To the Graduate Council:

I am submitting herewith a dissertation written by Jyh-Chyang Alex Wang entitled "A Scalar/Vector Potential Solution for Aerodynamic Coefficients in Wind Shear." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Aerospace Engineering.

Walter Frost

Walter Frost, Major Professor

We have read this dissertation
and recommend its acceptance:

Boris Kr

J Wu

John E. Caruthers

Accepted for the Council:

Vice Provost
and Dean of the Graduate School

A SCALAR/VECTOR POTENTIAL SOLUTION FOR AERODYNAMIC COEFFICIENTS IN WIND SHEAR

A Dissertation
Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Jyh-Chyang Alex Wang

December 1988

ACKNOWLEDGMENTS

The work reported herein was partially supported by the NASA Langley Research Center, under contract number NAS1-18344.

The author is deeply grateful to his advisor, Dr. Walter Frost, for his suggestion of the topic of this study and for his help in completing the research. Sincere appreciation is given to Dr. T. H. Moulden for his constant encouragement and helpful insight to this work.

The author is further indebted to other committee members, Dr. John Caruthers, Dr. B. A. Kupershmidt, and Dr. J. M. Wu for their review of the present work. Sincere thanks are expressed to Mrs. Mary Lo, head librarian, for her great assistance to the author's research effort. Thanks are also extended to Mr. Marvis White for diligent perusal of the manuscript, and Mr. Robert Warren Suttles and Ms. Martha Annes for preparing the illustrations in this study. Special appreciations are offered to associate dean, Dr. R. M. Roberds, and to the computer center of The University of Tennessee Space Institute.

Sincerest gratitude is given to the author's family, especially to his father, Mr. Chi-Min Wang, for his love and support. The author is deeply indebted to his beloved wife, Min-Huei Luo, for sharing joys and sorrows. Without their encouragement, I could not have completed my study toward the degree of Doctor of Philosophy in the United States.

ABSTRACT

A recent innovation in aerodynamic calculation techniques is a formulation for the velocity field based upon scalar and vector potentials. This technique is presented and implemented into the panel method such that it is able to solve not only irrotational flow but also rotational flow. The application of interest is to two-dimensional airfoils moving into a nonuniform approach flow. Comparisons with theoretical and numerical results are included. Furthermore, the variations of lift and moment coefficients of quasi-steady simulated flight of an airfoil through JAWS wind shear are also studied.

Although the present study does demonstrate the hazard of microbursts, the indicated wind shear effects on aerodynamic coefficients are not as great as expected. The present method only indicates the decrease of the effective angle of attack as an important factor. However, in a wind field with intense downdrafts, viscous effects and flow separation should also be crucial. Although the present inviscid model has shown great advantages as a preliminary study of the wind shear effects on aerodynamic coefficients, it cannot predict viscous effects and flow separation phenomena. Therefore, further investigation is recommended to implement viscous effects.

For potential flow calculations, comparable accuracy to that obtainable from higher-order panel methods can be achieved by the present formulation using the same number of control points. The present conceptual split of the onset flow into two components, $\vec{\mathbf{V}}_{0_n}$ normal and $\vec{\mathbf{V}}_{0_t}$ tangential to the zero lift line, is new. The proper vorticity distribution can be determined from $-\vec{\mathbf{V}}_{0_n} \cdot \vec{\mathbf{n}}$ directly without imposing an artificial vorticity distributor. This split also supplies a numerically superior form for the Neumann-type lower-order formulation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

$\vec{V}$	composition velocity field
V_{S_i}	tangential speed on each panel
$\vec{V}_o$	onset velocity field
$\vec{V}_{o_n}$	onset velocity perpendicular to the zero-lift line
$\vec{V}_{o_t}$	onset velocity along the zero-lift line
U	velocity
U_o	undisturbed stream velocity at $y = 0$
V_{air}	absolute airfoil velocity
$\vec{u}$	perturbed velocity field
$\vec{u}_1$	perturbed irrotational velocity field
$\vec{u}_2$	perturbed solenoidal velocity field
$\vec{\zeta}$	present vorticity field
$\vec{\zeta}_o$	onset vorticity field
$\vec{\zeta}'$	perturbed vorticity field
$\mathbf{p}$	static pressure
$\mathbf{p}_o$	static pressure at far upstream
C_p	specific pressure coefficient
C_l	lift coefficient
C_m	leading-edge moment coefficient
ρ	density
Y	pressure force in y direction
ϕ	scalar potential
$\vec{\psi}$	vector potential
σ	source strength

μ	doublet strength
γ	vortex strength
G, G_i	Green's function
x, y, z	cartesian coordinates
$\bar{x}, \bar{y}, \bar{z}$	local coordinates w. r. t. each panel
$\vec{\mathbf{x}}$	position vector
$\vec{n}_i$	normal on the ith panel
$\vec{t}_i$	tangential on the ith panel
c	reference chord length
α	angle of attack
α_{ZL}	zero lift angle
k	local stream shear, $(c/U_o)(dU/dy)$
H	transformation length
q	parameter defining nonuniform shear
$\mathbf{I^n}$	normal influence matrix due to source distribution
$\mathbf{D^n}$	normal influence matrix due to vortex distribution
$\mathbf{I}^t$	tangential influence matrix due to source distribution
$\mathbf{D}^t$	tangential influence matrix due to vortex distribution
Γ	circulation
Γ'	circulation due to perturbed velocity
$\vec{\mathbf{R}}_1$	right-hand-side of the system of equations for source
$\vec{\mathbf{R}}_2$	right-hand-side of the system of equations for vortex
λ_i	eigenvalue
$\vec{\mathbf{X}}_\mathrm{i}$	eigenvector
A	solid angle

CHAPTER I

INTRODUCTION

A. OBJECTIVE

The purpose of this study is to investigate the effects of a variable wind field on the aerodynamics of a body. The classical theory of aerodynamics usually assumes that the body is immersed in a uniform flow field. But in many engineering applications the approach flow is not uniform. An important example is that of an airplane taking-off or landing in a strong wind shear. This problem has recently received considerable attention because of the numerous aircraft incidents and accidents associated with severe microburst wind fields (Aviation Week & Space Technology, August 12, 1985 and September 22, 1986). Microburst wind fields are severe downdrafts accompanied by intense outflows (see Figure 1.1). It is strong wind shears of the microburst type which has prompted the present study. However, the general theory presented herein has applicability to many other areas such, for example, as the operation of helicopter rotor blades in the wakes of preceding blades. The specific examples presented are restricted to the two dimensional airfoil in an arbitrary wind field.

Wind shear is a spatial or temporal gradient in wind speed and/or direction. This variation can create changing tailwind, crosswind, and vertical wind as experienced by an aircraft in flight. Strong shears are generally associated with the presence of cold and warm fronts and with thunderstorms. Both fronts and microbursts induce low-level wind shear which endangers aircraft operations during the landing and takeoff phases of the flight envelope.

