions could be used across all runs.

9.3. CFD Results

After performing CFD analysis on the established range of nacelle locations, values of Lift-to-Drag ratio for the wing-nacelle-pylon assembly were obtained. The plot showing the relationship of lift-to-drag to nacelle location can be seen in Figure 17.

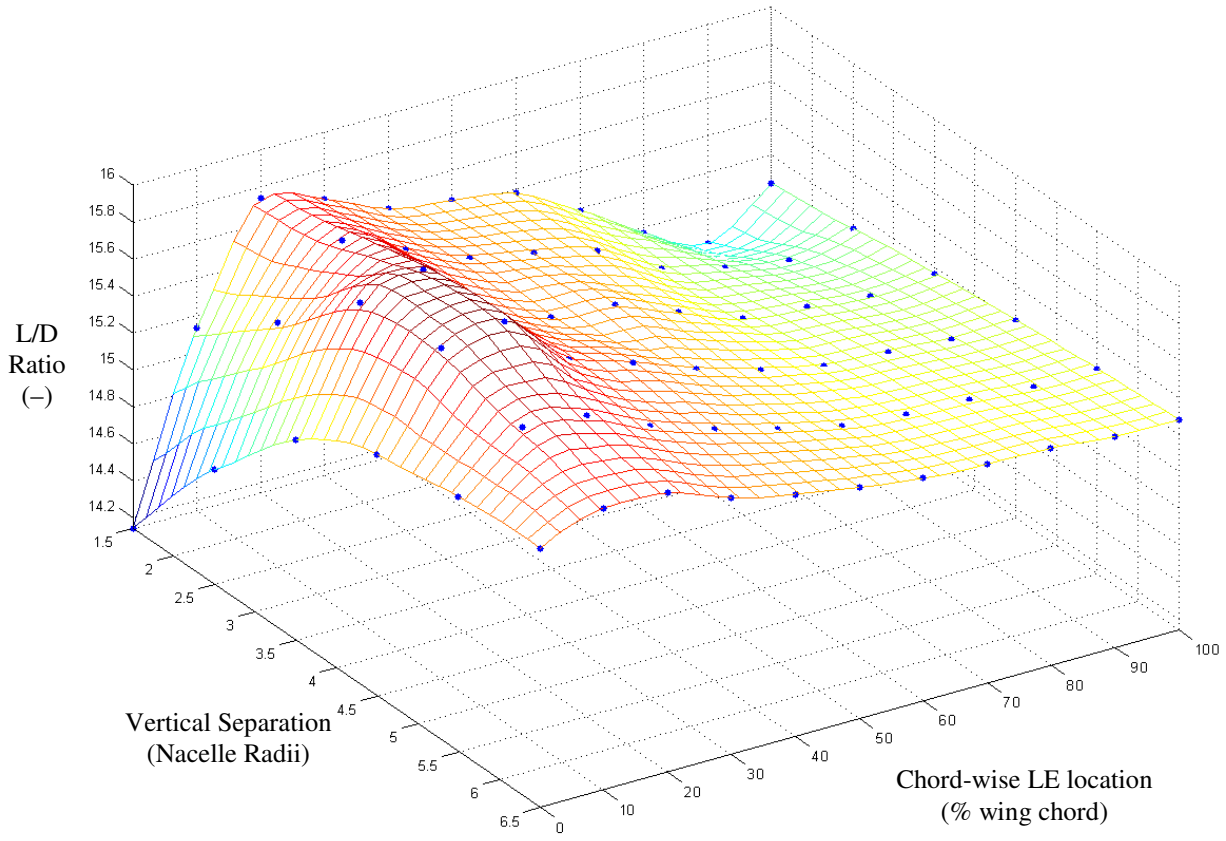


Figure 17 – Surface Map of CFD Results

A number of general trends can be noticed in the plot. Irrespective of nacelle mounting height, the L/D peak appears to correspond to an intake leading edge location at approximately 20% of wing chord. A 2-dimensional view of the L/D curves can be seen in Figure 18 below. Although the 3-dimensional L/D surface is more difficult to visualize in this view, the plot is much more useful for datapoint-acquisition purposes.

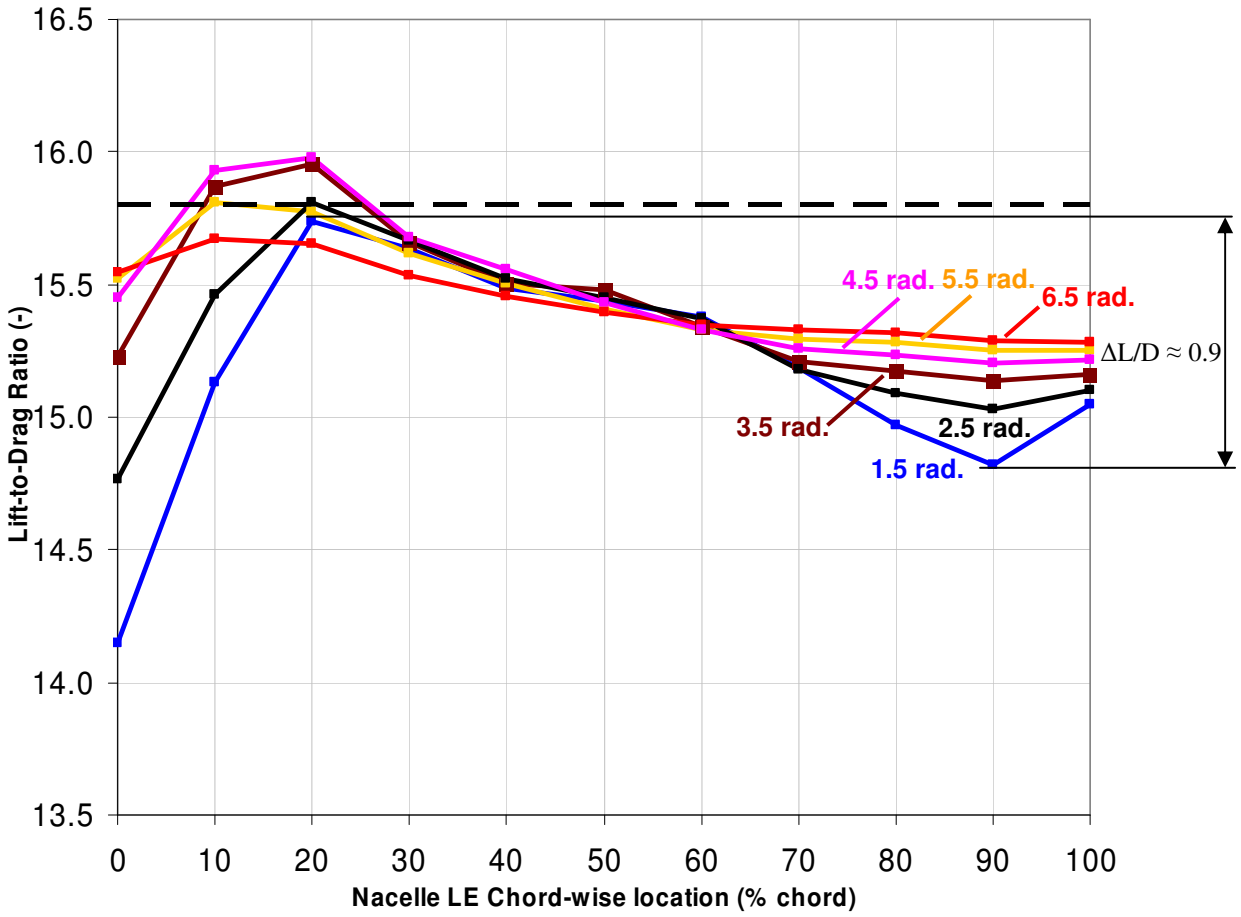


Figure 18 – 2-d Projection of CFD Results Map

The dashed horizontal line on the plot above indicates the L/D ratio of the wing tested alone, without the nacelle, at the prescribed C_L value. A number of the L/D points of the tested wing+nacelle geometries can be seen to actually exceed the lift-generating

efficiency of the clean wing, even with the addition of the nacelle. By careful positioning of the nacelle it is therefore possible to enhance the performance of the clean wing.

In the process of moving the nacelle chordwise between the two extremes at the leading and trailing edges of the wing, the L/D ratio of the entire installation can be seen to be significantly affected. The largest L/D ratio change caused by chord-wise movement of the nacelle, observed at a vertical separation of 1.5 radii is approximately 0.9. The L/D ratio change is the highest at this low separation, due to the high drag generated in the near-leading-edge nacelle locations, but the global peak L/D ratio is achieved at a higher vertical separation case.

A trend can be seen in the effects of nacelle vertical separation, defined in nacelle radii between axis of nacelle and wing chord line. The L/D ratio can be seen to peak significantly later with respect to this parameter, with the overall L/D peak corresponding to a nacelle axis vertical separation of 4.156 nacelle radii. With further vertical separation, the L/D ratio can be seen to smoothly decrease. From detailed flow visualizations addressed later in this study it is likely that this is caused by the diminishing of nacelle influence on the flow over the wing upper surface.

In order to better understand the trends portrayed in the L/D ratio surface, it will be useful to first analyze the local flow behaviors seen in two geometry cases differing in nacelle-wing surface separation.

The first case (50% chord, 1.75 radii), shown in Figure 19 displays a number of interesting flow characteristics. Most importantly, a zone of low-speed flow can be seen directly ahead of the intake face. This zone is generated by the upstream influence of the

nacelle body, and can be seen to decrease the speed of flow that would normally be seen at that point on the wing.

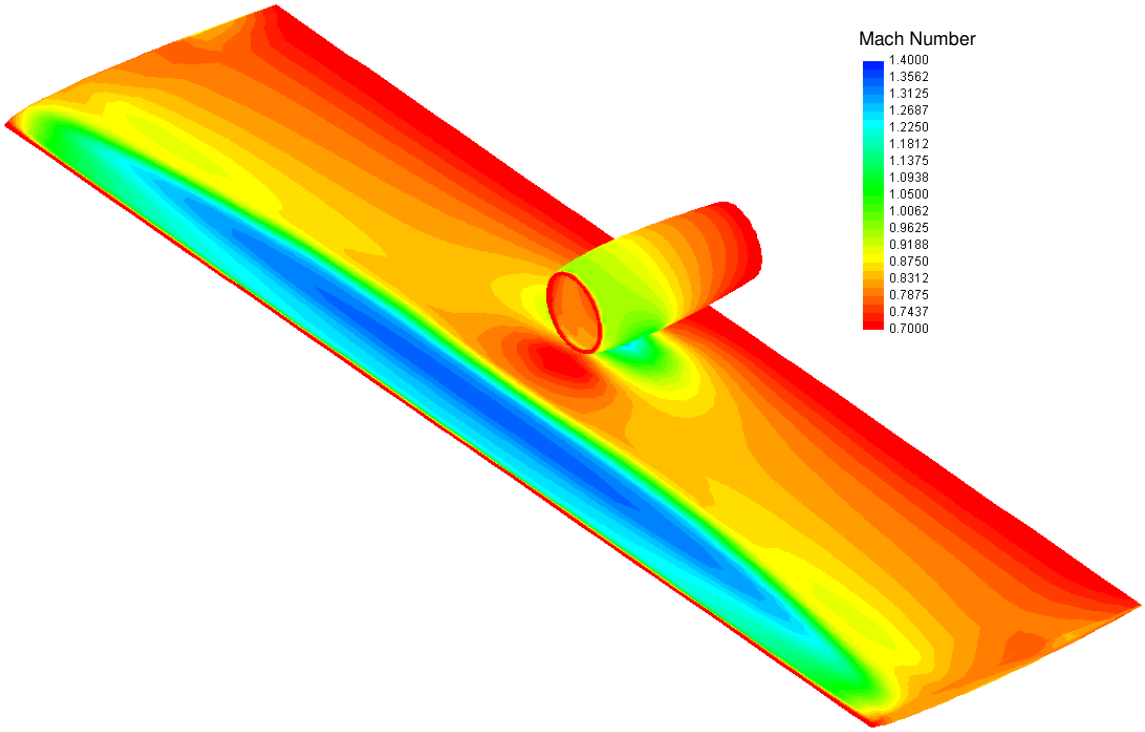


Figure 19 – Nacelle at 50% wing chord, 1.75 radii, C_L=0.4

However, a second, less beneficial behavior can also be observed in the figure. A local zone of high-speed flow can be seen at the nacelle-wing interface. The flow can be seen to accelerate to approximately Mach 1.3 within the convergent-divergent channel formed by this interface. This flow, and the shock associated with it produces various undesirable flow phenomena, which in the end amount to increasing the wing drag.

The second case (50% chord, 6.75 radii) shown in Figure 20 has a much higher degree of nacelle separation than that shown in the first case.

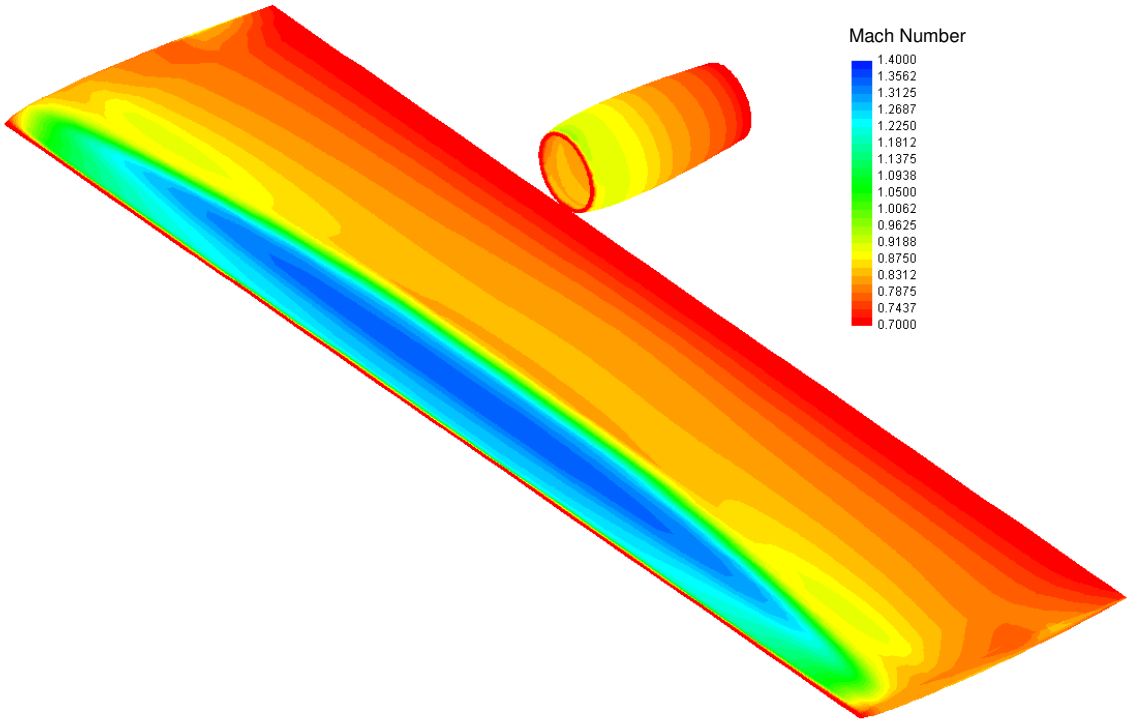
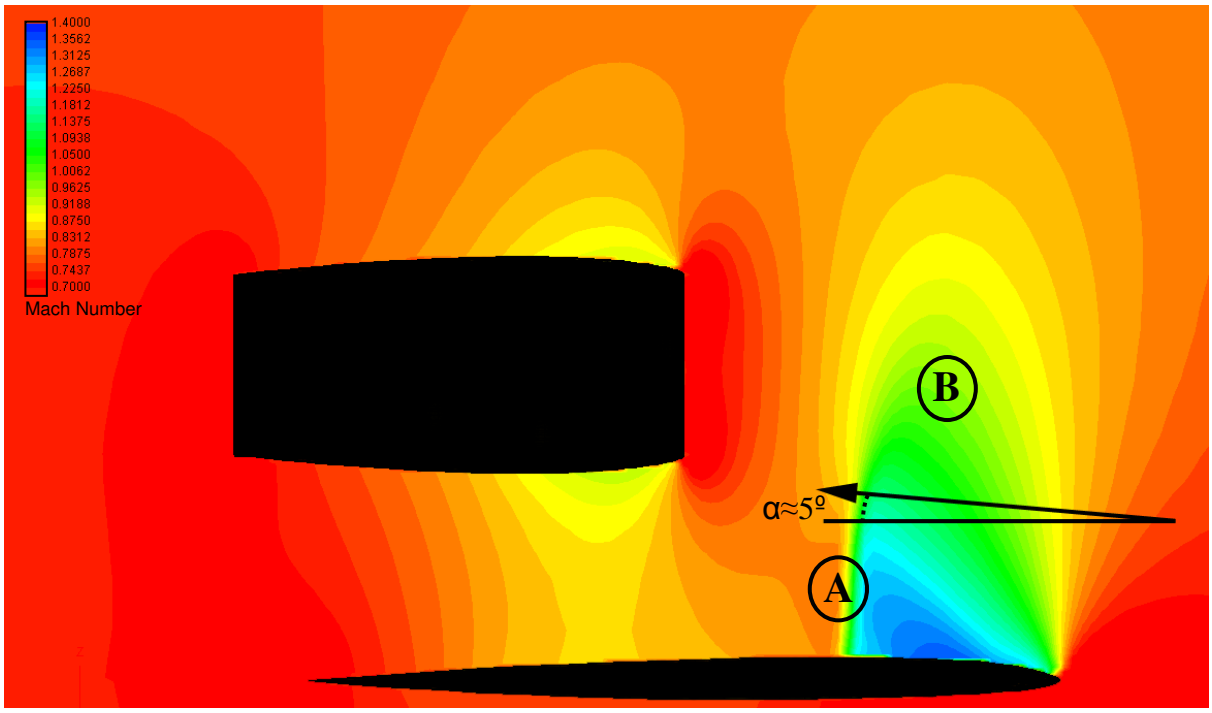


Figure 20 – Nacelle at 50% wing chord, 6.75 radii, C_L=0.4

The maximum local Mach number area is visible at the suction peak of the wing. Because of the large separation between the nacelle and the wing surface, no appreciable compression occurs in the wing-nacelle channel and the upper surface of the wing shows no signs of the influence of the nacelle.

It is also useful to evaluate the changes brought on by modifying the chord-wise location of the nacelle, while leaving the vertical separation constant. By taking a section through the geometry at a span-wise location coincident with the centerline of the nacelle, characteristics of the airflow around the geometry can be better visualized. In Figure 21 below, the initial effects of nacelle suppressing high-speed flow on the top surface of the wing can be seen.



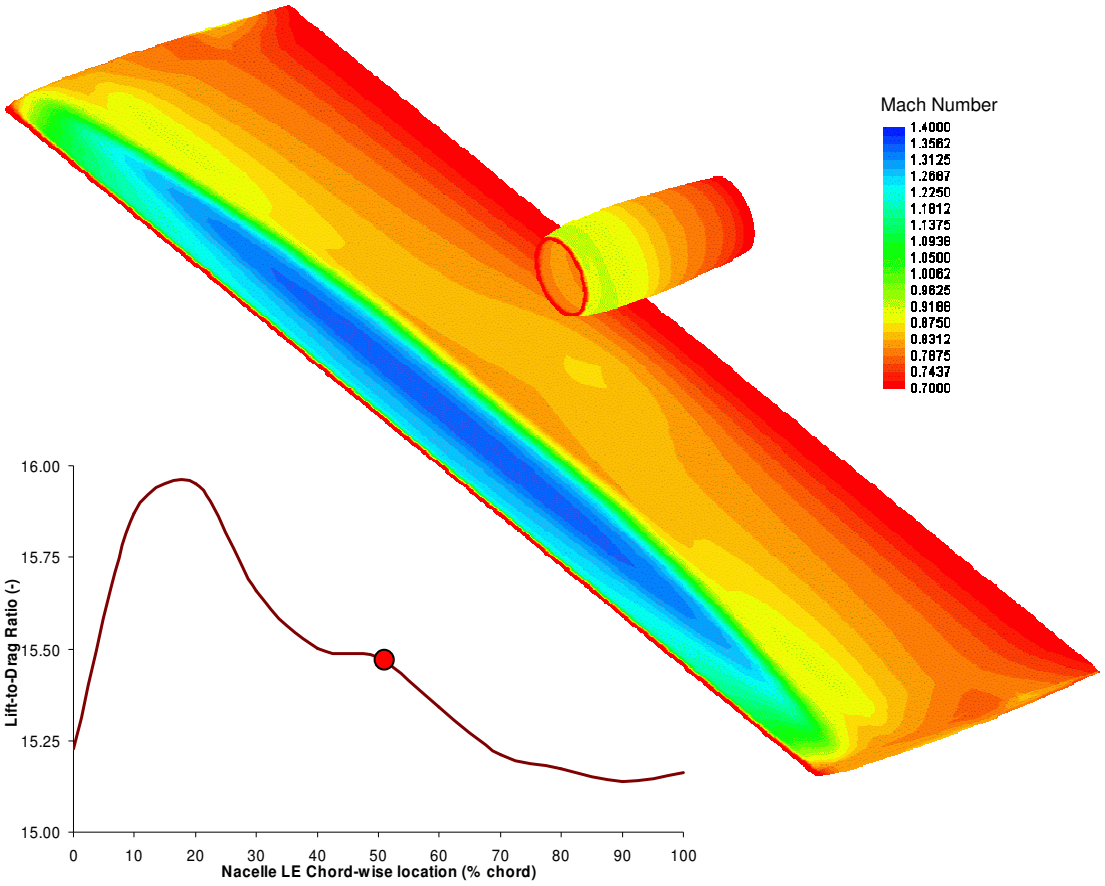


Figure 21 – Nacelle at 50% wing chord, 3.5 radii, C_L=0.4

The first flow example discussed shows little to no nacelle influence on the flow on top of the main wing. A flow behavior of particular interest is the shock that can be seen to occur in zone 'A', with the flow rapidly decreasing speed from Mach 1.4 to freestream speed of Mach 0.75. This is likely the cause of the reduced L/D ratio shown by this wing geometry, as the large area of shock significantly impairs the wing's aerodynamic performance. Also, the zone of high-speed Mach 1.4 flow can be noticed to be quite large, progressing approximately $1/4^{th}$ of the way along the chord of the wing until shocking down to subsonic speed. The vertical influence of the leading edge suction zone is quite significant, with the supersonic flow area, demarked by 'B', extending approximately one half chord length above the wing surface.

The flow behavior which results when a nacelle is positioned further forward to a location 20% behind the wing's leading edge can be seen in Figure 22, and is discussed in detail in the following paragraphs.

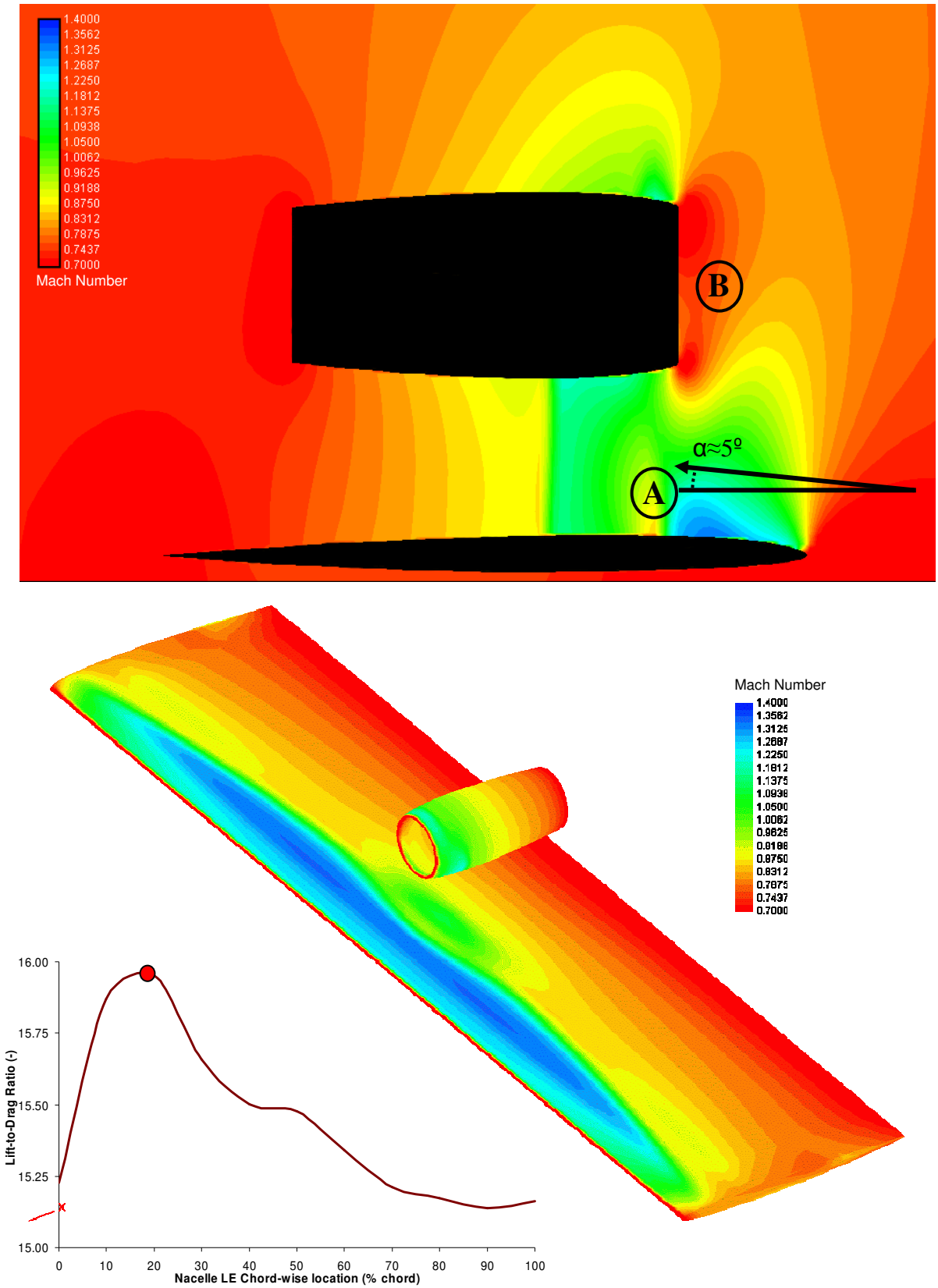
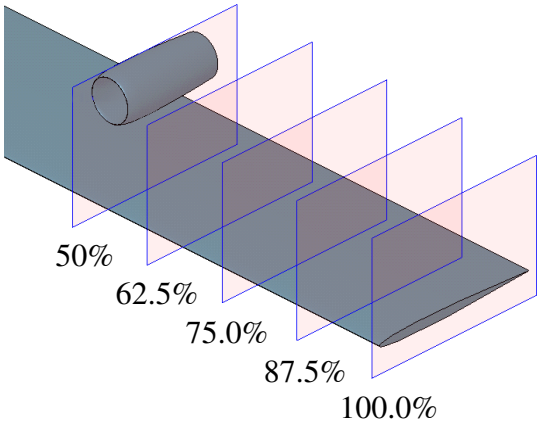


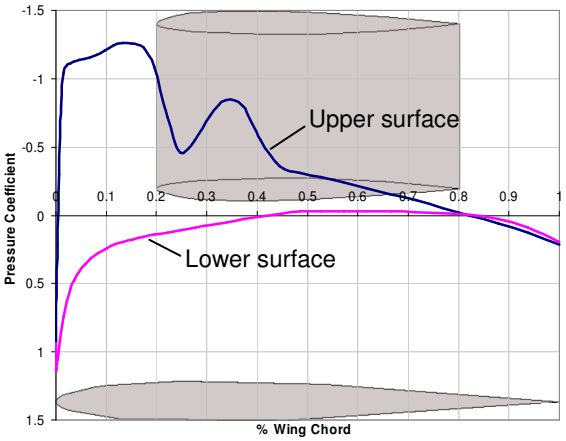
Figure 22 – Nacelle at 20% wing chord, 3.5 radii, C_L=0.4

The L/D ratio of the configuration can be seen to increase after the nacelle is introduced in a location closer to the leading edge suction zone. The flow in the above picture illustrates a reduction in the amount of supersonic shock-down present in front of the nacelle leading edge. A small shock, demarked by 'A', can be seen very close to the surface of the wing, followed by a zone of subsonic flow that is again rapidly accelerated. However, the main quantity of high-speed flow progresses further, undergoing shock-down to sonic speed only after it has passed approximately 1/3rd of the way down the nacelle. The main normal shock that occurs appears to be a much weaker than the one in the previous case, with the majority of the high-speed flow having a sonic number below Mach 1.1. The presence of the nacelle body acts to decrease the magnitude of the shock-down that occurs behind the suction zone.

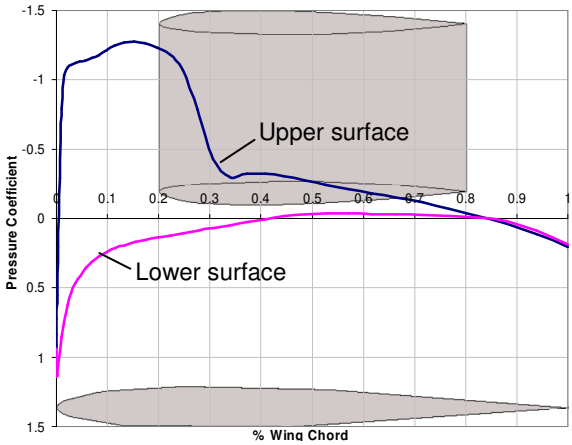
The span-wise extent of the influence the nacelle body has on the flow field is surprisingly large, considering the comparatively small diameter of the nacelle. This can best be seen in the isometric view of the geometry, which shows the increased sonic zone around the front 1/3rd of the nacelle, and the reduced high-speed flow zone around the leading edge suction peak. Also of interest is the decrease in the size of zone 'B', the separated zone of high-speed flow. Being suppressed by the low-speed front generated ahead of the nacelle, supersonic flow can be seen to occupy a smaller area above the wing suction zone.

In order to better evaluate the effect of the nacelle flowfield on wing performance, it is helpful to analyze the surface pressure coefficients on the top and bottom wing surfaces. These, displayed at a number of span-wise sections taken through two versions of the geometry are presented in Figure 23 and Figure 24 below.