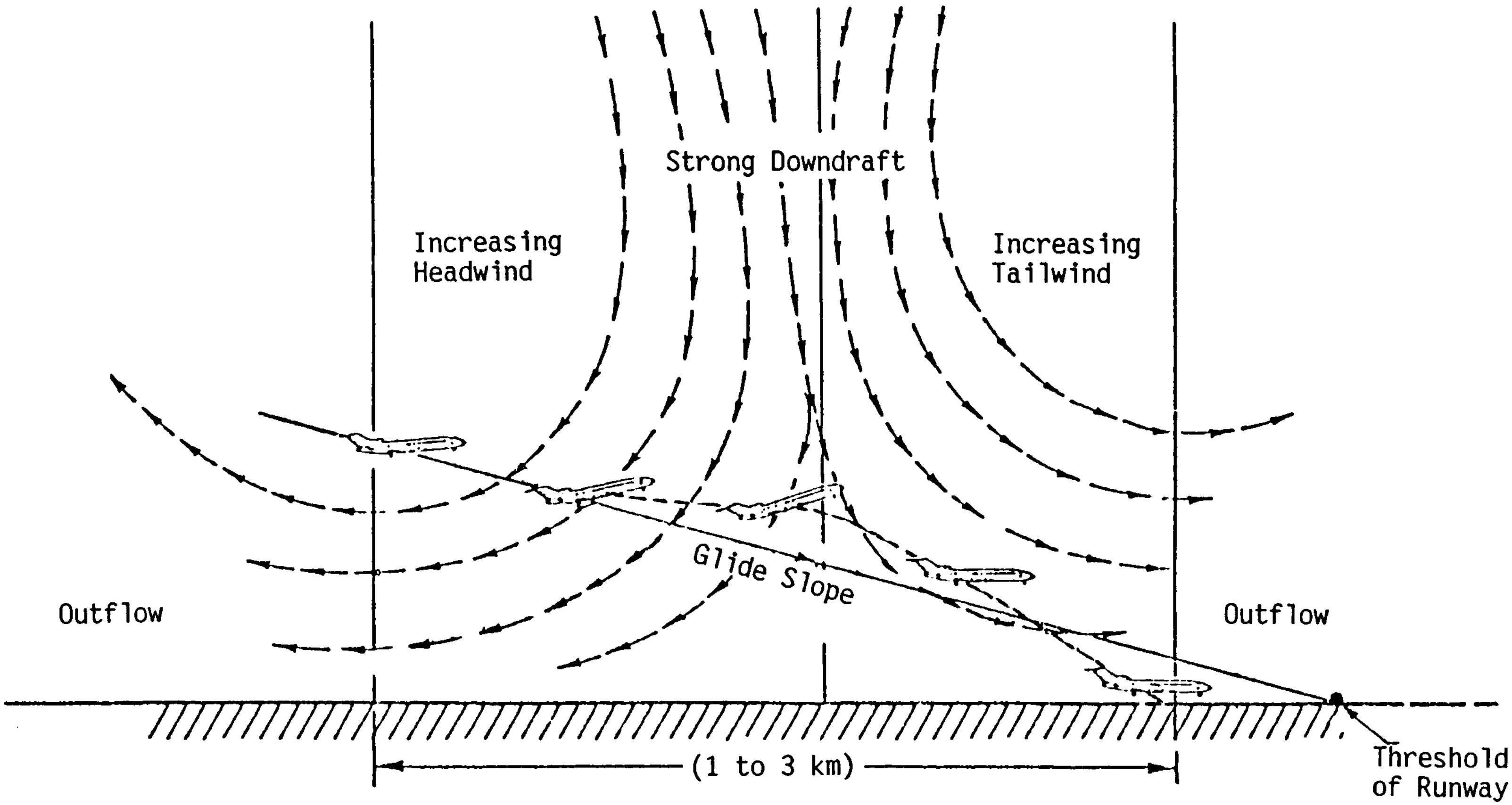


Figure 1.1. Schematic illustration of an aircraft encounter with a microburst [7].

Previous research relating to wind shear may be classified into two main categories. One category consists of meteorological analyses of the naturally occurring phenomenon itself such as its cause and prevalence [1,2]*, simulating wind shear with mathematical models [3,4], or employing multiple Doppler radar to measure the magnitude of shears as done in the Joint Airport Weather Studies (JAWS) [5]. The other category relates to aircraft control and performance such as implementing the wind shear terms into the equation of aircraft motion [6] and simulating the flight through microburst type wind shear [7], or determining the optimal flight paths through microbursts [8]. It is recognized that a principal cause for loss of lift is the rapid rotation of the wind approaching the aircraft and its attendant rapid change in effective incidences.

In the studies above the effects of variation of wind on the aerodynamic coefficients of the aircraft are not considered. To investigate what the magnitude of these effects may be, a computational technique is developed in the present study. Since most wind shear data consists of an array of the three wind speed components on a rectangular grid, the ultimate goal of the computational model was to solve for the aerodynamic forces on a body immersed in a volume element within the grid (see Figure 1.2). The wind field at any point within the volume element is computed from a volume weighted interpolation scheme. But the wind field between corner grids in Figure 1.2 is determined by linear interpolation. The ultimate goal is to solve the three-dimensional body in a three-dimensional non-uniform "onset" velocity flow field. A scalar/vector potential panel method is formulated for this three-dimensional, inviscid, incompressible, highly rotational flow problem. But solutions are only carried-out for the two-dimensional case. Besides, the quasi-steady state assumption is employed as follows: first, it is assumed

* Numbers in brackets refer to similarly numbered references in the Bibliography.

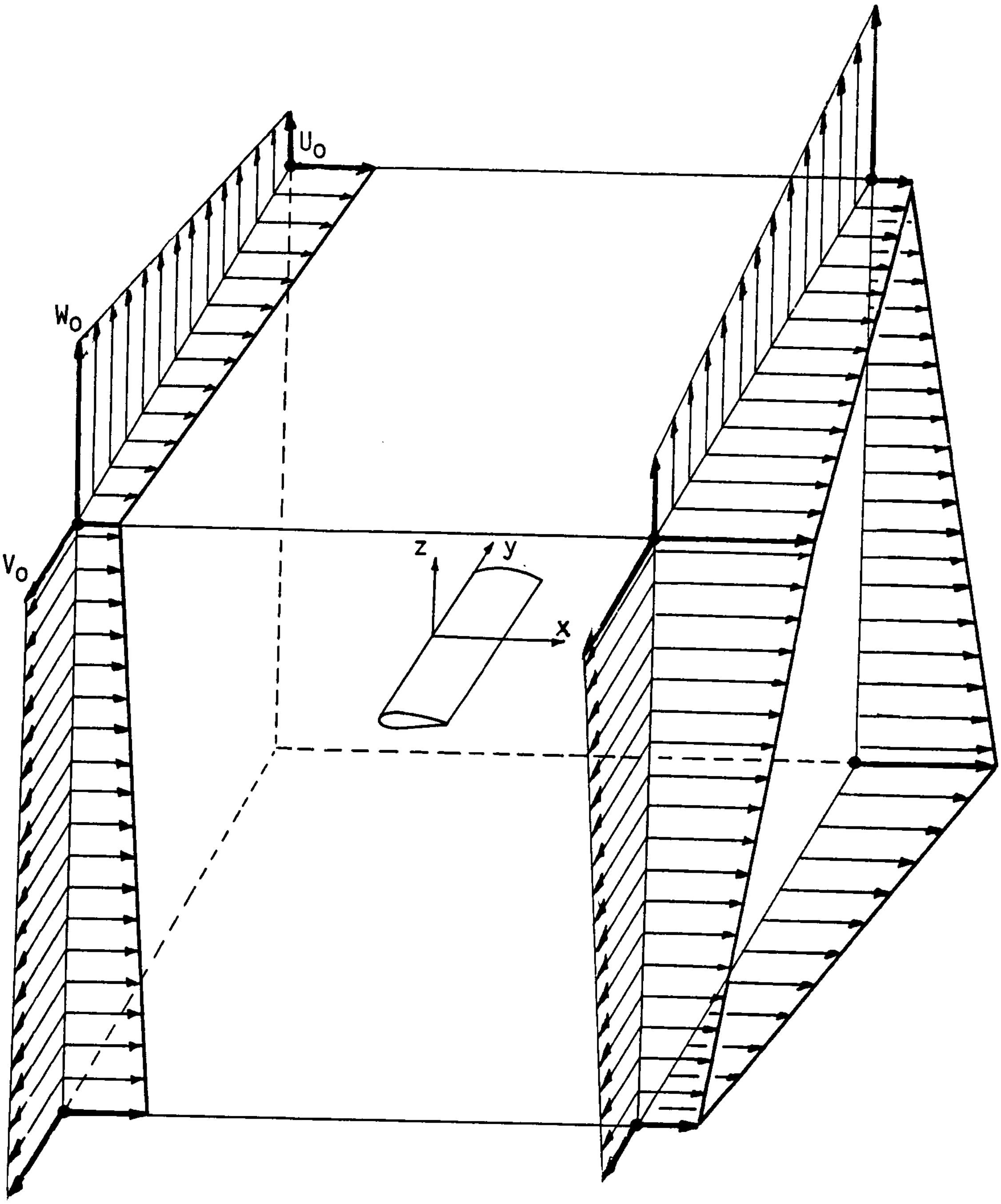


Figure 1.2. Schematic illustration of a wing in a volume element with a grid wind field determined either experimentally or computationally.

that the (input) wind field is "frozen", therefore each computational domain, or volume element, can be considered as a steady state. After calculating this wind field, a new wind field is input and "frozen" as before. Through this process, a continuous unsteady state can be broken into discrete quasi-steady states, which simplifies the problem. The model is believed to have more general applicability and accuracy and removes many of the restrictive assumptions of models developed by earlier researchers. The effects of wind shear on the coefficients of lift and moment of two-dimensional airfoils are investigated with the model for both theoretical and experimentally measured wind shears.

B. BACKGROUND

The problem of the influence of shear flow on the aerodynamics of bodies is not a new topic. It has existed virtually since the introduction of the airplane but only sporadic attention has been given to the problem. In the 1950s and 1960s, research related to the effect of shear flow on aerodynamic forces on a body sought either exact solutions or analytical solutions based on small perturbation assumptions. Since the 1970s, advances in digital computers have allowed computational methods to be applied to solutions of the problem for either inviscid or viscous flows. The following paragraphs briefly discuss the early progress in analytical and numerical methods for studying bodies in shear flow, and explain why the scalar/vector potential panel method of solution has been chosen for the present problem.

The effect of uniform shear flow was treated by von Sanden [9] in 1912. In his work, consideration was restricted to a wedge profile in a flow with a linear velocity gradient; that is, uniform shear. The same problem was taken up again by Tsien [10] in 1943, in connection with the influence of ground-level gradients in wind velocity on airfoil characteristics. Tsien considered a symmetrical Joukowski airfoil in a two-dimensional inviscid flow with uniform shear, and obtained an exact solution for the forces on the airfoil. In 1958, Sowyrda [11] extended Tsien's theory to the case of a cambered airfoil in uniform shear and obtained an exact solution for any thickness, camber, and angle of attack. Since the flow is two-dimensional and the onset vorticity is constant, the Joukowski transformation can be applied to the flow governing equations. This single fact makes it possible [10,11] to obtain exact solutions to these problems.

An understanding of the influence of nonuniform shear flows on airfoil characteristics can be obtained from the two-dimensional investigations made by Jones [12,13]. He considered a thin, cambered, Joukowski airfoil and an elliptic cylinder in flow with a parabolic and with a hyperbolic onset velocity distribution, respectively. This is in contrast to the previous uniform shear flow analyses for which the flow vorticity is a constant. From Helmholtz theorem, the vorticity must be conserved along a streamline ($D\zeta/Dt = 0$ for two dimensional ideal flow), thereby implying that the disturbance stream function does not satisfy the Laplace equation. This is the main reason that an exact solution for the nonuniform shear flow problem in closed form is difficult to find. Therefore, for nonuniform shear flow, small perturbation assumptions have to be made in order to linearize the governing equation.

To summarize the above, the existing theoretical treatment for airfoils in uniform or nonuniform shear flows has been applied only to two-dimensional flow in an unbounded field. The methods of treatment fall into two categories: exact solutions for airfoils in flows with uniform shear and small perturbation solutions for airfoils in flows with symmetric nonuniform shear. Both categories are constrained either to two-dimensional airfoils generated by conformal mapping, or to solution techniques which do not violate the small perturbation theory. From inspection of Figure 1.2 it is quite obvious that the above cited theoretical approaches cannot be applied to the present problem. However, these known theoretical results are valuable for checking the scalar/vector potential method developed in this study when it is applied to these simpler problems.

In the early 1970s, one group of researchers applied Fourier transformation and integral methods [14-17], and another group applied finite-difference and finite-element methods [18-20] to solve nonuniform (shear) flow problems. The different groups were interested in flow problems associated with panel flutter, propeller or jet slipstreams, body wakes, and boundary layers (including that generated in the atmosphere near the ground). The computational approach relaxed some of the assumptions required by the analytical approach, such as restrictions on airfoil geometry and the size of disturbances. However, several limitations still could not be removed; for example, linearized flow [14-17] and piecewise linear shear profiles [19]. Later in the 1980s, further finite-difference techniques were developed, but these in turn were limited to slender wings [21,22] and confined domains [23].

C. METHODOLOGY

For the reasons delineated in the preceding section, it is believed that a suitable computational methodology should be formulated and applied to the present nonlinear shear flow problem. Basically, there are two approaches for solving fluid flow problems. One is solving for the primitive or physical variables, such as velocity and pressure, etc. The other is to compute the scalar/vector potentials as, for example, in the vorticity-stream function method for two-dimensional flow, or the vorticity-potential method for three-dimensional flow.

In the approach based on primitive variables, there are two basic formulations to the problem: finite-difference and finite-element methods [24-26]. The major disadvantages of both of these relative to the scalar/vector potential panel method are: (1) the use of the Navier-Stokes or Euler equations requires the solution in the whole physical domain rather than just on the surface of the body and thus may be prohibited by the computer costs, (2) the incompressible viscous flow calculation lacks an evolution equation for the pressure (this difficulty does not exist, however, for compressible flow problems) [24].

To avoid some of the problems described above, a scalar/vector potential formulation is adopted in this study. Theoretical work on the scalar/vector potential method has been done [27-36]. However, only limited solutions of the formulated equations have been reported. Some examples include wind shear effects on windmills and buildings by Morino *et al* [37,38], rotational flow calculations in blade passages by Lacor and Hirsch [39] and flow in ducts by Chaviaropoulos *et al* [40], two-dimensional transonic rotational flow calculation by Chaderjian and Steger [41], and three-dimensional flow calculation for wind tunnels by Rao *et al* [42].

The advantage of using the vorticity stream function method for two dimensional flow is well known [24]. For example, the continuity equation is automatically satisfied. But in applying the vorticity stream function method based on either finite-difference or finite-element methods to solve the external flow problem, consumption of large amounts of computer memory space is inevitable. Since the panel method requires less memory space than other methods, a computer system with relatively small computer memory space, such as the VAX/11-785, can be used for the study of nonuniform shear flow over aerodynamic lifting bodies. Also the panel method is convenient for its simplicity and its versatility in solving flow problems in infinite space. The primary restriction of the panel method in previous studies is that it has been mainly used to solve irrotational (potential) flow problems. However, employing the classical Helmholtz decomposition technique, a scalar/vector potential approach using the panel method can be developed to solve external incompressible and rotational flow problems. The method is valid for either inviscid or viscous flow but is only developed for inviscid flow in the present study.

In the following sections, an introduction to the panel method and an implementation of the panel method for solution of the scalar/vector potential flow problem are given. The application of interest is to a two-dimensional airfoil embedded in a spatially nonuniform flow field. Comparisons with theoretical and numerical results, when available, are included. Finally, the computed variations of lift and moment coefficients for quasi-steady simulated flight of an airfoil through measured microburst wind shears are presented and discussed.

CHAPTER II

ON PANEL METHODS

Panel methods are useful for solving for aerodynamic forces on objects with complex geometry that are immersed in incompressible and inviscid flow fields. The method solves for a distribution of singularities (such as sources, doublets, or vortices) on the body surface. The main advantage of the panel method, compared with finite-difference and finite-element methods for external flow problems is that the "unknowns" are situated only on the surface of the configuration and not throughout the external space. This advantage enables very complex geometries to be handled with a reasonable number of unknowns. Furthermore, the panel method makes no approximations in its basic formulation as would be required for small perturbation or lifting-surface theories.