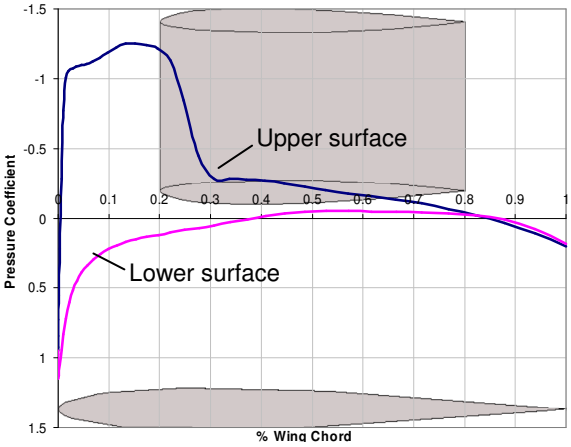




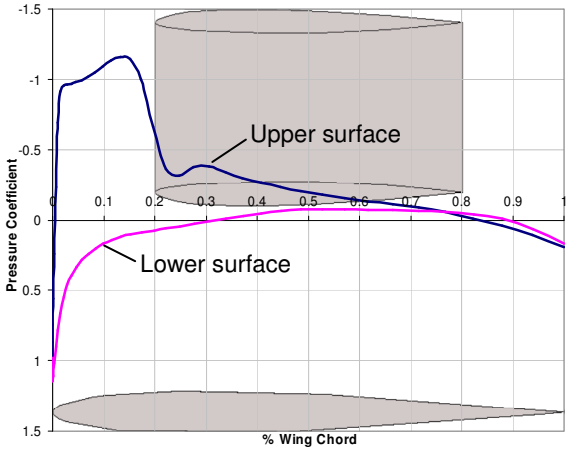
Pressure Coefficient, 50% Wing Span



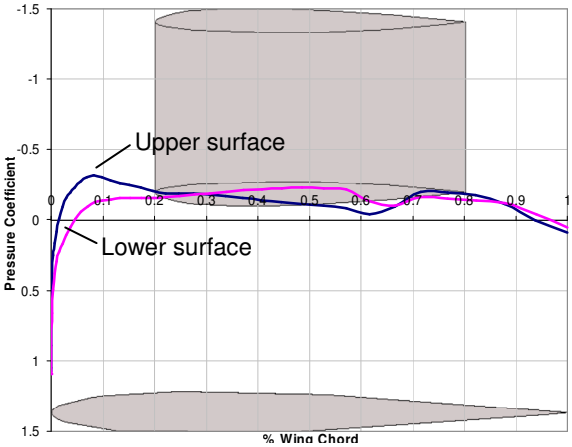
Pressure Coefficient, 62.5% Wing Span



Pressure Coefficient, 75% Wing Span

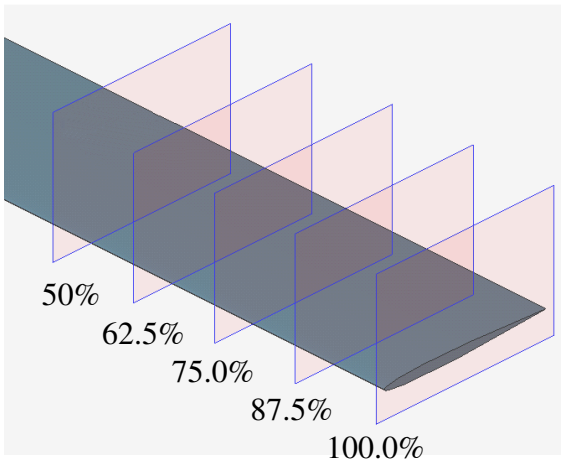


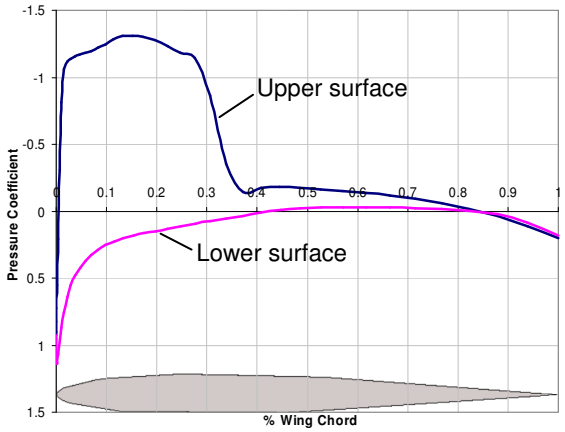
Pressure Coefficient, 87.5% Wing Span



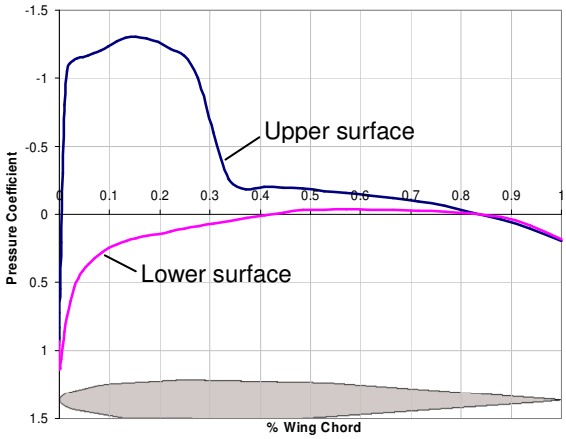
Pressure Coefficient, 100.0% Wing Span

Figure 23 – Upper and Lower Surface Pressure Coefficients, Nacelle Present

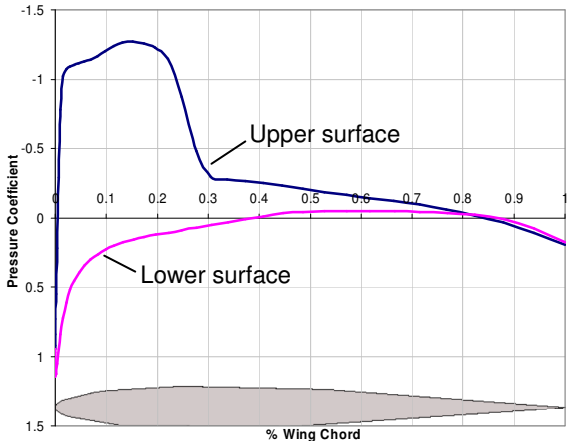




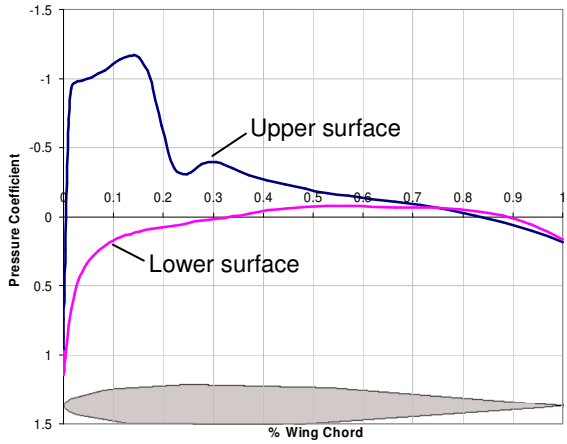
Pressure Coefficient, 50% Wing Span



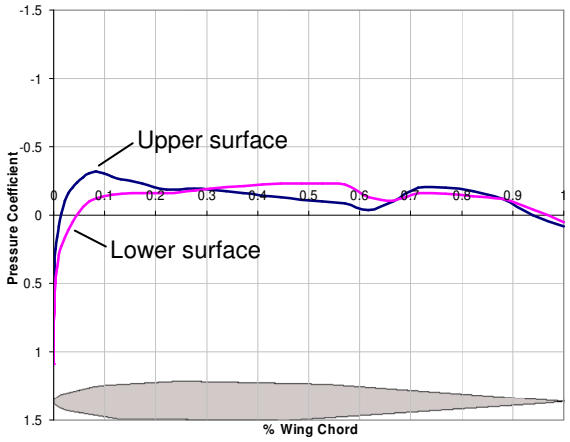
Pressure Coefficient, 62.5% Wing Span



Pressure Coefficient, 75% Wing Span



Pressure Coefficient, 87.5% Wing Span



Pressure Coefficient, 100.0% Wing Span

Figure 24 – Upper and Lower Surface Pressure Coefficients, No Nacelle

The pressure drop present behind the supersonic shock at approximately 30% wing chord is visible in all section cuts except the one at 100% wingspan, where wingtip effects are most prevalent. A comparison of the two pressure contour cases at 50% wing span shows the primary effect of the nacelle body on upper surface pressure coefficients, and as a result, on wing lift capability. While showing the same initial pressure drop as in the other section cuts, the pressure coefficient again increases between approximately 25% to 45% wing chord. This effect, discussed earlier in terms of airflow Mach number underneath the nacelle shows the drag reducing capability the presence of the nacelle body affords to the wing.

It is useful to compare the normalized spanwise lift distribution plots of the 20% wing chord 4.5 radii separation case and that of a clean wing with no nacelle present. A normalized parabolic lift distribution present on a clean wing is shown in Figure 25 below.

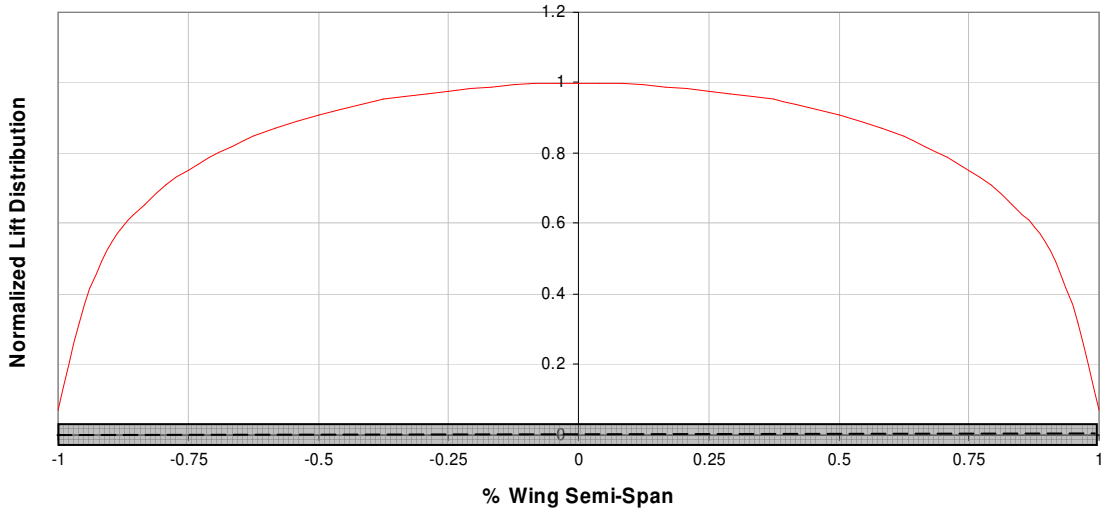


Figure 25 – No-nacelle Spanwise Lift Distribution

After conducting a comparison, it was observed that the lift distribution of the nacelle-on case is only slightly altered by the nacelle's presence. To clearly compare the

two curves without the overlap which tends to occur at the y-axis scale used in the previous figure, the difference between the nacelle-on lift distribution and the clean wing lift distribution is presented in Figure 26.

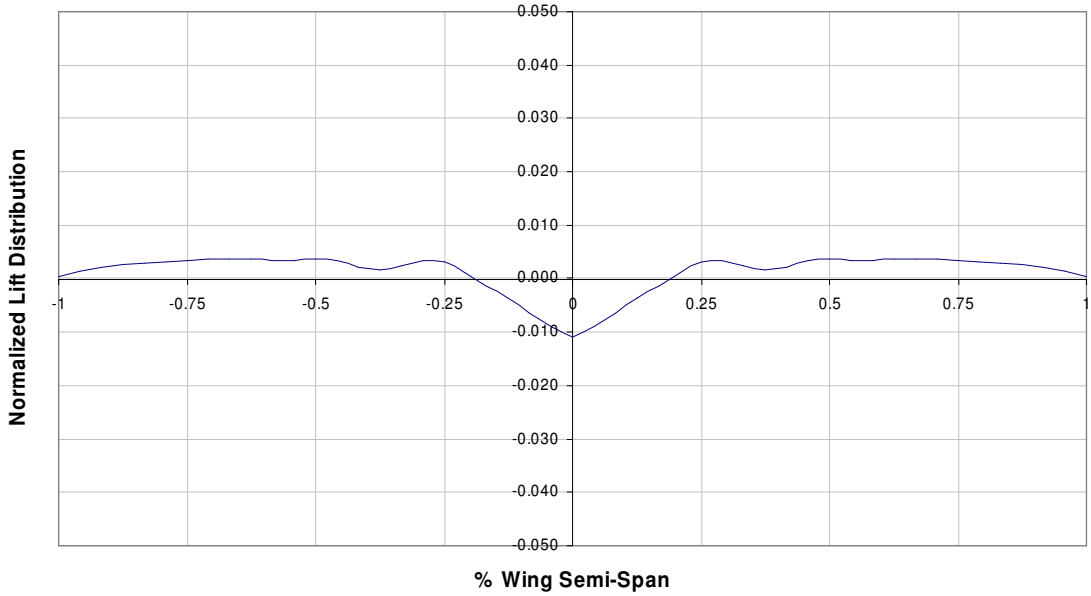


Figure 26 – Spanwise Lift Distribution Difference

It should be noted that in order to achieve equal values of lift on the two wings being compared, the C_L goal of 0.4 is set for the networks of the wing, while in the rest of the study the C_L goal is defined for the entire wing + nacelle geometry. As can be seen from the above plot the sectional lift is slightly effected by the nacelle presence (at most by 1% at the root). The wing's normalized lift distribution is still of a parabolic shape, but with a slight flattening present below the nacelle and an area of higher lift located outboard of approximately 25% semispan. With the areas under their respective lift distribution curves being equal, both of the wing cases produce the same amount of lift. The increase in the assembly's L/D ratio is achieved via a reduction in drag resultant from the lower flow Mach number at the suction zone and a weaker shock.

In order to conduct the study at a single lift coefficient, TranAir CFD code automatically modified the angle of attack to achieve a goal C_L of 0.4 for all geometry cases evaluated. The effects of both the nacelle chordwise and vertical location on the angle of attack necessary to achieve the required C_L can be seen in Figure 27 below.

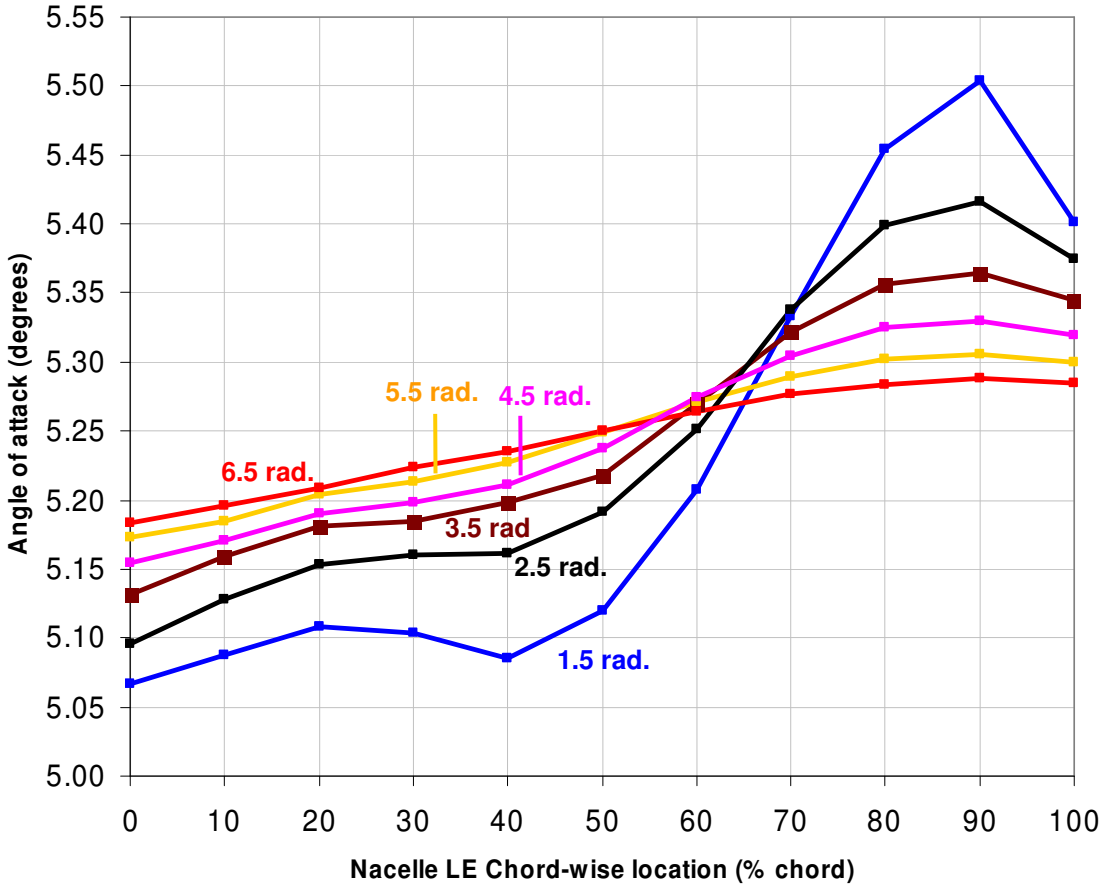


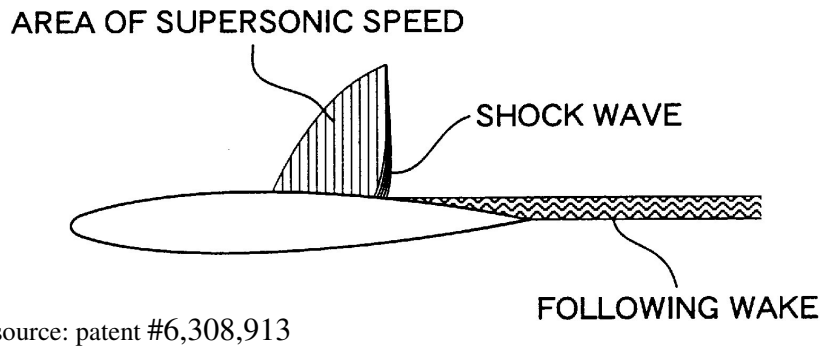
Figure 27 – Angle of Attack vs. Nacelle Location

The cases with the lowest vertical separation have the highest variance of angle of attack as a result of the stronger effect the nacelle flow field has on the upper surface of the wing. The general trend that can be observed is that of the angle of attack growing with the movement of the nacelle towards the trailing edge of the wing. This trend is most clearly observed at the higher vertical-separation cases, where it is not obscured by local flow phenomena present when the wing and nacelle are in close proximity.

Configurations with the nacelle close to the leading edge of the wing achieve a greater level of high-speed flow suppression in the vicinity of the leading-edge suction zone. The resultant lower pressure in the suction zone allows the wing to create lift at a lower angle of attack. As the nacelle nears the trailing edge, the reduction in its effectiveness in suppressing high-speed flow results in higher pressures in the suction zone, forcing the wing to a higher angle of attack to achieve a given C_L value.

An important point that requires discussion is the fact that the aerodynamic results obtained in this study appear to be very different from those claimed in the patent which prompted this study. The nacelle mounting range claimed to exhibit drag reduction and drag divergence Mach number increase in the patent is between 60% and 80% of wing chord. Because this study was conducted at a single Mach number, the drag divergence claims of the patent could not be verified, but obviously, the area of drag reduction that is seen in this set of results is significantly different from that in the patent. It is very likely that an aspect of the study which could not be replicated caused this discrepancy.

The wing airfoil used in the original study is proprietary to the Honda Corporation and therefore could not be used. No performance data is currently available on this airfoil, but a schematic obtained from the patent and presented in Figure 28 shows the wing generating a supersonic shock at an approximately 70% chord location.



source: patent #6,308,913

Figure 28 – Honda Airfoil Schematic

Although transonic or "supercritical" airfoils should produce a suction peak much further down the airfoil than the conventional 25% location, the observed edge of the suction and supersonic flow zones on the Gulfstream airfoil used reached approximately 30% from the leading edge. Obviously, this shock location is significantly different from that present on the wing used in the patent. As shown earlier, the main drag-reducing effect of nacelle is obtained from aerodynamic interaction with the wing's upper-surface shock. It is therefore likely that the location of the shock will define the optimum location of the nacelle, and a different, better performing airfoil would likely go a long way toward explaining the differences seen between the results of the two studies.

Establishing an ideal nacelle location is accomplished by suppression of local supersonic flow at the leading edge suction zone, and the reduction of flow speed at the wing-nacelle interface. However, the weight of structure required to achieve the best location from the aerodynamic standpoint is another aspect of this compromise that must be considered. Finally, the relative value of these factors varies with the mission that the aircraft with this engine installation is designed for. For example, a long-range aircraft will be able to garner more benefit from a low-drag installation, while a short-range aircraft is likely to benefit more from a higher-drag but lower-weight installation. To achieve a balance of all these aspects is the main goal of this study; it was addressed using Multi-Disciplinary Optimization.