Although the panel method has been utilized for more than three decades, it has generally been restricted to the solution of potential flow problems, namely irrotational flow problems. The restriction to irrotational flow can be removed if a scalar/vector potential formulation is employed. This approach is carried out in the present study. However, because many of the basic concepts of the panel method applied to potential flow problems are the same as for the scalar/vector potential problem of rotational flow, the panel method as applied to potential flow problems is reviewed in the next section.

A. POTENTIAL FLOW PROBLEMS

To review the panel method as applied to the irrotational flow problem, consider the potential flow of an incompressible, inviscid fluid. Let $\vec{\mathbf{V}}$ denote the fluid velocity, $\mathbf{p}$ the fluid pressure, and ρ the fluid density. If the viscosity is set equal to zero and the density is taken as constant, the general Navier-Stokes equations reduce to the well-known Euler equation

$$\frac{\partial \vec{\mathbf{V}}}{\partial t} + (\vec{\mathbf{V}} \cdot \nabla) \vec{\mathbf{V}} = \vec{\mathbf{f}} - \frac{\nabla \mathrm{p}}{\rho} \tag{2.1}$$

and the continuity equation becomes

$$\nabla \cdot \vec{\mathbf{V}} = 0. \tag{2.2}$$

In Equation (2.1), the gravity force, which is a body force, has been assumed to be conservative with $\vec{\mathbf{f}} = \nabla U$, where U is a scalar function of position. Equations (2.1) and (2.2) can be solved for the flow field in a region R exterior to a body denoted by S (see Figure 2.1).

For nonpotential flow problem, the vorticity transport equation needs to be coupled into the calculation and it can be expressed as

$$\frac{D\vec{\zeta}}{Dt} = \vec{\zeta} \cdot \nabla \vec{\mathbf{V}}$$

But for a potential flow field the flow velocity $\vec{\mathbf{V}}$ can be expressed as $\nabla\Phi$, where Φ is the total velocity potential, such that the vorticity field $\vec{\zeta}(= \nabla \times \vec{\mathbf{V}})$ is identically zero in the whole flow field. Therefore, the vorticity transport equation does not need to be coupled into the solution technique for the potential flow problem.

Boundary conditions must be added to the above equations to complete the problem statement. Attention will be restricted here to the so-called direct problem [43] of fluid dynamics. That is specifically where the locations of all boundary surfaces are known, possibly as functions of time, and the normal component of fluid velocity is prescribed on these boundaries. In general there may be several bodies moving with respect to each other. But the entire boundary surface will be denoted by S, and the boundary condition will be written as

$$\vec{\mathbf{V}} \cdot \vec{\mathbf{n}} = F(\vec{\mathbf{x}}, t) \tag{2.3}$$

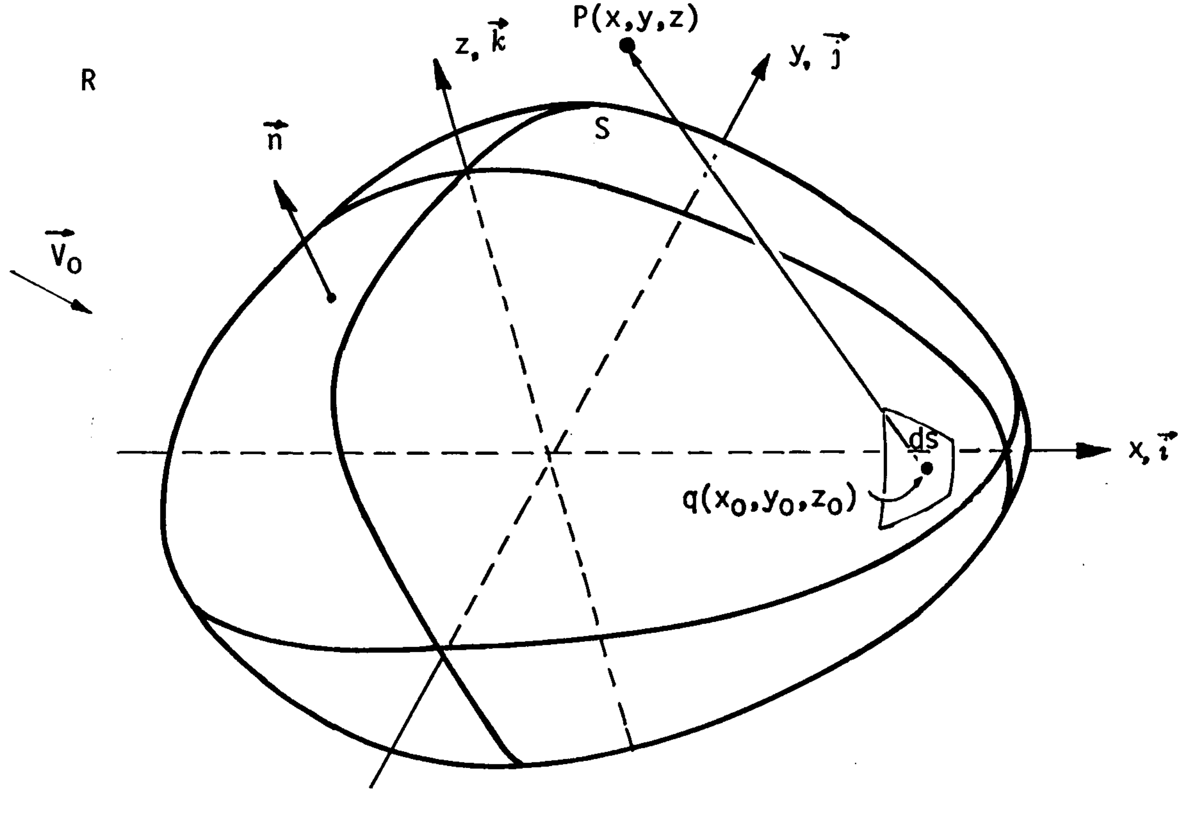


Figure 2.1. Nomenclature for a three-dimensional body in a flow field [43].

where $\vec{\mathbf{n}}$ is the unit outward normal vector at a point on S, and F is a known function of the position vector $\vec{\mathbf{x}}$ on S and possibly also a known function of time. For the external flow problem a regularity condition at infinity must also be imposed (that is, the perturbation velocity due to the body should die out at infinity).

A solution for the fluid velocity field $\vec{\mathbf{V}}$ at any point in the region R can be obtained by expressing it as the sum of two velocities:

$$\vec{\mathbf{V}} = \vec{\mathbf{V}}_o + \vec{\mathbf{u}} \tag{2.4}$$

The velocity $\vec{\mathbf{V}}_o$ is the undisturbed flow field without the presence of the body and is called the "onset velocity". The $\vec{\mathbf{u}}$ is the perturbed velocity field due to the presence of the body. The velocity $\vec{\mathbf{V}}$ must satisfy the incompressible flow condition $\nabla \cdot \vec{\mathbf{V}}$. This is satisfied if both $\vec{\mathbf{V}}_o$ and $\vec{\mathbf{u}}$ are divergence free fields. It is further assumed that $\vec{\mathbf{u}}$ is irrotational, that is: $\nabla \times \vec{\mathbf{u}} = 0$. Thus $\vec{\mathbf{u}}$ may be expressed as the gradient of a perturbed potential function ϕ,

$$\vec{\mathbf{u}} = \nabla\phi \tag{2.5}$$

The condition of zero divergence then yields the Laplace equation for ϕ,

$$\nabla^2\phi = 0 \tag{2.6}$$

The boundary conditions on S are derived from the requirement that on a stationary impermeable surface S, the normal component of fluid velocity must vanish, that is $F(\vec{\mathbf{x}}, t) = 0$. Thus, Equation (2.3) becomes

$$\nabla\phi \cdot \vec{\mathbf{n}} = \frac{\partial\phi}{\partial n} = -\vec{\mathbf{V}}_o \cdot \vec{\mathbf{n}} \tag{2.7}$$

Since the right-hand-side is known, Equation (2.7) expresses a Neumann boundary condition for ϕ. If the boundary S is moving or if a nonzero normal velocity (such as a porous wall condition) is prescribed, the right-hand-side of Equation (2.7) can be easily modified.