9.4. MDO Results

After an iterative optimization and analysis of aircraft performance, the optimum nacelle location was found at 15.9% wing chord, and with a separation of 4.0 radii. Although the most optimal position from an aerodynamic standpoint occurs at a nacelle

location of 15.9% wing chord and a separation of 4.156 radii, the structural constraint forces the optimum overall location to a point slightly closer to the wing surface. The final nacelle location is illustrated in Figure 29 below.

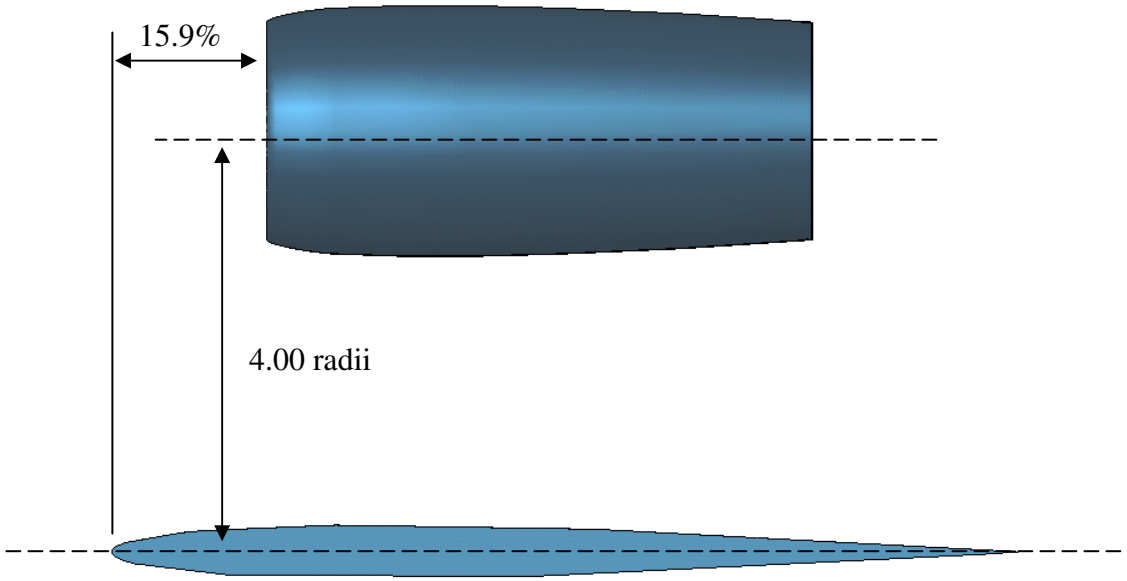


Figure 29 – Optimum Nacelle Position at 15.9% chord, 4.0 radii.

It is difficult to compare the overall efficiency results obtained in this section with those that could be expected from an aircraft with a conventional engine mounting scheme. However, it is possible to show the beneficial effects of mounting location optimization using the MDO study. The locations evaluated within the MDO study range ranged from Nacelle LE from 0% to 100% wing chord, and Nacelle Axis from 1.5 to 5.5 radii separation from wing chord line. In this range, the most optimal location allows for an aircraft that carries 160 lb more payload than the least optimum location, equating to a 4.5% increase in payload capacity. If the same aircraft has an option of using the excess payload capacity in form of fuel, it can extend its mission cruise range by 155 nautical miles, a 6.2% increase.

10. Conclusion

In the course of this study, a complete set of tools to develop and analyze wing and nacelle geometries in Boeing TranAir was created, with an MDO tool used to analyze the effects as applied to a notional aircraft. The results obtained in the aerodynamic part of the study, although significantly different from those described in US Patent #6,308,913, do appear to show a number of important trends.

It is obvious that in order to achieve an optimum benefit from this technology, a large amount of detailed design must be performed. The airfoils used on the wing and nacelle are certain to have a significant effect on the efficiency gains accomplished via the engine mounting scheme explored in this paper. Structural optimization and detailed mission requirements are also sure to play their role, and must both be explored in higher detail. Since detailed optimization was not the goal, the study was performed with general components, and the results achieved are not necessarily indicative of the highest possible level of performance benefits of this type of a nacelle mounting.

From the flow visualizations obtained in the CFD portion of the study, it can be seen that the superposition of the nacelle body and flow-field generated ahead of it has a significant impact on the performance of the wing. By beneficially interacting with an existent zone of supersonic flow, the nacelle allows for a weaker shock, and therefore a smaller pressure increase to occur after the suction zone on the wing's upper surface. This increase in area of high-speed flow and decrease in strength of supersonic shock are the basis the drag-reducing and lift-improving behavior of conventional supercritical airfoils. Finally, it also can be seen that although the nacelle body produces some beneficial results when interacting with an already-formed area of supersonic flow, care

should be exercised in nacelle positioning to avoid creating a secondary area of high-speed flow via acceleration in the interface between the nacelle and the wing's upper surface.

While more detailed research is likely to be necessary to fully appreciate the effects of this nacelle-wing interaction, it has been shown that the basic technology seems to have some promise. An optimized nacelle location was achieved and a set of real-life performance benefits were demonstrated. Additionally, a set of important, yet difficult to quantify benefits, such as increased internal space afforded to the passengers, and the structural synergy of the landing gear, wing and nacelle mount are associated with this method of engine mounting are all sure to benefit a prospective business jet design. Finally, a set of tools have been created and a large amount of work has been conducted to enable further and more in-depth research into the subject.

11. Future Research

The research subject at hand is a very extensive one, and many ways exist in which the fidelity of results achieved could be enhanced, or knowledge in a certain area of study could be increased. These additional studies could not be accomplished because of time limitation, tool availability or being outside of originally defined scope of study.

It would be very helpful to evaluate and confirm the data obtained in this study via a wind-tunnel study of a model configuration. With wind tunnel studies methods being especially suitable for studies featuring multiple iterations with similar geometries, the results produced would be both extensive and accurate.

In order to increase the fidelity of results obtained, a full Navier-Stokes code could be used to evaluate the same problem. However, this is likely to take a large amount of computing time and cost. If a choice was made to continue using TranAir CFD tool, the current solution method could be enhanced by a more detailed simulation of a jet engine installation. The effects of an exhaust plume and a compressor face could both be explored. It is the author's belief that these additions would not fundamentally change the trends observed between nacelle location and configuration lift-to-drag ratio, but a more accurate model of the real-life case would certainly result.

Bibliography

1 VFW 614. 17/5/2004. Aviation Archive: Aviation Heritage. 28 Jan. 2006 <http://www.aviationarchive.org.uk/Gpages/html/G3770.html>.

2 Ber Beent. VFW 614 history. Trans. Oswald Zeijlstra. F-27 Friendship Association. 28 Jan. 2006 <http://www.fokkerf27.nl/index.php?option=com_content&task=view&id=51&Itemid=48&lang=en>.

3 Fujino, M., Kawamura, Y. Honda Giken Kogyo Kabushiki Kaisha. Method for reducing wave resistance in Airplane. United States Patent #6,308,913. October 30, 2001

4 Charlton. E. F., An Octree Solution to Conservation-laws over Arbitrary Regions (OSCAR) with Applications to Aircraft Aerodynamics. University of Michigan, 1997.

5 Aero Grid and Paneling System (AGPS) Reference Manual, Revision 20.00. Boeing Commercial Airplanes Aerodynamics CFD Development & Geometry. The Boeing Company. 2003

6 TRANAIR Users's Manual. The Boeing Company. 2004.

7 Bertin, J. J. and Smith, M. L. Aerodynamics for Engineers, 2nd ed. Prentice-Hall, Englewood Cliffs, New Jersey. 1989., 456

8 TRANAIR Users's Manual. 91

9 Johnson, F. T., Samant, S. S., Bieterman, M. B., Melvin, R. G., Young D. P., Bussoletti, J. E., and Hilmes, C. L., TRANAIR Computer Code Theory Document. National Air and Space Administration (NASA). (NASA CR-4348). Dec. 1992. Originally 1989., 13

10 TRANAIR Computer Code Theory Document, 15-17

11 Drela, M. and Giles, M. B. ISES: a two-dimensional viscous aerodynamic design and analysis code. American Institute of Aeronautics and Astronautics (AIAA). (Paper No. 87-0424). 1987.

12 MISES brochure, Analytical Methods, Inc., Redmond, WA.

13 Terry, Eric L.N. Extension of the aerodynamic program MSES for the simulation of boundary layer suction. Delft University of Technology. November 2004

14 Ibid, 13

15 TRANAIR Computer Code Theory Document, 29

16 TRANAIR Computer Code Theory Document

17 TRANAIR Users's Manual.

18 Jackson, P. FRAeS, ed., Jane's All the World's Aircraft, Jane's Publishing Inc., Annual.

19 Mattingly, J. D., Heiser, W. H., and Pratt, D. T., Aircraft engine design. AIAA Education Series. 2002. 368-9.

20 TRANAIR Users's Manual. 170

21 Meyers, M. K. Aircraft Data Base System: Presentation of Select Weight, Design and Dimensional Data in Graphical and Table Formats – Documentation and User's Guide. Penthouse Productions. April 1988

22 XlXtrFun™ Extra Functions for Microsoft Excel. Advanced Systems Design and Development. <www.xlxtrfun.com>. 28 Jan. 2006

23 Roache, P.J., Ghia, K., and White, F. Editorial Policy Statement on the Control of Numerical Accuracy. ASME Journal of Fluids Engineering, Vol. 108, No. 1., March 1986, p. 2.

Appendix A – Geometry Description (Version with Pylon)

The generalized geometry used in the study is comprised of 13 networks, which can be seen in detail in Figure 30.

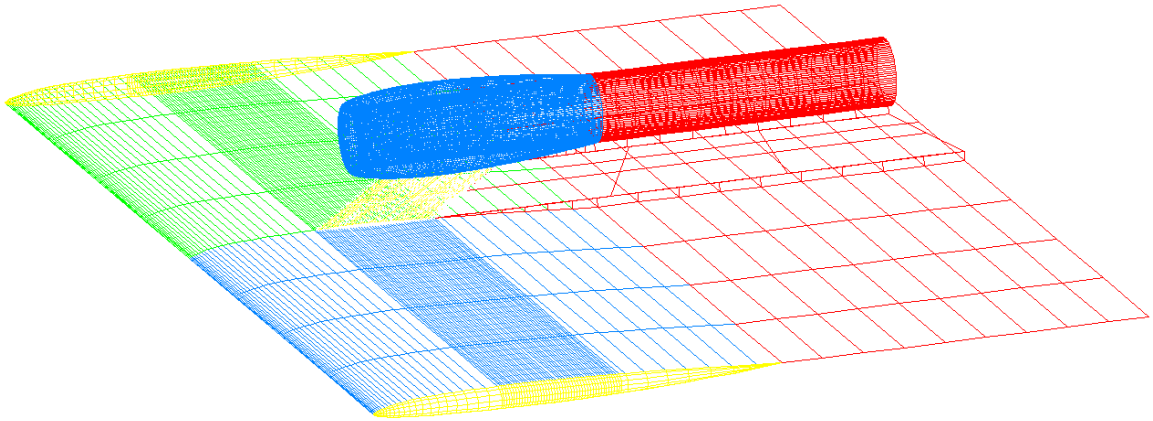


Figure 30 – Full Geometry, Nacelle Position at 40.0% chord, 3.0 radii

In the Figure 30 above, the viscous wake and carryover wake networks are shown in red. The single type-1 wraparound network making up the nacelle is outlined in blue. The two rounded wing cap networks are outlined in yellow. The two networks making up the bottom of the wing are not displayed for clarity of presentation.

To obtain a clearer view of the wake networks, they are displayed individually in Figure 30 below. The two carryover networks are employed in order to allow for the vorticity of the various component wakes to be continuous.

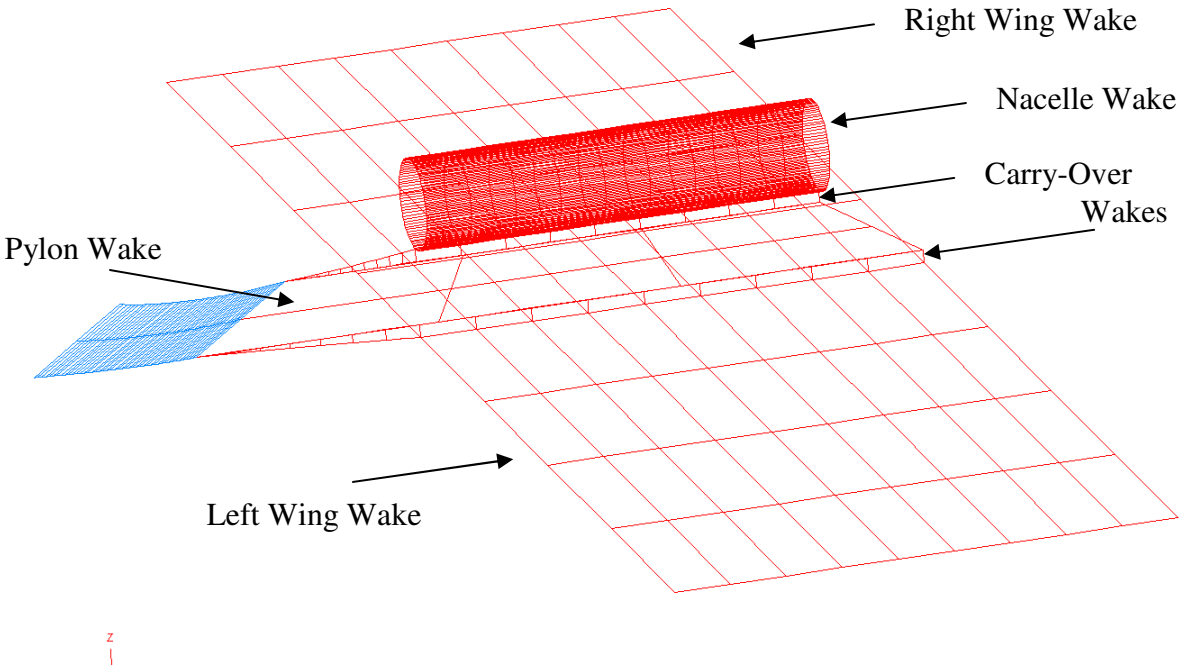


Figure 31 – User-specified wakes

When executing a run with this geometry, TranAir code automatically generates 6 additional networks in order to represent the non-viscous trailing wakes. These 6 additional wakes are automatically attached to the downstream edges of the viscous and carryover wakes mentioned previously. The user-generated surface and wake networks are specified in the .poi input file. The networks used in the geometry, their names and gridding type implemented are all listed in Table 8.

Table 8 – Networks Used, 3-component Geometry

	Grid Size (Row:Column)	Network Names	Gridding type (Row:Column)
Nacelle	124:100	ng	uniform:custom
Wing Surfaces	75:5	wltg, wlbg, wrtg, wrbg	custom:uniform
Wing Caps	75:2	rwcr, rwcl	custom:uniform
Pylon Grids	45:3	plg, prg	uniform:uniform
Viscous Wakes	4:6, 4:100, 4:31, 4:31	prvw, ngvw, wrvw, wlvw	uniform:uniform

Appendix B – TranAir Usage Notes

Drag Values Output

Although the manual for the code states that drag output for the program is to be obtained from pressure-integrated drag values contained one of the output (.ggp) files, specifically the case.pd.ggp file, this proved to be incorrect. After consultation with an experienced TranAir user, it was discovered that it is customary to not use the pressure-integrated drag forces to calculate drag. The pressure integrated drag values output in the .ggp file require the pressures at the leading edge of the geometry to be resolved at a level of accuracy that is impractical for a 3-d case. Because it is much easier to provide for a very fine global grid refinement level in a 2-d case, the output from the .ggp file can be used in that case.

Instead, the preferred method of extracting drag is to get the numbers from the last grid "SOLVER SUMMARY" from the .so file. The drag results are contained in the term called "CDTOT1(1+2+3)" which is the sum of Trefftz plane, Profile and Wave drag coefficients. It is still important to have an adequate level of grid refinement. Typical refinement levels on the final grid are DXmin of ~1% chord. Near the leading edge, this is typically allowed to refine one level smaller to DXmin of ~1/2% chord.

A final and a very important piece of information gained from this communiqué was that the skin friction integration capability of TranAir is still a feature that is not incorporated into the code. The manual for the “G00” version of TranAir mentions this feature as still incomplete.

Rounded Wing Tips

Although not mentioned anywhere in the manual, TranAir suffers frequent failures if a simple “cut-off” wingtip is used on the geometry. It is theorized that the adaptive-gridding portion of the software has difficulty dealing with the 90-degree angles that end up being present in the geometry. Instead, a rounded wing-tip such as that shown in Figure 32 is the preferred method to cap a wing. A procedure to produce such geometry by interpolating a list of points comprising a half-circle is provided in the Appendix D. All of the semicircular sets of points created for the two edge points to be connected can be then joined to make a new network, creating the wing cap.