A regularity condition at infinity is also necessary. In the usual external flow problem this condition is

$$| \nabla\phi | \longrightarrow 0; \quad \vec{\mathbf{x}} \longrightarrow \infty \tag{2.8}$$

For a simply connected region R, Equations (2.6) - (2.8) comprise a well-posed problem for the perturbed potential ϕ.

The essential simplicity of potential flow derives from the fact that the velocity field is determined by the continuity Equation (2.2) and the condition of irrotationality Equation (2.5). Thus the Euler equation Equation (2.1), is not used directly, and the velocity may be determined independently of the pressure. For a steady state problem the velocity is obtained from the boundary condition Equation (2.7), and once the velocity is known, the pressure is calculated from Equation (2.1). Equation (2.1) can be integrated to give the Bernoulli equation as

$$\frac{| \vec{\mathbf{V}} |^2}{2} + \frac{\mathbf{p}}{\rho} - U = const. \tag{2.9}$$

and the pressure coefficient C_p can be expressed as

$$C_p = \frac{\mathbf{p} - \mathbf{p_o}}{\frac{1}{2}\rho| \vec{\mathbf{V}}_o |^2} = 1 - \frac{| \vec{\mathbf{V}} |^2}{| \vec{\mathbf{V}}_o |^2} \tag{2.10}$$

where $\mathbf{p}_o$ is the far upstream pressure. The general approach to solving this well-posed problem is to apply Green's theorem of distributing surface singularities (such as sources, doublets or vortices) on the body surface, and use the panel method to discretize the resulting integral. Figure 2.2 schematically illustrates the computational approach. More details are given in the next section.

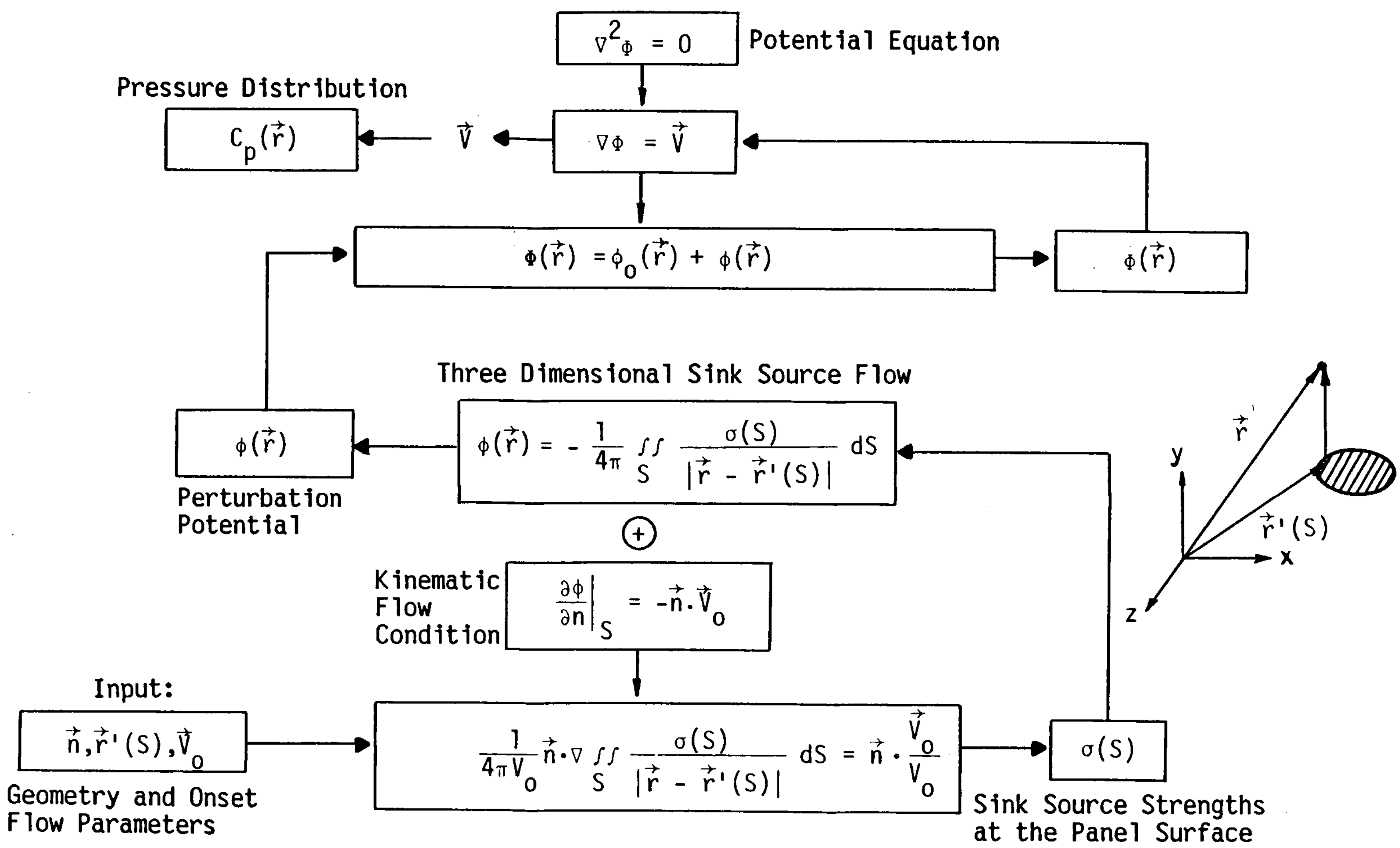


Figure 2.2. The general approach for solving three-dimensional potential flow problems by the source panel method.

B. METHOD OF SOLUTION

To solve the system of equations described above consider the three-dimensional domain R in Figure 2.3. An enclosed region is considered initially but this will be converted to the external flow region of interest later. Define two arbitrary scalar fields Φ_1 and Φ_2 which are second-order differentiable throughout R. According to Green's second theorem, the following equality holds

$$\int\int_R\int(\Phi_1\nabla^2\Phi_2 - \Phi_2\nabla^2\Phi_1)\,dv = \int_S\int(\Phi_2\nabla\Phi_1 - \Phi_1\nabla\Phi_2)\cdot\vec{\mathbf{n}}\,ds, \qquad (2.11)$$

where S is a piecewise smooth boundary of R and $\vec{\mathbf{n}}$ is defined as positive when pointing into the domain R. Because of the definition of the direction of $\vec{\mathbf{n}}$, the right-hand-side will have a different sign from the usual Green's theorem, where $\vec{\mathbf{n}}$ is normally taken as positive outward.

Applying Greens theorem to the potential flow problem, the function Φ_1 will be replaced by the total velocity potential Φ, which satisfies the Laplace equation, and the function Φ_2 is replaced by the elementary solution of the Laplace equation, $1\ /\ r(p,q)$, where $r(p,q)$ is defined as the length or absolute value of the vector $\vec{\mathbf{r}}(p,q)$. The vector $\vec{\mathbf{r}}(p,q)$ is from point q, a dummy variable, to point p, any point in the domain. The derivatives in the vector operator ∇ are taken with respect to the coordinates of the point q.