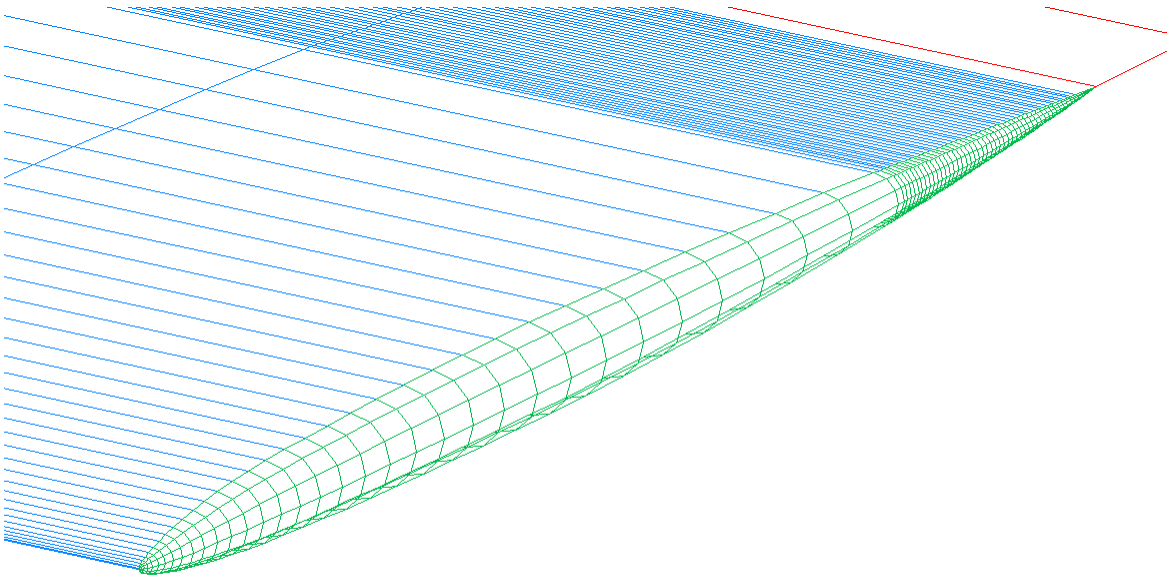


Figure 32 – Wing Cap

Carry-over wakes

While mentioned in some detail in the manual, the importance of these is not strongly emphasized. It appears to be imperative to connect the various wakes trailing the geometry with a set of carry-over wakes, in order to obtain reasonable results.

Edge Abutments

Although the TranAir software features a number of tools which allow for geometries comprised of networks with non-coincident edges to be used ($PEA or $ABU), the author found it significantly easier to use geometries with exact panel edge matching. As mentioned previously, this is accomplished using an "edge point borrowing" technique, which can be easily automated and requires no case-to-case input file modification. A number of examples of such procedures are provided in the Appendix D.

Need to modify initial grid box

An occasional failure is encountered with the following error message in the .so file.

****** FATAL ERROR ****, INCONSISTENT T REGION**

TBOX = 1479 OBOX = 19160 NETWORK = 1 UPPER SURFACE

This error is reliably fixed by adjusting the global box parameters to slightly move the locations where the global grid intersects the geometry.

BAD TGT Vector

An as-of-yet unsolved crash rarely occurs with the error message "BAD TGT VECTOR" contained in the .so file. The only method for avoiding with this failure has been to implement a moderate (~ 10%) reduction in the chord-wise panel density on the network referenced in the error message.

Intermittent failures with no error messages in the .so file

One of the most difficult failures to find a cause for occurred with no error message in the .so file – the .so file is simply cut off in mid-process, and the job did not complete. This error was traced to temporary files generated during CFD runs filling up the limited space on the TranAir server. To avoid this problem, the directory /usr/local/tranair/scratch must be cleaned out periodically.