Following the formulation given by Hunt [44], the expression for the total velocity potential Φ_p is

$$\begin{aligned}4\pi T\Phi_p = 4\pi\Phi_o &+ \int_{S_{ab}}\int(\frac{-1}{r})\ \vec{\mathbf{n}}_a\cdot(\nabla\Phi_a - \nabla\Phi_b)\ ds \\ &+ \int_{S_{ab}}\int(\Phi_a - \Phi_b)\ \vec{\mathbf{n}}_a\cdot\nabla(\frac{1}{r})\ ds \qquad (2.12)\end{aligned}$$

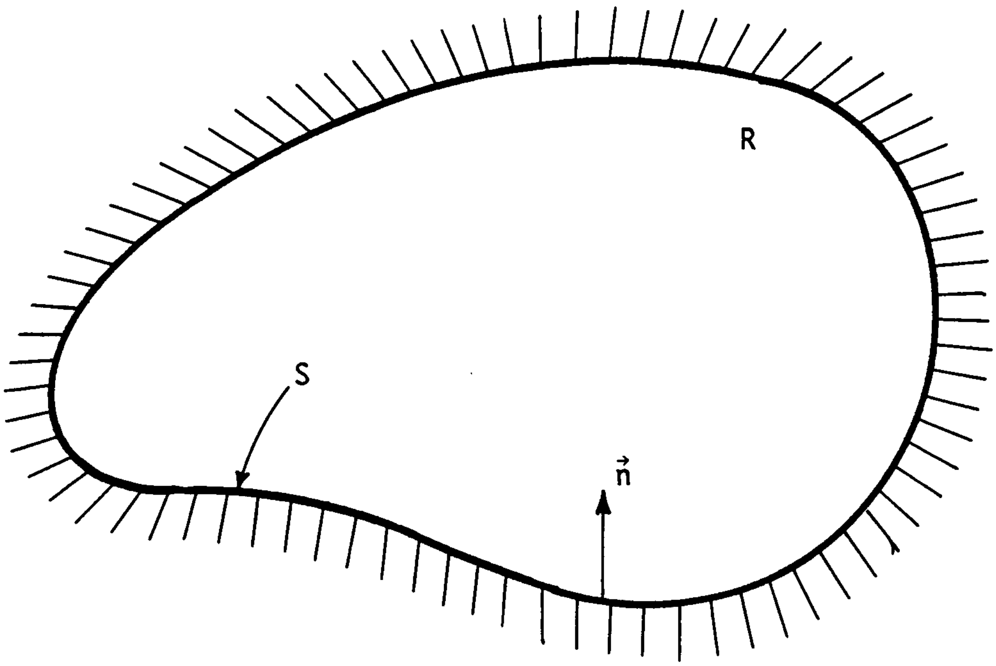


Figure 2.3. The three-dimensional mathematical domain.

where Φ_o is the unperturbed velocity potential. Φ_a and Φ_b are the velocity potentials inside and outside of the body surface S_{ab}, respectively, and $\vec{\mathbf{n}}_a$ is the outward normal. In the following paragraphs the two integrals on the right-hand-side of Equation (2.12) will be identified as source and doublet distributions, respectively.

For a source distribution of variable density σ per unit area on an arbitrary surface S (see Figure 2.4 (a)), the perturbed potential and velocity associated with σ can be expressed [51] as:

$$4\pi\phi_{s_p} = \int_S\int \frac{-\sigma(q)}{r(p,q)}\, ds \tag{2.13}$$

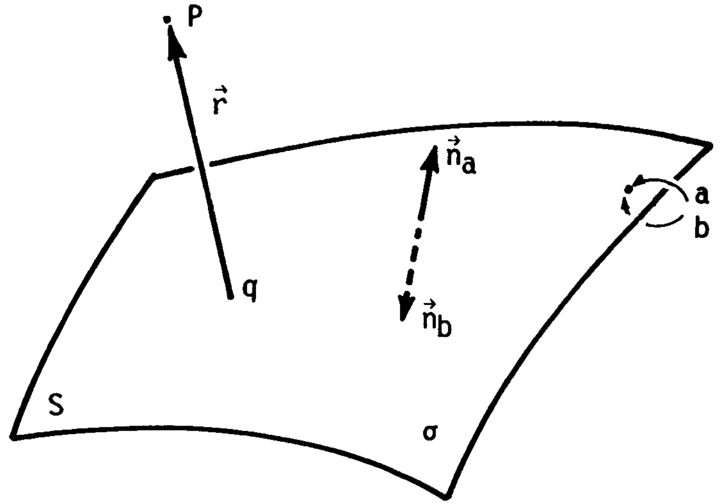
$$4\pi\vec{\mathbf{V}}_{s_p} = \int_S\int -\sigma(q)\ \nabla_p(\frac{1}{r(p,q)})\, ds \tag{2.14}$$

In a limiting analysis as $q \rightarrow p$, the contribution to $4\pi\vec{\mathbf{V}}_{s_p}$ due to the infinitesimal area of S is equal to $2\pi\sigma_p\vec{\mathbf{n}}_a$ on the side a of S and equal to $2\pi\sigma_p\vec{\mathbf{n}}_a(\equiv -2\pi\sigma_p\vec{\mathbf{n}}_a)$ on the side b of S. The normal velocity jump $\vec{\mathbf{n}}_a\cdot(\vec{\mathbf{V}}_a - \vec{\mathbf{V}}_b)$ is equal to σ. Therefore, the first integral on the right-hand-side of Equation (2.12) can be identified as a source distribution.

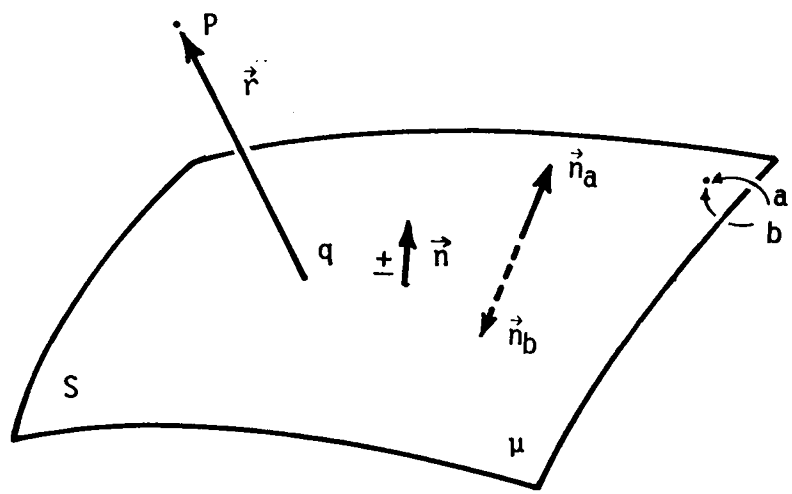
A surface doublet distribution of variable density μ per unit area on an arbitrary surface S (see Figure 2.4 (b)) produces at an external point p a perturbed potential ϕ_{d_p} as follows:

$$4\pi\phi_{d_p} = -\int_S\int \mu\ \vec{\mathbf{n}}\cdot\nabla(\frac{1}{r(p,q)})\, ds, \tag{2.15}$$

where the normal $\vec{\mathbf{n}}$ and the derivatives in ∇ are taken with respect to the coordinates of the variable point q lying on the surface S. The local unit normal $\vec{\mathbf{n}}$ is the doublet axis drawn from the negative to the positive end of the doublet. A similar limiting analysis as noted before is required as $q \rightarrow p$ on S. The contribution to $4\pi\phi_{d_p}$ due to the side a of S is $-2\pi\mu\vec{\mathbf{n}}_a\cdot\vec{\mathbf{n}}$ and equal to $-2\pi\mu\vec{\mathbf{n}}_b\cdot\vec{\mathbf{n}}(\equiv 2\pi\mu\vec{\mathbf{n}}_a\cdot\vec{\mathbf{n}})$ on side b. The potential jump $(\Phi_a - \Phi_b)$ is thus equal to $-\mu\vec{\mathbf{n}}_a\cdot\vec{\mathbf{n}}$. The normal



(a) Source sheet with strength σ.



(b) Doublet sheet with strength μ.

Figure 2.4. Singularity distributions.

velocity is continuous across S while the tangential velocity jumps by an amount equal to the tangential derivative of the jump in potential. Hence, the second integral on the right-hand-side of Equation (2.12) can be identified as the doublet distribution.

However, while the vortex and doublet sheets are computationally equivalent, the former has more appeal, from a physical point of view, in the present application. The theorem relating doublet and vortex sheets has been established by Hess [45]. The theorem is stated as follows:

> **Theorem:** Let the surface S be covered with a variable doublet distribution of intensity μ (the doublet axes are along $\vec{\mathbf{n}}$). The velocity at (x, y, z) due to the doublet sheet is equal to the sum of the velocities due to a certain vortex sheet of strength $\vec{\gamma}$ on S and due to a vortex filament of strength Γ along C, where C is the curve surrounding the surface S. The strength of the vortex filament is the local (edge) value of the doublet strength, i.e. $\Gamma = \mu$ on C. The vorticity on the sheet is a vector everywhere tangent to the curves of constant μ and has an intensity equal to the magnitude of $\nabla\mu$. More specifically, if $\vec{\gamma}$ is the vector vortex strength on S, then $\vec{\gamma} = -\vec{\mathbf{n}} \times \nabla_q \mu$. See Appendix A for proof.

A general formulation of how to distribute and identify the surface singularities is derived in the above paragraphs. The procedures for solving potential flow problems which involves these surface singularities are identical to those described in the previous section. Let sources and vortices be distributed on a body surface such as to satisfy the normal boundary condition Equation (2.7). Thus

$$\begin{pmatrix} Normal \\ B.C. \end{pmatrix} = \int_S\int \sigma(q) \begin{pmatrix} source \\ kernel \end{pmatrix} ds + \int_S\int \gamma(q) \begin{pmatrix} vortex \\ kernel \end{pmatrix} ds \qquad (2.16)$$

where Equation (2.16) is expressed in symbolic form. The source or vortex kernel functions are directly related to the geometry of the body, which is independent of the normal boundary condition. If the integration over surface S is divided into discrete "panels" and the sources and vortices are distributed piecewise constant on the panels, Equation (2.16) can be expressed in a discrete form as

$$\begin{pmatrix} Normal \\ B.C. \end{pmatrix} = \sum_j \left[\sigma_j \int_j\int \begin{pmatrix} source \\ kernel \end{pmatrix} ds + \gamma_j \int_j\int \begin{pmatrix} vortex \\ kernel \end{pmatrix} ds \right] \qquad (2.17)$$

where summation extends over all panels with j standing for the j th panel. Equation (2.17) can be split into two equations and the resulting system of equations can be solved for the strengths of σ_j and γ_j by matrix inversion or some other technique. Once the σ_j and γ_j are obtained, the velocity and pressure distribution can be solved. An illustration of the numerical solution procedure for the (source) surface singularity distribution is shown in Figure 2.5.

Different choices for the singularity distributions and panel geometry in Equation (2.17) are possible. Generally speaking, the simpler form of singularity distributions and panel geometry, such as constant or piecewise constant singularity distributions and the use of flat panels, are named a "lower-order" formulation, in contrast to the so-called "higher-order" formulation which uses linear or parabolic singularity distributions and/or curved panels. More details on panel order is given in Section D. Before describing these details, however, the important Kutta condition required to render a unique solution for the lift on the airfoil is discussed.

C. KUTTA CONDITION

In two-dimensional external flow problems, the region exterior to the object is not simply-connected, and Equations (2.5) - (2.8) do not define a unique velocity

$$\frac{1}{4\pi V_o}\vec{n}\cdot\nabla\iint_S \frac{\sigma(S)}{|\vec{r} - \vec{r}\,'(S)|}\,dS = \vec{n}\cdot\frac{\vec{V}_o}{V_o}$$

$$\left[-\vec{n}_i\cdot\nabla\iint_{\substack{\text{Panel}\\ j}}\frac{1}{|\vec{r}_i - \vec{r}_j\,'(\hat{S})|}\,d\hat{S}\right] \qquad \left\{\frac{\sigma_j}{4\pi V_o}\right\} = \left\{-\vec{n}_i\cdot\frac{\vec{V}_o}{V_o}\right\}$$

Influence Coefficients Matrix

$$[-\vec{n}_i\cdot\vec{C}_{ij}]$$

Linear Equation System

$$[I_{ij}] \qquad \left\{X_j\right\} = \left\{R_i\right\}$$

Matrix Elements — Solution Vector — Right Hand Side

$$[\vec{C}_{ij}]\left\{X_j\right\} + \left\{\frac{\vec{V}_o}{V_o}\right\}_i = \vec{V}_i$$

Figure 2.5. Illustration of numerical solution procedure for the (source) surface singularity distribution.

field. To see this let the total circulation Γ around any closed path C in the field be defined as the line integral:

$$\begin{aligned}\Gamma &= \int_C \vec{\mathbf{V}} \cdot d\,\vec{\mathbf{l}} \\ &= \int_C \vec{\mathbf{V}}_o \cdot d\,\vec{\mathbf{l}} + \int_C \vec{\mathbf{u}} \cdot d\,\vec{\mathbf{l}} \\ &= \Gamma_o + \Gamma',\end{aligned} \tag{2.18}$$

where Γ' is the circulation associated with the perturbed velocity due to the body. If C does not enclose all or part of S, then $\Gamma' = 0$. If S is a single surface, it can be shown [43] that the velocity field $\vec{\mathbf{u}}$ is rendered unique by specifying Γ' for any C that encloses S.

For a two-dimensional airfoil in a uniform inviscid stream, the drag force is zero, and the lift is proportional to the perturbed circulation Γ' (because for uniform undisturbed flow the total circulation Γ is equal to Γ'). However, for a doubly-connected domain, Equation (2.18) is not equal to zero, and can have any value. Thus, in two-dimensional flow problems the lift can also be any value. Some auxiliary condition is needed to fix the indeterminancy of lift (or Γ'). For a two-dimensional airfoil, the well-known"Kutta condition" can be applied to insure that the flow leaves the trailing edge smoothly and has a finite value such that the value of lift is unique.

Unlike the two dimensional case, the region exterior to a closed three dimensional body is simply connected, so that if the flow is potential, i.e., has zero *curl*, and is free from singularities, then Equation (2.18) will equal zero. Therefore, with inviscid potential flow the resulting net force on a closed three-dimensional body is zero. This is because of the fact that all components of force on a body, both the lift and drag, are ultimately due to viscosity. Hence the formulation of the potential flow model for the three dimensional lifting body problems must impose a Kutta condition as in the two dimensional case if a nonzero lift value is to be computed.

In order to obtain nonzero values of section circulation, there must be some form of singularity in the external flow. The nature of the singularity can be exhibited by considering the path C shown in Figure 2.6. The line integral of velocity around this path is

$$\int_C \vec{\mathbf{u}} \cdot d\,\vec{\mathbf{l}} = \Gamma_1' - \Gamma_2' + \int_I (\vec{\mathbf{u}}_{upper} - \vec{\mathbf{u}}_{lower}) \cdot d\,\vec{\mathbf{l}} \tag{2.19}$$

For the definition of the subscripts 1 and 2 see Figure 2.6. Let I represent the intersection of a sheet of vorticity with the body surface. To complete the potential flow model, a stream surface (i.e. wake) leaving the body is selected based on physical intuition. It is reasoned that vorticity is introduced only to the fluid that passes by the body and that the path I of Equation (2.19) must lie along the trailing edge of the wing [45]. Therefore, a vortex sheet issues from the trailing edge and for steady flow proceeds to infinity. The average strength of the sheet along I is proportional to the difference $\Gamma_1' - \Gamma_2'$. Taking the limit as the two section curves approach each other gives the result that the local strength of the trailing vortex sheet is proportional to the spanwise derivative of the section circulation.

The physical Kutta condition requires the pressure (or velocity) on the upper and lower surfaces to have a common limit at the trailing edge. There are two alternative ways of imposing the Kutta condition: one is at the two control points on the two adjacent and rearmost panels, and the other is on a certain number of control points which are placed on a sheet extending downstream from the trailing edge. However determining an appropriate location for this sheet and the downstream position of the control points is critical and requires considerable experience [45]; therefore it is not used in this study.