Appendix C – AGPS Geometry with Superimposed Global Grid

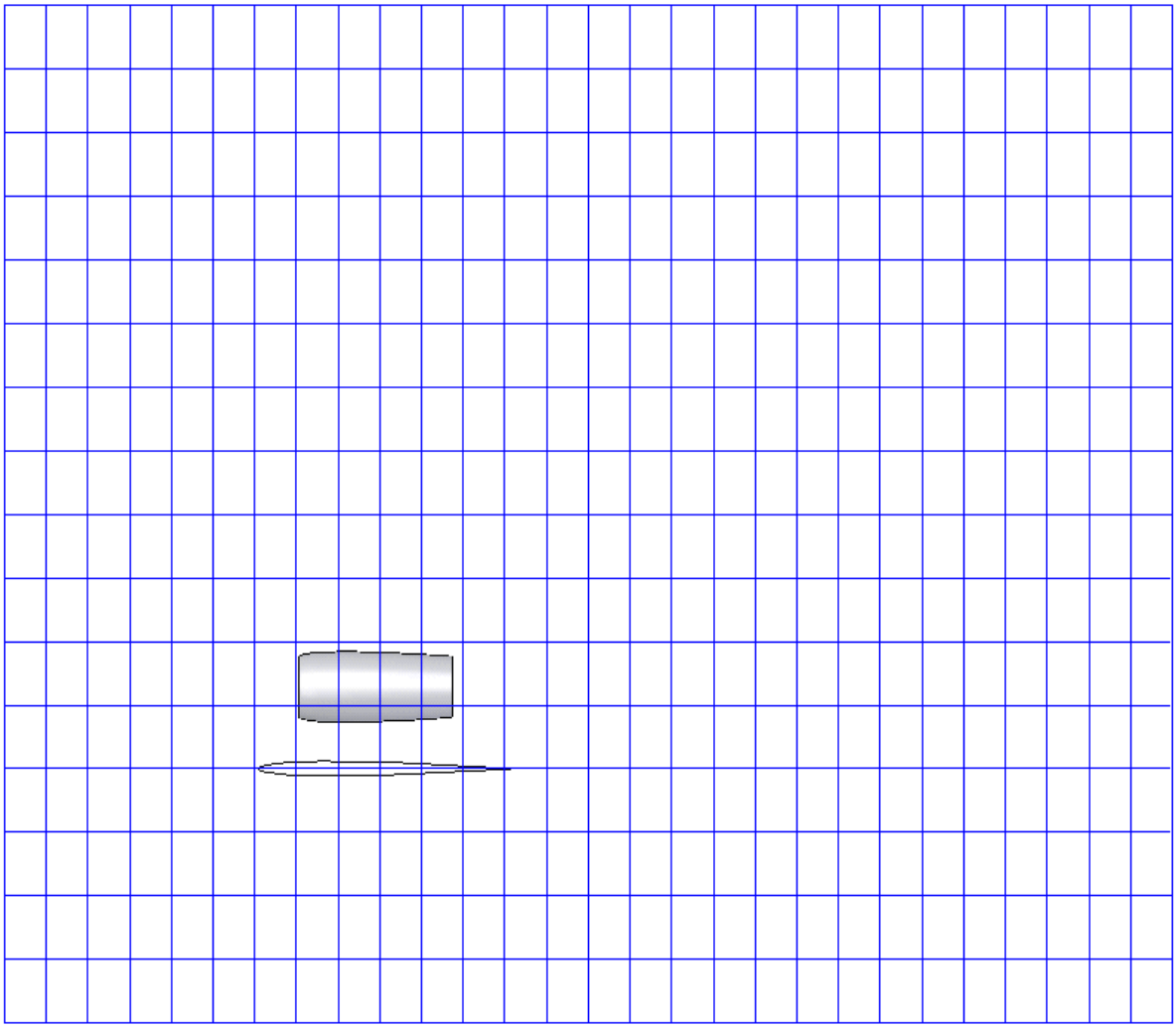


Figure 33 – Geometry and Global Grid, Side View

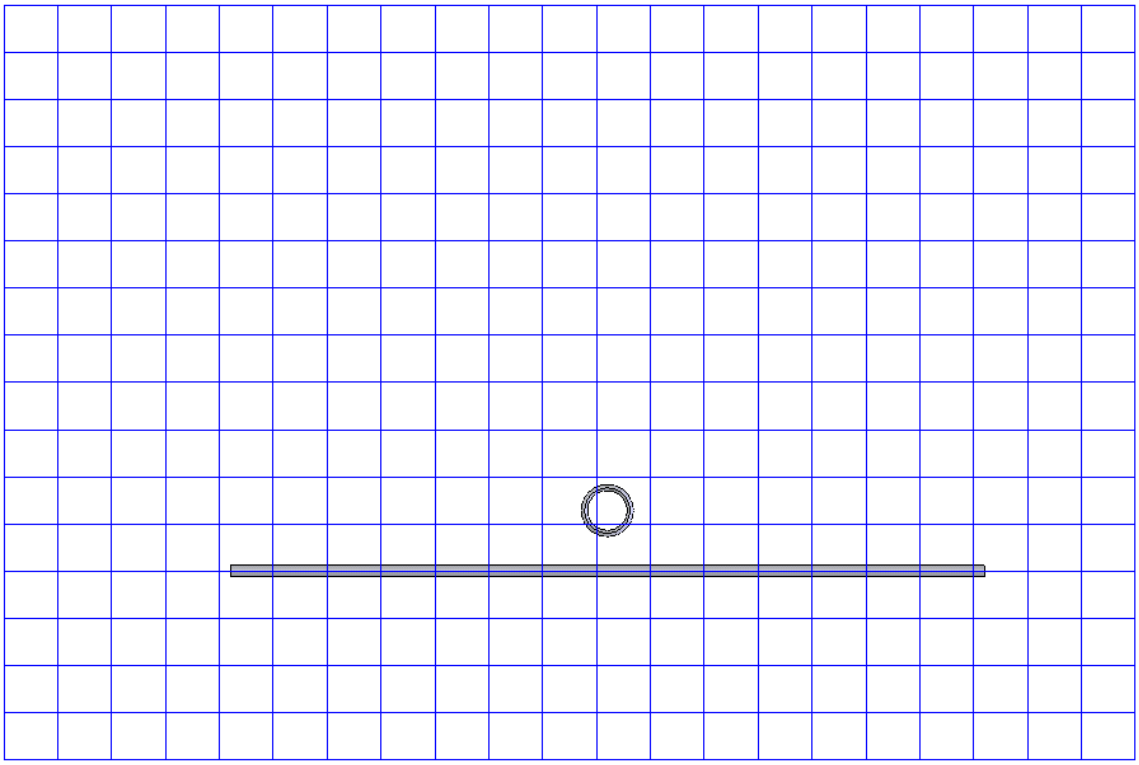


Figure 34 – Geometry and Global Grid, Front View

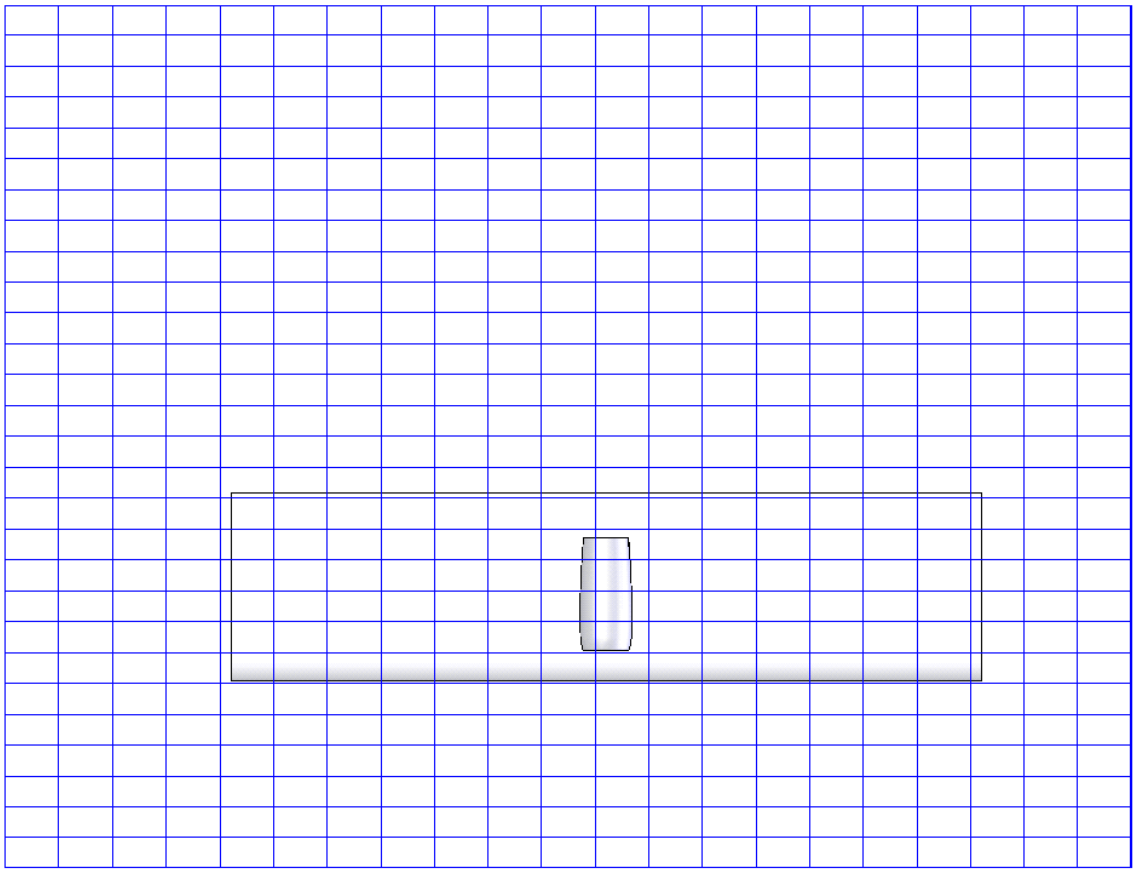


Figure 35 – Geometry and Global Grid, Top View

Appendix D – AGPS Geometry Generation Source Code

```
!upgrade from 33mw3 - rounded-off wing caps
!---CLEAR IT ALL
del *                    !delete all variables
$DEALL #*,*()            !deallocate all symbols
PI = ATAN(1)*4
!---HERE WE READ IN THE AIRFOIL AND MAKE THE ROOT AND TIP SEGMENTS
rms g3am4.txt root
scale roots root (3,1,1.5)
fcv rootq roots quintic
cop root_sm roots.1
del root

scale root_su root_sm (.833333,1,.833333)
scale mid_su root_sm (.833333,1,.833333)
scale tip_su root_sm (.833333,1,.833333)

trn root_s root_su (0,0,0)
trn mid_s mid_su (0, 3.75, 0)
trn tip_s tip_su (0,7.5, 0)
cop mid_q midq.2

!-------------------------------------------------------------------------
!----------------------------------NACELLE CREATION-----------------------
!-------------------------------------------------------------------------
vdgd = 0.35                                          !data gather grid separation from the nacelle LE
vnrad = .20                                          !nacelle diameter
vnlen = 1.5                                          !basically nacelle chord length
vplen = 0.75                                         !pylon chord length
PySwp = 25                                           !pylon sweep
!---------------
fixer = 1
!---------------

vnle = 0+(2.5/10)*10.0                               !nacelle leading edge chord-wise location
!---------------
vple = vnle + .25                                    !pylon leading edge chord-wise location
vple = (vple/vplen)                                  !pylon leading edge chord-wise location
vnle = (vnle/vnlen)                                  !nacelle leading edge chord-wise location
vnhei = 2.0                                          !nacelle symmetry axis z-separation from wing chord,
normalized to n_radius
```

```
rms gn4.txt nac_str
fcv nac_q nac_str quintic
trn nac_base nac_q (vnle, 0, -(vnrad/1.5))
del nac_q
scale nac_q nac_base (vnlen,1,1.50)                    !scale nacelle airfoil thickness
CSR bNACS NAC_Q.1 AXIS=X ANGLE=360
vnhei = vnhei * vnrad
trn NACSURF bNACS (0, 3.75, vnhei)
trn NAC_Q2 NAC_Q (0, 3.75, vnhei)

!-------------------------------------------------------------------------
!----------------------------------WING CREATION--------------------------
!-------------------------------------------------------------------------

!-------HERE WE MAKE THE --LEFT WING SURFACE-- FROM THESE SEGMENTS----------

cay warrl [root_s,mid_s]
fsu WSURFL warrl quintic

!------HERE WE MAKE THE --RIGHT WING SURFACE-- FROM THESE SEGMENTS----------

cay warrr [mid_s,tip_s]
fsu WSURFR warrr quintic

!-------------------------------------------------------------------------
!-------------------------------LEFT PYLON CREATION-----------------------
!-------------------------------------------------------------------------
rms pnl.txt pl_str
fcv pl_q pl_str quintic

bpl   = (vnle/.75)                                     !bottom pylon location

!---HERE WE CREATE THE PYLON 'DRIVER' CURVES
rot pl_r_sr pl_str (90,0,0)
trn  pl_r_srt pl_r_sr (vple,(3.75/.75),(vnhei - vnrad))          !top pylon loft airfoil.
scale pl_r_srts pl_r_srt (vplen,.75,1)                           !make it to scale, make top pylon
curve pl_r_srts
trn  pl_t_srt pl_r_sr (bpl,(3.75/.75),0)                         !bottom pylon airfoil.
scale pl_t_srts pl_t_srt (0.750,.75,1)                           !make it to scale, make bottom pylon
curve pl_t_srts
del [pl_r_sr,pl_r_srt]
```

```
!---HERE WE CREATE THE ACTUAL PYLON SURFACE

cay parrl [pl_r_srts,pl_t_srts]
fsu SUBPL parrl quintic

!-------------------------------------------------------------------------
!------------------------------RIGHT PYLON CREATION-----------------------
!-------------------------------------------------------------------------
rms pnr.txt pr_str
fcv pr_q pr_str quintic

!---HERE WE CREATE THE PYLON 'DRIVER' CURVES
rot pr_r_sr pr_str (90,0,0)
trn  pr_r_srt pr_r_sr (vple,(3.75/.75),(vnhei - vnrad))                !top pylon loft airfoil.
scale pr_r_srts pr_r_srt (vplen,.75,1)                                 !make it to scale, make top pylon
curve pl_r_srts
trn  pr_t_srt pr_r_sr (bpl,(3.75/.75),0)                               !bottom pylon airfoil.
scale pr_t_srts pr_t_srt (0.750,.75,1)                                 !make it to scale, make bottom pylon
curve pl_t_srts
del [pr_r_sr,pr_r_srt]

!xc = vple*vplen-.5*.750
!zc = (vnhei-vnrad)
!pang=ATAN(xc/zc)*57.2957795

!---HERE WE CREATE THE ACTUAL PYLON SURFACE

cay parrr [pr_r_srts,pr_t_srts]
fsu SUBPR parrr quintic



!-------------------------------------------------------------------------
!---------------------------Find out S-extent of LE of wing---------------
CLO wsurfl LEpt (0,0,0)
$CALL GET_VAL(LEpt.1,0,0,0,0,wLEs)
!wLEs=1-wLEs

!-------------------------------------------------------------------------
!---------------------------------CREATE WSURF R TOP----------------------
CBN wb1 [(0,0),((wLEs),0)]
CBN wt1 [(0,1),((wLEs),1)]
CBN wl1 [(0,0),(0,1)]
```

```
CBN wr1 [((wLEs),0),((wLEs),1)]
stn wmap1 [wb1,wt1] [wr1,wl1] linear
CSB wrt wmap1 wsurfr                                    !Create subrange wing_right_top

!-------------------------------------------------------------------------
!---------------------------------CREATE WSURF R BOT----------------------
CBN wb2 [(wLEs,0),(1,0)]
CBN wt2 [(wLEs,1),(1,1)]
CBN wl2 [(wLEs,0),(wLEs,1)]
CBN wr2 [(1,0),(1,1)]
stn wmap2 [wt2,wb2] [wl2,wr2] linear
CSB wrb wmap2 wsurfr                                    !Create subrange wing_right_bot

!-------------------------------------------------------------------------
!---------------------------------CREATE WSURF L TOP----------------------
CBN wb3 [(0,0),((wLEs),0)]
CBN wt3 [(0,1),((wLEs),1)]
CBN wl3 [(0,0),(0,1)]
CBN wr3 [((wLEs),0),((wLEs),1)]
stn wmap3 [wb3,wt3] [wr3,wl3] linear
CSB wlt wmap3 wsurfl                                    !Create subrange wing_right_top

!-------------------------------------------------------------------------
!---------------------------------CREATE WSURF L BOT----------------------
CBN wb4 [(wLEs,0),(1,0)]
CBN wt4 [(wLEs,1),(1,1)]
CBN wl4 [(wLEs,0),(wLEs,1)]
CBN wr4 [(1,0),(1,1)]
stn wmap4 [wt4,wb4] [wl4,wr4] linear
CSB wlb wmap4 wsurfl                                    !Create subrange wing_right_top

!-------------------------------------------------------------------------------------------------
!---HERE WE FIND INTERSECTION CURVES BETWEEN WING TOPS AND PYLON SIDES, TO BE USED FOR WING/PYLON TRIMMING
!-------------------------------------------------------------------------------------------------

ssi wrt SUBPR wintR pintR
ssi wlt SUBPL wintL pintL
!-------------------------------------------------------------------------------------------------
!---HERE WE FIND INTERSECTION CURVES BETWEEN NACELLE AND PYLON SIDES, TO BE USED FOR NACELLE/PYLON TRIMMING
!-------------------------------------------------------------------------------------------------

ssi NACSURF SUBPR nintR pint2R
ssi NACSURF SUBPL nintL pint2L
```

```
dra NAC_Q.1 200 200 200 200 red no
dra nintl 500 500 500 500 blue no
dra pint2l 500 500 500 500 green no

!-------------------------------------------------------------------------------
!---------------------TRIM LEFT PYLON AGAINST THE WING SURFACE------------------
!-------------------------------------------------------------------------------
$CALL GET_COORD(pintl.1.1,0,0,pintlS0,pintlT0)        !this stores coordinates of start and end of the
intersection curve
$CALL GET_COORD(pintl.1.1,1,0,pintlS1,pintlT1)        !this stores coordinates of start and end of the
intersection curve

CBN pltRIG [(1,0),(1,1)]
CBN pltLEF [(0,1),(0,0)]
CBN pltTOP [(1,0),(0,0)]
stn pltMAP [pint2L.1.1,pintL.1.1] [pltLEF,pltRIG] linear
CSB SUBPLT1 pltMAP SUBPL                              !Create subrange left pylon

$CALL GET_COORD(pint2R.1.1,1,0,pniS1,pniT1)           !pylon-nacelle intersection, for pylon-nacelle joint
$CALL GET_VAL(SUBPR,pniS1,pniT1,0,0,pnxe,pnys)
$CALL GET_COORD(pint2R.1.1,0,0,pniS2,pniT2)
$CALL GET_VAL(SUBPR,pniS2,pniT2,0,0,pnxs,pnye)

!-------------------------------------------------------------------------
!---------------------TRIM RIGHT PYLON AGAINST THE WING SURFACE-------------
!-------------------------------------------------------------------------
$CALL GET_COORD(pintr.1.1,0,0,pintrS0,pintrT0)        !this stores coordinates of start and end of the
intersection curve
$CALL GET_COORD(pintr.1.1,1,0,pintrS1,pintrT1)        !this stores coordinates of start and end of the
intersection curve

$CALL GET_VAL(SUBPR,pintrS0,pintrT0,0,0,pwxs,pwye)    !pylon-wing intersection, for pylon-wing joint
$CALL GET_VAL(SUBPR,pintrS1,pintrT1,0,0,pwxe,pwys)

deall pwye, pwys, pnys, pnye

CBN prtRIG [(1,0),(1,1)]
CBN prtLEF [(0,0),(0,1)]
CBN prtTOP [(1,1),(0,1)]
!stn prtMAP [pintR.1.1,prtTOP] [prtLEF,prtRIG] linear
!stn prtMAP [pintr.1.1,pint2R.1.1] [prtLEF,prtRIG] linear
stn prtMAP [pintR.1.1,pint2R.1.1] [prtLEF,prtRIG] linear
```

```
CSB SUBPRT1 prtMAP SUBPR                                    !Create subrange right pylon

!-------------------------------------------------------------------------------
!---------------------------------GRID PYLON MAPS-------------------------------
!-------------------------------------------------------------------------------
grd SUBPRT1 prg 45 LIN LIN 3 LIN LIN
grd SUBPLT1 plg 45 LIN LIN 3 LIN LIN

dra prg 100 100 100 100 red no
dra plg 100 100 100 100 red no

!-------------------------------------------------------------------------------
!---------------------------------RIGHT WING TOP SURFACE TRIMMING---------------
!-------------------------------------------------------------------------------
$CALL GET_COORD(wintr.1.1,0,0,wintrS0,wintrT0)        !this stores coordinates of start and end of the
intersection curve
$CALL GET_COORD(wintr.1.1,1,0,wintrS1,wintrT1)        !this stores coordinates of start and end of the
intersection curve

!-----------------------------------------
!--------------DO FUNKY STUFF-------------
!-----------------------------------------
cst wrcrv1 [(1,0),(wintrS1,wintrT1)]
cst wrcrv2 [(wintrS0,wintrT0),(0,0)]
fcv crv1 wrcrv1 quintic                                 !Fit a curve to the two points defined above
fcv crv2 wrcrv2 quintic                                 !Fit a curve to the two points defined above

cat [crv1, wintr.1.1] rescrv Starting-Point= 1e-3     !join crv1 and intersection 2D curve into rescrv
cat [rescrv, crv2] wrBOT Starting-Point= 1e-3         !join rescrv and crv2

n=40

!---------------------------------------MAKE WING CHUNK GRID RULES---------------

$FOR I = n to 0 step -1
   ATL wprp ((((1-(I/n))^2.5))*1)                     !wing pre pylon rule
$ENDDO

!WFD FILE=g:\MT\whee.txt OBJ=wprp FORMAT=(15F10.4)

ESK wrt prePER S-PARAM=wprp T-PARAM=(0)               !extract surface knots at these s-params and t-
params=0, get some points
```

```
n=1
m=1
call get_length(wprp,I)


!--------TAKEOUT THE POINTS THAT WILL CONFLICT WITH THE PYLON GRIDDING
$WHILE n <> I DO
      $call get_val(prePer.1.<n>.1,0,0,0,0,A)
      $if (A > wintrS0) and (A < wintrS1) then
            RFL LIST-NAME=prePer.1 ELEMENTS=prePer.1.<n>
      else
            n=n+1
      $endif
      call get_length(prePer.1,I)
$ENDDO


$call get_val(prePer.1.1.1,0,0,0,0,A)
n=1
!--------GET TO THE PYLON START POINT
$WHILE (A < wintrS0) DO
      $call get_val(prePer.1.<n>.1,0,0,0,0,A)
      n=n+1
$ENDDO
n=n-1

ATL prePER.1 prg.1.* I=prePER.1.<n>

cop rwgc prePer.1                                    !here we collate pre, pylon and post chunks to a
single list called rwgc

del [crv1, crv2, rescrv, wrcrv1, wrcrv2]             !delete all used variables

CBN wrRIG [(1,0),(1,1)]
CBN wrLEF [(0,0),(0,1)]
CBN wrTOP [(1,1),(0,1)]
stn wrMAP [wrBOT,wrTOP] [wrLEF,wrRIG] linear         !for not-full chord pylon

CSB wrtt wrMAP wrt                                   !Create subrange right wing
dra wrtt
!-------------------------------------------------------------------------
!---------------------------------LEFT WING TOP SURFACE TRIMMING-------------------------
!-------------------------------------------------------------------------
```

```
$CALL GET_COORD(wintl.1.1,0,0,wintlS0,wintlT0)        !this stores coordinates of start and end of the
intersection curve
$CALL GET_COORD(wintl.1.1,1,0,wintlS1,wintlT1)        !this stores coordinates of start and end of the
intersection curve

cst wlcrv1 [(1,1),(wintlS1,wintlT1)]
cst wlcrv2 [(wintlS0,wintlT0),(0,1)]
fcv crv1 wlcrv1 quintic                          !Fit a curve to the two points defined above
fcv crv2 wlcrv2 quintic                          !Fit a curve to the two points defined above

cat [crv1, wintl.1.1] rescrv Starting-Point= 1e-3     !join crv1 and intersection 2D curve into rescrv
cat [rescrv, crv2] wlTOP Starting-Point= 1e-3         !join rescrv and crv2

!---------------------------------------MAKE WING CHUNK GRID RULES----------------
cop lwgc prePer.1
RFL LIST-NAME=lwgc ELEMENTS=lwgc.L
ATL lwgc plg.6.*
!RFL LIST-NAME=postPer.1 ELEMENTS=postPer.1.1        not necessary because already removed this element
above
ATL lwgc postPer.1.*
del lwgc
cop lwgc preper.1

del [crv1, crv2, rescrv, wlcrv1, wlcrv2]              !delete all used variables

CBN wlRIG [(1,0),(1,1)]
CBN wlLEF [(0,0),(0,1)]
CBN wlBOT [(1,0),(0,0)]
stn wlMAP [wlBOT,wlTOP] [wlLEF,wlRIG] linear          !for not full chord pylon
CSB wltt wlMAP wlt                                    !create subrange left wing

!-------------------------------------------------------------------------
!----------------------------Mid-process VIZ------------------------------
!-------------------------------------------------------------------------
!draw postper.1 1000 1000 1000 1000 green
!draw preper.1 1000 1000 1000 1000 green
!dra wrtt 100 100 100 100 red

!-------------------------------------------------------------------------
!-------------------TRIM NACELLE AGAINST THE PYLON SURFACE-----------------
!-------------------------------------------------------------------------
$CALL GET_COORD(nintR.1.1,0,0,tnRS0,tnRT0)      !this stores coordinates of start and end of the
intersection curve
```

```
$CALL GET_COORD(nintR.1.1,1,0,tnRS1,tnRT1)      !this stores coordinates of start and end of the
intersection curve
$CALL GET_COORD(nintL.1.1,0,0,tnLS0,tnLT0)      !this stores coordinates of start and end of the
intersection curve
$CALL GET_COORD(nintL.1.1,1,0,tnLS1,tnLT1)      !this stores coordinates of start and end of the
intersection curve

!first we make a set of points in s, going from 0 to 1 to mark where it is that we wish to put points on the
nacelle
!then we find out which of these points go before pylon intersection (0-tnRS0) and after (tnRS1-1).
!then, we find out what XYZ points on the untrimmed surface those S-points correspond to. Then, we join the
pre and post
!pylon XYZ points with the points of the pylon intersection, gotten from the already-made pylon grid

!---------------MAKE RIGHT BOUNDARY-----------------
cst nLcrv1 [(1,1),(tnLS1,tnLT1)]
cst nLcrv2 [(tnLS0,tnLT0),(1,0)]
fcv crv1 nLcrv1 quintic                                 !Fit a curve to the two points defined above
fcv crv2 nLcrv2 quintic                                 !Fit a curve to the two points defined above

cat [crv1, nintL.1.1] rescrv Starting-Point= 1e-8       !join crv1 and intersection 2D curve into rescrv
cat [rescrv, crv2] ntRIG Starting-Point= 1e-8           !join rescrv and crv2

del [crv1, crv2, rescrv, nLcrv1, nLcrv2]                !delete all used variables

!---------------MAKE LEFT BOUNDARY-----------------
cst nRcrv1 [(0,1),(tnRS1,tnRT1)]
cst nRcrv2 [(tnRS0,tnRT0),(0,0)]
fcv crv1 nRcrv1 quintic                                 !Fit a curve to the two points defined above
fcv crv2 nRcrv2 quintic                                 !Fit a curve to the two points defined above

cat [crv1, nintR.1.1] rescrv Starting-Point= 1e-3       !join crv1 and intersection 2D curve into rescrv
cat [rescrv, crv2] ntLEF Starting-Point= 1e-3           !join rescrv and crv2
del [crv1, crv2, rescrv, nRcrv1, nRcrv2]                !delete all used variables

!-----------------MAKE TRIM MAP--------------------

CBN ntTOP [(1,1),(0,1)]
CBN ntBOT [(1,0),(0,0)]
stn ntMAP [ntBOT,ntTOP] [ntLEF,ntRIG] linear
CSB subNACS ntMAP NACSURF                               !Create subrange nacelle

!-------------------------------------------------------------------------------
```

```
!----------------MAKE RIGHT WING GRID ACCOUNTING FOR THE PYLON--------------------
!---------------------------------------------------------------------------------

CLO wrtt wrclo rwgc                                    !Find closest points on the right wing top surface

call get_length(wrclo,n)
cst RWmap (0)
for i=2 to n-1 do
 call get_val(wrclo.<i>.1,0,0,0,0,s,t)
 cst RWmap (s)
enddo
cst RWmap (1)

$WRITE RWmap

!---------------------------------------------------------------------------------
!----------------MAKE LEFT WING GRID ACCOUNTING FOR THE PYLON--------------------
!---------------------------------------------------------------------------------

CLO wltt wlclo lwgc                                          !Find closest points on the right wing top
surface

call get_length(wlclo,n)
cst LWmap (0)
for i=2 to n-1 do
 call get_val(wlclo.<i>.1,0,0,0,0,s,t)
 cst LWmap (s)
enddo
cst LWmap (1)

draw wrtt 100 100 100 100 green
draw psur 50 50 199 199 red no
!---------------------------------------------------------------------------------
!---------------------------------GRID WING MAPS----------------------------------
!---------------------------------------------------------------------------------
grd wlb wlbg 2 LWmap LWmap 5 LIN LIN GRIDDING-RULE=PAR
grd wltt wltg 2 LWmap LWmap 5 LIN LIN GRIDDING-RULE=PAR
grd wrb wrbg 2 LWmap LWmap 5 LIN LIN GRIDDING-RULE=PAR
grd wrtt wrtg 2 LWmap LWmap 5 LIN LIN GRIDDING-RULE=PAR


!---------------------------------------------------------------------------------
!---------------------------------GRID NACELLE------------------------------------
```

```
!-------------------------------------------------------------------------------
n=100

$FOR I = 0 to (n/2) step 1
   ATL nr (((1-(I/(n/2))^2)/2)+(((I/(n/2))-1)*.002))
!the second part of the equation, separated by the + sign compensates for the
!likely assymmetricity of the airfoil used on the nacelle

$ENDDO
rvs nr
$FOR I = ((n/2)-1) to 0 step -1
   ATL nr ((.5+(((1-(I/(n/2)))^2)/2))-((I/((n/2)-1))*.002))
$ENDDO
!pause
s=0
I = 1

$WHILE (s < tnRT0) DO
    call get_val(nr.<i>,0,0,0,0,s,t)
      ATL nprp nr.<i>
    I = I + 1
$ENDDO

$WHILE (s < tnRT1) DO
    call get_val(nr.<i>,0,0,0,0,s,t)
    I = I + 1
$ENDDO

$WHILE (s < 1) DO
    call get_val(nr.<i>,0,0,0,0,s,t)
      ATL npop nr.<i>
    I = I + 1
$ENDDO

ESK NACSURF nprpR S-PARAM=(0) T-PARAM=nprp
ESK NACSURF npopR S-PARAM=(0) T-PARAM=npop

cop rngc nprpR.*.1
RFL LIST-NAME=rngc ELEMENTS=rngc.L
ATL rngc prg.6.*
!RFL LIST-NAME=npopR.1 ELEMENTS=npopR.1.1
ATL rngc npopR.*.1
```

```
CLO SUBNACS nrclo rngc                                  !Find closest points on the right wing top surface

call get_length(nrclo,n)
cst RNmap (0)
$FOR i=2 to n-1 do
   call get_val(nrclo.<i>.1,0,0,0,0,s,t)
   cst RNmap (t)
$ENDDO
cst RNmap (1)

!pare RNMAP for points that are made out of order b/c AGPS oftentimes malfunctions (only when nacelle
airfoil is way thin)

vos = -.1

call get_length(RNmap,n)
i = 1
$WHILE i <> n DO
   call get_val(RNmap.<i>,0,0,0,0,vs)
   $IF vos > vs THEN
       rfl RNmap RNmap.<i>
   ELSE
       vos=vs
         i=i+1
   $ENDIF
$ENDDO

grd SUBNACS ng 100 LIN LIN 30 RNmap RNmap GRIDDING-RULE=PAR
draw ng 200 200 200 200 blue no

!-------------------------------------------------------------------------------
!---------------------------------CREATE PYLON CAP------------------------------
cay pcl plg.1
cay pcr prg.6
rvs pcl
GTX pcm pcr TYPE=PRO ARG1=(0,1,0,-3.75)
cay pclg [pcm.1,pcl.1]                                  !pylon cap left gridded
cay pcrg [pcm.1,pcr.1]                                  !pylon cap right gridded

!-------------------------------------------------------------------------------
!---------------------------------CREATE WING CAP L TYPE 2----------------------
cay wcltop wltg.1
cay wclbot wlbg.31
```

```
rvs wclbot
GTX wclmid wcltop TYPE=PRO ARG1=(0,0,1,0)
cay wcltg [wclmid.1,wcltop.1]                                  !wing cap left top gridded
cay wclbg [wclbot.1,wclmid.1]                                  !wing cap left bot gridded

!-------------------------------------------------------------------------------
!----------------------------------CREATE WING CAP R TYPE 2-----------------------
cay wcrtop wrtg.31
cay wcrbot wrbg.1
rvs wcrbot
GTX wcrmid wcrtop TYPE=PRO ARG1=(0,0,1,0)
cay wcrtg [wcrtop.1,wcrmid.1]                                  !wing cap right top gridded
cay wcrbg [wcrmid.1,wcrbot.1]                                  !wing cap right bot gridded

!-------------------------------------------------------------------------------
!--------------------------CREATE DATA GATHER NETWORK n WAKES----------------------

CST GT [(-.5,3.75,1.2+vnhei),(5.0,3.75,1.2+vnhei)]
CST GB [(-.5,3.75,-.1),(5.0,3.75,-.1)]
cay DGN [GT,GB]
fsu dgs dgn
grd dgs dgg 150 LIN LIN 150 LIN LIN

CST GT2 [((vnle*vnlen+4.0),3.0,0.5+vnhei),((vnle*vnlen+4.0),2.0,0.5+vnhei)]
CST GB2 [((vnle*vnlen+4.0),3.0,-.5+vnhei),((vnle*vnlen+4.0),2.0,-.5+vnhei)]
cay DGN2 [GT2,GB2]
fsu dgs2 dgn2
grd dgs2 dgg2 100 LIN LIN 100 LIN LIN

CST GT3 [((vnle*vnlen+0.2),3.0,0.9+vnhei),((vnle*vnlen+0.2),2.0,0.9+vnhei)]
CST GB3 [((vnle*vnlen+0.2),3.0,-.1+vnhei),((vnle*vnlen+0.2),2.0,-.1+vnhei)]
cay DGN3 [GT3,GB3]
fsu dgs3 dgn3
grd dgs3 dgg3 100 LIN LIN 100 LIN LIN

!-----------------------REVERSE NORMALS-------------------

rvs wltg
rvs wlbg
rvs wrtg
rvs wrbg

rvs wcltg
```

```
rvs wclbg
rvs wcrtg
rvs wcrbg

rvs ng

rvs pcrg

!ARF wlvw wlvwU

!-------------------------------------------------------------------------------
!------------------------------MAKE VISCOUS WAKES--------------------------------

TRN ng1 ng.L (0,0,0)
TRN ng2 ng.L (.2,0,0)
TRN ng3 ng.L (.4,0,0)
TRN ng4 ng.L (.6,0,0)
TRN ng5 ng.L (.8,0,0)
TRN ng6 ng.L (1.0,0,0)
TRN ng7 ng.L (1.2,0,0)
TRN ng8 ng.L (1.4,0,0)
TRN ng9 ng.L (1.6,0,0)
TRN ng10 ng.L (1.8,0,0)
CAY ngvwU [ng1,ng2,ng3,ng4,ng5,ng6,ng7,ng8,ng9,ng10]
ARF ngvw ngvwU                                          !because TranAir likes its wakes in a certain way

cst wrwv1 wrtg.*.L
TRN wrwv2 wrwv1 (.25,0,0)
TRN wrwv3 wrwv1 (.5,0,0)
TRN wrwv4 wrwv1 (.75,0,0)
TRN wrwv5 wrwv1 (1.0,0,0)
TRN wrwv6 wrwv1 (1.25,0,0)
TRN wrwv7 wrwv1 (1.5,0,0)
TRN wrwv8 wrwv1 (1.75,0,0)
TRN wrwv9 wrwv1 (2.0,0,0)
TRN wrwv10 wrwv1 (2.25,0,0)
CAY wrvwU [wrwv1,wrwv2,wrwv3,wrwv4,wrwv5,wrwv6,wrwv7,wrwv8,wrwv9,wrwv10]
ARF wrvw wrvwU                                          !because TranAir likes its wakes in a certain way

cst wlwv1 wltg.*.L
TRN wlwv2 wlwv1 (.25,0,0)
TRN wlwv3 wlwv1 (.5,0,0)
TRN wlwv4 wlwv1 (.75,0,0)
```

```
TRN wlwv5 wlwv1 (1.0,0,0)
TRN wlwv6 wlwv1 (1.25,0,0)
TRN wlwv7 wlwv1 (1.5,0,0)
TRN wlwv8 wlwv1 (1.75,0,0)
TRN wlwv9 wlwv1 (2.0,0,0)
TRN wlwv10 wlwv1 (2.25,0,0)
CAY wlvwU [wlwv1,wlwv2,wlwv3,wlwv4,wlwv5,wlwv6,wlwv7,wlwv8,wlwv9,wlwv10]
ARF wlvw wlvwU                                          !because TranAir likes its wakes in a certain way

cst wcltgw1 wcltg.L.*
TRN wcltgw2 wcltgw1 (0,-1,0)
CAY wcltgwake [wcltgw2,wcltgw1]

cst wcrtgw1 wcrtg.1.*
TRN wcrtgw2 wcrtgw1 (0,1,0)
CAY wcrtgwake [wcrtgw2,wcrtgw1]

!-------------------------------------------------------------------------------
!--------------------DO OPERATIONS TO HAPPIFY THE NETWORK ABUTMENTS----------------
cay wrbg2 [wlbg.L,wrbg.2:L]
ren wrbg2 wrbg SUPERSEDE=YES
!---------------merge points on the top left and top right networks---------------
!wrtg.1 and wrlg.L (aka 31)

!-------------------merge points in the pylon/wing top networks--------------------
call get_length(wrtg.1,n)
call get_length(plg.1,m)

!=========================================
J=fixer
!=========================================

!WINTRS0 S-value of start of pylon/wing intersection curve on WING surface
!WINTRS1 S-value of end of pylon/wing intersection curve on WING surface

I=n
call get_val(wltg.1.<I>,0,0,0,0,X,Y,Z)
call get_val(plg.6.<m>,0,0,0,0,psx,Y,Z)

!going down from L on the wing, going up from 1 on the pylon

$WHILE (X > (pwxe+.000)) DO                             !Find wing top location corresponding to pylon Traling
Edge
```

```
      call get_val(wltg.1.<I>,0,0,0,0,X,Y,Z)
      MAL wrtg.31 wrtg.31.<I> (X,Y,Z)                !Merge wing networks prior to that
      I=I-1
$ENDDO
!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!
I=I+2                                                !Adjust where the network merge starts
!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!
!=======================make carryover wake on wing=======================
TRN wcow1 wltg.1.<I>:<n> (0,0,0)
TRN wcow2 wltg.1.<I>:<n> (0,0,0)
cst wcow1s wcow1.*
cst wcow2s wcow2.*
cst wcow1s wlvw.1.2:L
cst wcow2s wlvw.1.2:L
CAY wcowU [wcow1s,wcow2s]

call get_val(wcowU.1.1,0,0,0,0,X,Y,Za)
call get_length(wcowU.1,n)
$FOR wi=1 TO n STEP 1 DO
      call get_val(wcowU.1.<wi>,0,0,0,0,X,Y,Z)
      MAL wcowU.2 wcowU.2.<wi> (X,Y,Za)              !Merge wing networks prior to that
$ENDDO

!ARF wcow wcowU                                      !because TranAir likes its wakes in a certain way
!=============================================================================

call get_val(wltg.1.<I>,0,0,0,0,X,Y,Z)
era *
dra wrtg
dra wltg
dra plg
dra prg
dra [(X,Y,Z)] color=RED SYMBOLS=10 era=N

$WRITE m

!pause

$FOR K=m to 1 STEP -1 do                                !For all points in the pylon network do...
      call get_val(plg.6.<K>,0,0,0,0,X,Y,Z)
      MAL wltg.1 wltg.1.<I> (X,Y,Z)
```

```
      call get_val(prg.1.<K>,0,0,0,0,X,Y,Z)
      MAL wrtg.31 wrtg.31.<I> (X,Y,Z)
      !dra [(X,Y,Z)] color=green SYMBOLS=10 era=N
      !$WRITE K
      I=I-1
      !pause
$ENDDO

$WHILE I >= 1 do
      call get_val(wltg.1.<I>,0,0,0,0,X,Y,Z)
      MAL wrtg.31 wrtg.31.<I> (X,Y,Z)
      I=I-1
$ENDDO

!-------------------merge points in the pylon/nacelle networks--------------------
call get_length(ng,n)
call get_length(plg.1,J)
!pause
I=1
$WHILE J >= 1 DO
      call get_val(ng.<I>.1,0,0,0,0,X,Y,Z)
      $IF (X>=(pnxs-.01)) and (X<=(pnxe+.01)) then
            IF I<=90 then
                  call get_val(prg.6.<J>,0,0,0,0,X,Y,Z)
                  MAL ng.<I> ng.<I>.1 (X,Y,Z)
                  !dra [(X,Y,Z)] color=orange SYMBOLS=10 era=N
                  call get_val(plg.1.<J>,0,0,0,0,X,Y,Z)
                  MAL ng.<I> ng.<I>.L (X,Y,Z)
                  !dra [(X,Y,Z)] color=green SYMBOLS=10 era=N
                  !pause
                  !draw ng 200 200 200 200 blue yes
                  ne = I                        !record the last nacelle point modified
            $endif
            J=J-1
      $else             !record value of pylon TE on the nacelle for the carryover wake procedure below
            ni = I + 1
      $endif
    I = I + 1
$ENDDO

!=======================make carryover wake on nacelle=======================
TRN ncow1 ng.<1>:<ni>.1 (0,0,0)
TRN ncow2 ng.<1>:<ni>.1 (0,0,0)
```

```
cst ncow1s ncow1.*
cst ncow2s ncow2.*
RVS ncow1s
RVS ncow2s
cst ncow1s ngvw.1.2:L
cst ncow2s ngvw.1.2:L
CAY ncowU [ncow1s,ncow2s]

call get_val(ncowU.1.1,0,0,0,0,X,Y,Za)
call get_length(ncowU.1,n)
$FOR ni=1 TO n STEP 1 DO
      call get_val(ncowU.2.<ni>,0,0,0,0,X,Y,Z)
      MAL ncowU.1 ncowU.1.<ni> (X,Y,Za)              !Merge wing networks prior to that
$ENDDO

ARF ncow ncowU                                       !because TranAir likes its wakes in a certain way
!=============================================================================

!=======================make pylon wake=======================
cst pvw1 prg.*.L
cst pvw2 [wcowU.2.l,ncowU.1.l]
fcv pvw2c pvw2 linear
CST pvw2 DATA=(.5) SPACE=pvw2c I=pvw2.2              !third point..
CAY pvw [pvw1,pvw2]
draw pvw 1 1 1 1 red no
fsu pvwSURF pvw linear
grd pvwSURF pvwakeU 3 LIN LIN 4 LIN LIN
ARF pvwake pvwakeU
!=============================================================================

era n*

!------------------merge points in nacelle self-joint-------------------
draw nacsurf 100 100 100 100 red no

call get_length(plg.1,J)
call get_length(ng,n)

$FOR I=1 to n do
      call get_val(ng.<I>.L,0,0,0,0,X,Y,Z)
      $IF (I<=(ne-J)) or (I>ne) then
            call get_val(ng.<I>.1,0,0,0,0,X,Y,Z)
            MAL ng.<I> ng.<I>.L (X,Y,Z)
```

```
            !dra [(X,Y,Z)] color=orange SYMBOLS=10 era=N
            !pause
      $endif
$ENDDO

!-------------------split nacelle into inner and outer-------------------
cay ngo ng.1:72
cay ngi ng.72:1
!was 65
!-------------------------------------------------------------------------------
!------------------------CREATE ROUNDED-OFF WING CAPS---------------------------
!-------------------------------------------------------------------------------
!-------------------------------------RIGHT CAP---------------------------------
call get_length(wrtg.1,n)
cst rwcR1 wrtg.1.*
cst rwcR2 wrtg.1.*
cst rwcR3 wrtg.1.*
cst rwcR4 wrtg.1.*
cst rwcR5 wrtg.1.*
cst rwcR6 wrtg.1.*
cst rwcR7 wrbg.L.*
CAY rwcRu [rwcR1,rwcR2,rwcR3,rwcR4,rwcR5,rwcR6,rwcR7]       !Rounded Wing Cap Right

$FOR I=1 to n-1
      call get_val(wrtg.1.<I>,0,0,0,0,Xt,Yt,Zt)
      call get_val(wrbg.L.<I>,0,0,0,0,Xb,Yb,Zb)
      cz = (Zt+Zb)/2                                          !center in z
      cy = 7.50
      rad = (Zt-Zb)/2                                         !Radius
      $FOR J=1 to 5                                           !For each of 7 strings...
            Y=(sin(((J)*(180/6))/57.2957795)*rad+cy)
            Z=(cos(((J)*(180/6))/57.2957795)*rad+cz)
            X=Xt+((Xb-Xt)*(J/6))
            MAL rwcRu.<J+1> rwcRu.<J+1>.<I> (X,Y,Z)
            !dra [(X,Y,Z)] color=orange SYMBOLS=10 era=N
            !pause
      $ENDDO
$ENDDO
!-------------------------------------------------------------------------------
!-------------------------------------LEFT CAP----------------------------------
call get_length(wltg.1,n)
cst rwcL1 wltg.L.*
cst rwcL2 wltg.L.*
```

```
cst rwcL3 wltg.L.*
cst rwcL4 wltg.L.*
cst rwcL5 wltg.L.*
cst rwcL6 wltg.L.*
cst rwcL7 wlbg.1.*
CAY rwcL [rwcL1,rwcL2,rwcL3,rwcL4,rwcL5,rwcL6,rwcL7]        !Rounded Wing Cap Right

$FOR I=1 to n-1
      call get_val(wltg.1.<I>,0,0,0,0,Xt,Yt,Zt)
      call get_val(wlbg.L.<I>,0,0,0,0,Xb,Yb,Zb)
      cz = (Zt+Zb)/2                                        !center in z
      cy = 0.0
      rad = (Zt-Zb)/2                                       !Radius
      $FOR J=1 to 5                                         !For each of 7 strings...
            Y=(-sin(((J)*(180/6))/57.2957795)*rad+cy)
            Z=(cos(((J)*(180/6))/57.2957795)*rad+cz)
            X=Xt+((Xb-Xt)*(J/6))
            MAL rwcL.<J+1> rwcL.<J+1>.<I> (X,Y,Z)
            !dra [(X,Y,Z)] color=orange SYMBOLS=10 era=N
            !pause
      $ENDDO
$ENDDO
!----------------------------------FIX CAP ABBUTMENTS----------------------------

!---------------------------------------DRAW THEM---------------------------------
ARF rwcR rwcRu
dra rwcR
dra rwcL
!pause

!---------------------------------DRAW THEM ALL-------------------------------------
era *

draw wltg 0 0 0 0 BLUE no                !bad parameterization
!draw wlbg 0 0 0 0 RED no                !bad parameterization
draw wrtg 0 0 0 0 GREEN no               !bad parameterization
!draw wrbg 0 0 0 0 YELLOW no             !bad parameterization

draw plg 75 75 75 75 GREEN no            !bad parameterization
draw prg 75 75 75 75 YELLOW no           !bad parameterization

!draw wcltg 75 75 75 75 BLUE no          !bad parameterization
!draw wclbg 75 75 75 75 RED no           !bad parameterization
```

```
!draw wcrtg 75 75 75 75 GREEN no          !bad parameterization
!draw wcrbg 75 75 75 75 YELLOW no         !bad parameterization

draw rwcR 75 75 75 75 YELLOW no           !bad parameterization
draw rwcL 75 75 75 75 YELLOW no           !bad parameterization

!draw pcrg 70 70 70 70 YELLOW no          !bad parameterization
!draw pclg 70 70 70 70 YELLOW no          !bad parameterization
draw ng 200 200 200 200 BLUE no           !bad parameterization
!draw ngi 200 200 200 200 blue no         !bad parameterization
!draw rwgc.* 100 100 100 100 red no
!draw dgg 70 70 70 70 GREEN no

@C:\Program Files\Calmar Research Corporation\AGPS20.10\pack\com\trandc.com(KT01=[wrtg, wrbg, wltg, wlbg,
rwcR, rwcL, plg, prg, ng],KTX=[wcowU,ncowU],KT14=[pvwake, ngvw, wrvw, wlvw],NDC=7,CID=g3)

dra ngvw 100 100 100 100 red no
dra wlvw 100 100 100 100 red no
dra wrvw 100 100 100 100 red no

dra pvwake 1 1 1 1 red no
draw ncowu 1 1 1 1 red no
draw wcowu 1 1 1 1 red no

draw (0,2.9,-.1) 1 1 1 1 RED NO
draw (2.5,2.9,-.1) 1 1 1 1 RED NO
draw (0,4.6,-.1) 1 1 1 1 RED NO
draw (2.5,4.05,-.1) 1 1 1 1 RED NO
draw (0,2.9,2) 1 1 1 1 RED NO
draw (2.5,2.9,2) 1 1 1 1 RED NO
draw (0,4.6,2) 1 1 1 1 RED NO
draw (2.5,4.6,2) 1 1 1 1 RED NO

!draw (1.5,3.45,0) 1 1 1 1 BLUE NO
!draw (4.0,3.45,0) 1 1 1 1 BLUE NO
!draw (1.5,4.05,0) 1 1 1 1 BLUE NO
!draw (4.0,4.05,0) 1 1 1 1 BLUE NO
!draw (1.5,3.45,.25) 1 1 1 1 BLUE NO
!draw (4.0,3.45,.25) 1 1 1 1 BLUE NO
!draw (1.5,4.05,.25) 1 1 1 1 BLUE NO
!draw (4.0,4.05,.25) 1 1 1 1 BLUE NO

del (xb,yb,zb,xt,yt,zt,zc,xc,pang)
```

```
call get_val(plg.1.1,0,0,0,0,Xb,Yb,Zb) !pylon bottom
call get_val(plg.1.1,0,0,0,0,Xt,Yt,Zt) !pylon top

zc = Zt-Zb
xc = Xt-Xb

pang=90-ATAN(xc/zc)*57.2957795

$WRITE pang
```

Appendix E – AGPS Post-Processing Code

```
!DONT FORGET TO PUT THE DGG AXIS IN THE CORRECT SPOT
del *
r23 'g3.net' LIST-ID=NETS

vnaxz = 0.2625                    !nacelle symmetry axis Z location (we know X and Y is set at 2.5)
vndia = 0.35                !nacelle diameter
vpsum = 0                   !summary pressure
vnpts = 0                   !number of pressure points being averaged
vavep = 0                   !average pressure through the nacelle

vpsumq = 0                  !summary q
vnptsq = 0                  !number of q points being averaged
vaveq = 0                   !average pressure through the nacelle

GGP g3.ggp PLOC SEL=X,Y,Z
GGP g3.ggp PRESS SEL=CP
GGP g3.ggp MASSFLO SEL=WN
GGP g3.ggp VEL_U SEL=U

COP NAME=PRESS1 OLD=[PRESS.1, PRESS.2, PRESS.3, PRESS.4, PRESS.5, PRESS.6, PRESS.7, PRESS.8, PRESS.9,
PRESS.10, PRESS.11, PRESS.12, PRESS.13, PRESS.14, PRESS.15, PRESS.16, PRESS.17, PRESS.18, PRESS.19,
PRESS.20, PRESS.21, PRESS.22, PRESS.23]
COP NAME=LOCSS1 OLD=[PLOC.1, PLOC.2, PLOC.3, PLOC.4, PLOC.5, PLOC.6, PLOC.7, PLOC.8, PLOC.9, PLOC.10,
PLOC.11, PLOC.12, PLOC.13, PLOC.14, PLOC.15, PLOC.16, PLOC.17, PLOC.18, PLOC.19, PLOC.20, PLOC.21, PLOC.22,
PLOC.23]
COP NAME=NETS1 OLD=[NETS.1, NETS.2, NETS.3, NETS.4, NETS.5, NETS.6, NETS.7, NETS.8, NETS.9, NETS.10,
NETS.11, NETS.12, NETS.13, NETS.14, NETS.15, NETS.16, NETS.17, NETS.18, NETS.19, NETS.20, NETS.21, NETS.22,
NETS.23]
COP NAME=MFLO1 OLD=[MASSFLO.1, MASSFLO.2, MASSFLO.3, MASSFLO.4, MASSFLO.5, MASSFLO.6, MASSFLO.7, MASSFLO.8,
MASSFLO.9, MASSFLO.10, MASSFLO.11, MASSFLO.12, MASSFLO.13, MASSFLO.14, MASSFLO.15, MASSFLO.16, MASSFLO.17,
MASSFLO.18, MASSFLO.19, MASSFLO.20, MASSFLO.21, MASSFLO.22, MASSFLO.23]
COP NAME=VELU OLD=[VEL_U.1, VEL_U.2, VEL_U.3, VEL_U.4, VEL_U.5, VEL_U.6, VEL_U.7, VEL_U.8, VEL_U.9,
VEL_U.10, VEL_U.11, VEL_U.12, VEL_U.13, VEL_U.14, VEL_U.15, VEL_U.16, VEL_U.17, VEL_U.18, VEL_U.19,
VEL_U.20, VEL_U.21, VEL_U.22, VEL_U.23]

P3D  DISPLAY1  NETS1 PRESS1 1
LEG DISPLAY1.1 MIN=-.6 MAX=.1
EXE  DISPLAY1  ERA=N

!dft press1
```

```
!dft locss1

pause
P3D  DISPLAY2  locss1.21 PRESS1.21 1
LEG DISPLAY2.1 MIN=-.6 MAX=.1
EXE  DISPLAY2  ERA=Y

pause

SCL MFLUX MASSFLO.21 (-1)

P3D  DISPLAY3  locss1.21 MFLUX 1
LEG DISPLAY3.1 MIN=1.25 MAX=.95
EXE  DISPLAY3  ERA=Y

pause

P3D  DISPLAY4  locss1.21 VELU.21 1
LEG DISPLAY4.1 MIN=.9 MAX=1.3
EXE  DISPLAY4  ERA=Y

call get_length(locss1.21,m)
call get_length(locss1.21.1,n)

$FOR I=1 TO n DO
      $FOR J=1 TO m DO
            call get_val(locss1.21.<I>.<J>,0,0,0,0,X,Y,Z)
          varm = (((2.5-Y)^2+(vnaxz-Z)^2)^0.5)
        $IF varm < (vndia/2) then                              !means we're evaluating a point inside the
nacelle circle
            call get_val(PRESS1.21.<I>.<J>,0,0,0,1,A)
            $IF A<>0 THEN                                            !and to make sure we're not counting one
of the internal areas the plane intersects
            vpsum = vpsum + A
                vnpts = vnpts + 1
            $ENDIF
            call get_val(MFLO1.21.<I>.<J>,0,0,0,1,B)
            $IF B<>0 THEN                                            !and to make sure we're not counting one
of the internal areas the plane intersects
            vpsumq = vpsumq + B
                vnptsq = vnptsq + 1
            $ENDIF
        $ENDIF
```

```
      $ENDDO
$ENDDO

vavep = vpsum/vnpts
vaveq = vpsumq/vnptsq
```

Appendix F – Tran Air Input File (less the .poi definitions)

```
$TITle of the project file
0012 WING
GENE GISIN MASTERS THESIS
$CAS
1.
$MACh number
=amach    =linr     =amfgas   =fsvmi    =fviscm   =fvisco   =rholim
0.75                                    1.        2.0
$SYMmetry
=xzsym    =yzsym
0.        0.
$ITErations
=niter    =tol      =ncdamp   =dropt    =nsrch    =njac
1500.     5.        5.        .0005     60.       5.
$DAT                                                                    dat1
=ndtchk                     igeom     ipcut     ngstop    ndstop      dat2
0.        0.        0.      1.        1.        0.
$BOX the geometry is contained in
-2.5      9.0       28.
-2.0      13.       21.
-2.5      7.5       16.
$TOLerance
=adpfac   panfac    dxmin     dxmax     epsf      noptgg
1.        -1.0      0.15      50.       1.51      1.
=========================================================
$SUR
=ipggp    lvlout
4.0       0.0
*NETwork
=netbeg   netend    isurf
*PARameter, flow properties
CP        MACH      RHO       U         V         W
============================================================
$LBO Grid Refinement for Special Region                                 lbo1
!adpfac   panfac    dxmin     dxmax     lbnam                           lbo2
1.0       2.0       0.2       .5        LBOSUR
!x1       y1        z1        reldx1                              lbo3
-2.5      0.2       -2.5      1.0
```

```
!x2        y2         z2         reldx2                                                  lbo4
3.5        0.2        -2.5       1.0
!x3        y3         z3         reldx3                                                  lbo5
-2.5       9.8        -2.5       1.0
!x4        y4         z4         reldx4                                                  lbo6
3.5        9.8        -2.5       1.0
!x5        y5         z5         reldx5                                                  lbo7
-2.5       0.2        3.0        1.0
!x6        y6         z6         reldx6                                                  lbo8
3.5        0.2        3.0        1.0
!x7        y7         z7         reldx7                                                  lbo9
-2.5       9.8        3.0        1.0
!x8        y8         z8         reldx8                                                  lbo10
3.5        9.8        3.0        1.0
=============================================================
$LBO Grid Refinement for Special Region                                                  lbo1
!adpfac    panfac     dxmin      dxmax      lbnam                                        lbo2
1.0        -2.0       0.2        .5         LBOVOL
!x1        y1         z1         reldx1                                                  lbo3
-2.5       0.2        -2.5       1.0
!x2        y2         z2         reldx2                                                  lbo4
3.5        0.2        -2.5       1.0
!x3        y3         z3         reldx3                                                  lbo5
-2.5       9.8        -2.5       1.0
!x4        y4         z4         reldx4                                                  lbo6
3.5        9.8        -2.5       1.0
!x5        y5         z5         reldx5                                                  lbo7
-2.5       0.2        3.0        1.0
!x6        y6         z6         reldx6                                                  lbo8
3.5        0.2        3.0        1.0
!x7        y7         z7         reldx7                                                  lbo9
-2.5       9.8        3.0        1.0
!x8        y8         z8         reldx8                                                  lbo10
3.5        9.8        3.0        1.0
=============================================================
$REFerence Lengths and Areas                                                             ref1
!xref      yref       zref                                                               ref2
0.         0.         0.
!sref      bref       cref       zref                                                    ref3
25.        1.0        1.0        1.0
=============================================================
$CLMatching                                                                              clm1
!cl           grid0     ndamp                                    option    Evcfg     clm2
```

```
0.4       1.        0.0                                        1.0       FULLCONFIG
!fm       grid0     ndamp                            FMname    option    EVcfg     clm3
!0.4       1.        0.0                              CL        0.        FULLCONFIG
================================================================
$ADAptive Grid Refinement
=ncycle   nbxsur
10.       3.
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf                Cycle 1, makes GRID 2
1.0       9000.     -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
0.0
=adpfac   panfac    dxmin     dxmax     lbnam
!NONE
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf                Cycle 2, makes GRID 3
1.0       10000.    -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
0.0
=adpfac   panfac    dxmin     dxmax     lbnam
!NONE
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf                Cycle 3, makes GRID 4
1.0       22000.    -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
2.0
=adpfac   panfac    dxmin     dxmax     lbnam
2.00      2.        0.15      1.05      LBOSUR
2.00      -2.       0.15      89.       LBOVOL
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf                Cycle 4, makes GRID 5
1.0       24200.    -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
```

```
0.06      0.01
=numlbo
0.0
=adpfac   panfac    dxmin     dxmax     lbnam
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf              Cycle 5, makes GRID 6
1.0       50000.    -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
2.0
=adpfac   panfac    dxmin     dxmax     lbnam
2.00      2.        0.077     .5        LBOSUR
2.00      -2.       0.077     39.       LBOVOL
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf              Cycle 6, makes GRID 7
1.0       55000.    -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
0.0
=adpfac   panfac    dxmin     dxmax     lbnam
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf              Cycle 7, makes GRID 8
1.0       110000.   -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
2.0
=adpfac   panfac    dxmin     dxmax     lbnam
2.00      2.        0.038     0.5       LBOSUR
2.00      -2.       0.038     20.       LBOVOL
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf              Cycle 8, makes GRID 9
1.0       120000.   -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
0.0
```

```
=adpfac   panfac    dxmin     dxmax     lbnam
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf                Cycle 9, makes GRID 10
1.0       240000.   -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
2.0
=adpfac   panfac    dxmin     dxmax     lbnam
2.00      2.        0.0193    0.5       LBOSUR
2.00      -2.       0.0193    20.       LBOVOL
=maxgrd   nbxtgt    ratio1    ratio2    fracrf    fracdf                Cycle 10, makes GRID 11
1.0       270000.   -1.       0.0       0.2       0.4
=ecutrp   ecutdp
-0.8      10000.0
=ecutr    ecutd
0.06      0.01
=numlbo
0.0
=adpfac   panfac    dxmin     dxmax     lbnam
============================================================
$BOUndary layer                                              bou1
============================================================
=    reinf      tinf    fomgbl   blfnopt    blnewt    nblcod     ifstf          bou2
     4.E+6    300.20       0.3        0.        7.        2.        1.
=       xt    dmpsep        pr       prt      eta2    etamax                   bou3
    0.0000      0.00      0.71      0.90    0.0001       15.
=   fngrpl    namdau    namdal     namde    namdax    namdex                   bou4
!fngrpl stands for number of BL ribs in the problem
        2.
=    gfacu     pfacu     eastu     finvu     jetau     profu                   bou5
        0.      0.30        1.        0.       40.        0.
=  blndm1u   blndm2u   blndr1u   blndr2u                                        bou6
        0.        0.        0.        0.
=   fwedgu    fwdg1u    fwdg2u   fblnd1u   fblnd2u     frufu                   bou7
        1.        1.        1.        1.        1.
=    gfacl     pfacl     eastl     finvl     jetal     profl                   bou8
        0.      0.30        1.        0.       40.        0.
=  blndm1l   blndm2l   blndr1l   blndr2l                                        bou9
        0.        0.        0.        0.
=   fwedgl    fwdg1l    fwdg2l   fblnd1l   fblnd2l     frufl                   bou10
        1.        1.        1.        1.        1.
```

```
=     nrib   coupflg     crefg    fatlin    fbltyp    fblini    namgrp          bou11
        2.        1.    1.0000        0.        1.        0.    BL_WING
=    nnetr     swple     swpte      xctl      xctu    htripl    htripu    iflcdpbou13
      1.0        0.        0.      0.02      0.04        0.        0.        0.
=      net     colst     rowst    colend    rowend              cuplow          bou14
        WL       13.        1.       13.       81.
=    nnetr     swple     swpte      xctl      xctu    htripl    htripu    iflcdpbou13
      1.0        0.        0.      0.02      0.04        0.        0.        0.
=      net     colst     rowst    colend    rowend              cuplow          bou14
        WR       13.       81.       13.        1.
!----------------------next rib group here-----------------------------
=     nrib   coupflg     crefg    fatlin    fbltyp    fblini    namgrp          bou11
        2.        1.    1.0000        0.        2.        0.    BL_NACELLE
=       x1        y1        z1        x2        y2        z2                    bou12
        0.      3.75      0.40        2.      3.75      0.40
=    nnetr     swple     swpte      xctl      xctu    htripl    htripu    iflcdpbou13
      1.0        0.        0.      0.04      0.04        0.        0.        0.
=      net     colst     rowst    colend    rowend              cuplow          bou14
       NG       101.       25.        1.       25.
=    nnetr     swple     swpte      xctl      xctu    htripl    htripu    iflcdpbou13
      1.0        0.        0.      0.04      0.04        0.        0.        0.
=      net     colst     rowst    colend    rowend              cuplow          bou14
       NG       101.       75.        1.       75.
============================================================
$FORces and Moments                                                    for1
!iclopt (optional)                                                     for2
*CONfiguration
      for3
!netbeg   netend    isurf     iggpop    icalp     icalmo    icalsf    idfpid    for4
1.0       5.0       0.        0.        1.0       0.0       0.0       FULLCONFIG
============================================================
$FIL
$CASE.poi
$FIE
=npbnp    npbnm     npbnr     npbnv     npbne     npbnq     lstbng
0.0       1.0       0.0       0.0       0.0       1.0
$EAT
!epsgeo                     nwxref              iabsum
0.0       0.        0.        0.        0.        0.
============================================================
============================================================
$TRAiling wakes
============================================================
```

```
=kn
3.
=kt
18.0
=inat     insd      xwake                                          netname
WRVW      3.        24.                                            WAKER
WLVW      3.        24.                                            WAKEL
NGVW      3.        24.                                            NWAKE
$END
```