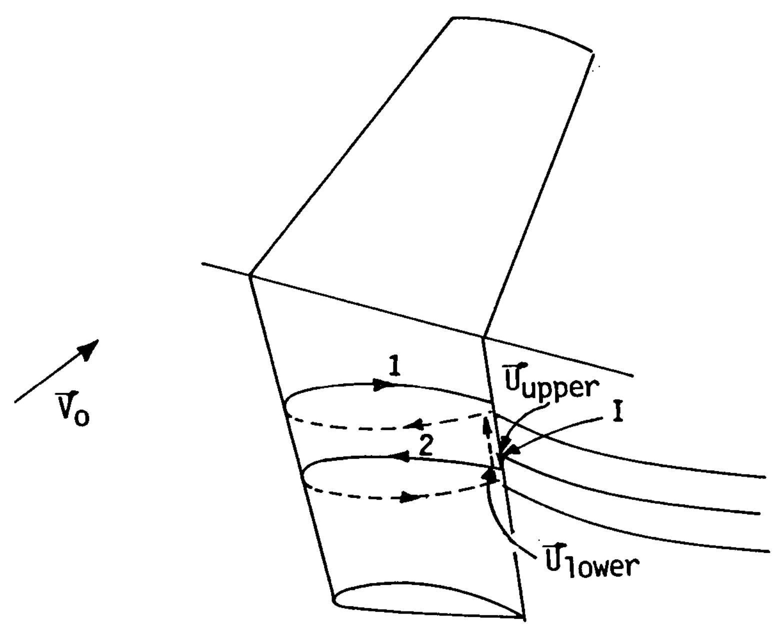


Figure 2.6. Circulation on a three-dimensional wing.

Some properties that may be deduced from the Kutta condition are as follows [45]:

Two-dimensions:

(1) A stream line of the flow leaves the trailing edge with a direction along the bisector of the trailing edge angle.

(2) As the trailing edge is approached, the surface pressures (velocity magnitudes) on the upper and lower surfaces have a common limit, which equals the stagnation pressure if the trailing edge angle is nonzero.

(3) The source density of variable strength is zero at the trailing edge.

Three-dimensions:

(1) A stream surface of the flow leaves the trailing edge with a direction that is known, or at least can be approximated (see below).

(2) As each point on the trailing edge is approached, the surface pressures (velocity magnitudes) on the upper and lower surfaces have a common limit.

(3) The source density of variable strength is zero at each point on the trailing edge.

This section has pointed out that a strong relationship between the Kutta condition and the trailing edge exists. For airfoils with cusped trailing edges (such as the Joukowski airfoil) the panel method has difficulty because the two rearmost panels are almost identical causing the matrix inversion to become ill-conditioned. The same is true if the trailing edge angle is very small. Raising the order of the panel method to overcome this difficulty has been proposed for years. Recently, it has been found, however, that the lower-order formulation is able to handle these airfoil types if the problem is properly specified. This will be discussed in the next section and some new observations will be presented.

D. ORDER OF PANEL METHOD

The surface singularity method provides a concise and accurate way to compute potential flow field problems and is an extension of the classical slender body theory. Hess and Smith [43] suggested that the boundary condition of zero normal velocity on the surface of the body (rather than on the chord line as is done in slender body theory) be used to solve for the source and vortex singularity strengths. They adopted a piecewise constant source strength and a constant surface vorticity distribution on flat surface elements. However this formulation is not capable of handling airfoils with very thin and/or highly loaded trailing edges. This is manifest by a calculated surface pressure distribution exhibiting an inappropriate crossing of the upper surface and lower surface pressure distributions close to the trailing edge, thereby indicating a fictitious negative loading (see Figure 2.7). The correct solution of the C_p distribution of the Joukowski airfoil can be found from the classical aerodynamics textbook.

In order to alleviate this unacceptable phenomenon, Hess [46] proposed a higher-order method which led to a new fashion in panel methods. The higher order formulation uses parabolic surface elements and piecewise parabolic source distributions. However, the higher order methods still do not eliminate the unrealistic calculation of the pressure distribution. With the higher-order methods (but with a constant surface vorticity distribution), the crossing at the trailing edge is replaced by a "hump" (illustrated in Figure 2.7 from Hess [46]) at the trailing edge. Therefore, Hess proposed implementing the calculation by a vorticity distributor whose strength approaches zero at the trailing edge on both the upper and lower surfaces. He believed that this vorticity distributor would stop the source distribution from simulating a doublet sheet at the trailing edge (since

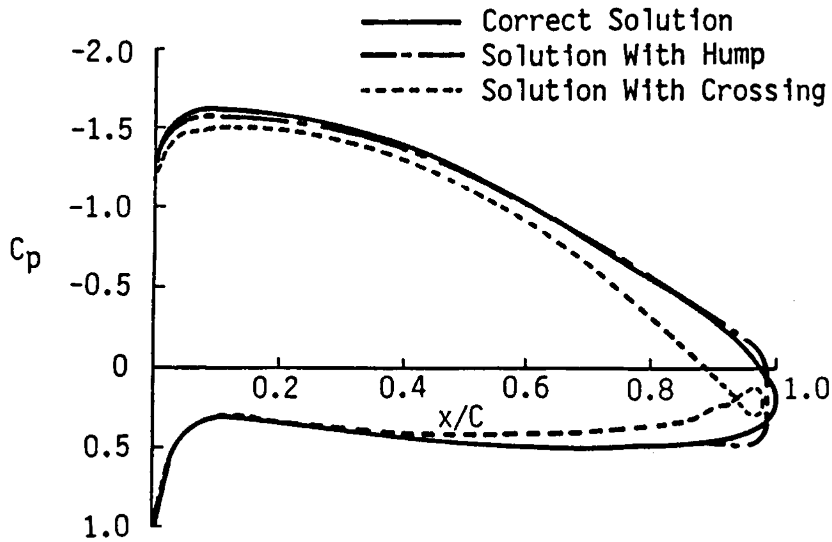


Figure 2.7. Illustration of the pressure distributions with inappropriate crossing or hump at the trailing edge [46].

this phenomenon was thought to have caused the singularity problem). According to his numerical experiments, a vorticity distribution which was parabolic over the entire airfoil surface gave smooth results.

Recently Maskew [47] suggested a lower-order panel method for solving subsonic aerodynamic flow characteristics. He felt that although the higher-order panel methods reduced the number of panels they did not result in any benefits in computing costs. Moreover care and experience are needed in the choice of the interpolation function for the piecewise higher-order representation. Hence Maskew proposed a piecewise constant singularity panel method for internal Dirichlet boundary value problems. Comparable accuracy to solutions given by higher-order panel methods can be obtained by his formulation using the same number of control points. However, asymmetrical panelling, i.e. unequal number of panels on the upper and lower surfaces, gave a poor pressure distribution near the trailing edge. This was clearly observed in Maskew's results. Even for a conventional airfoil such as the NACA 0012 the results were poor with asymmetrical panelling as shown in Figure 2.8.

Ardonceau [48] adopted the well-known idea of transforming an airfoil with small trailing edge angle into a near circle via the Joukowski transformation. Panel methods work well in the transformed plane. Groh [49] pointed to the numerical condition of the problem by noting (in a formal sense) that $\mathbf{AX} = \mathbf{B}$ and $(\mathbf{A} + \mathbf{A}')\mathbf{X} = \mathbf{B} + \mathbf{A}'\mathbf{X}$ ($\mathbf{B}, \mathbf{X} \in \mathbf{V}_n$; while $\mathbf{A}, \mathbf{A}' : \mathbf{V}_n \rightarrow \mathbf{V}_n$) are algebraically identical equations but can have different numerical conditioning; that is the smallest $\mid \lambda \mid$ where λ is an eigenvalue for $\mathbf{A} + \mathbf{A}'$ can be made larger than the corresponding one for $\mathbf{A}$ alone.

Since no common viewpoint on the preferred order of the panel method has emerged from previous research, this issue was further pursued by the present

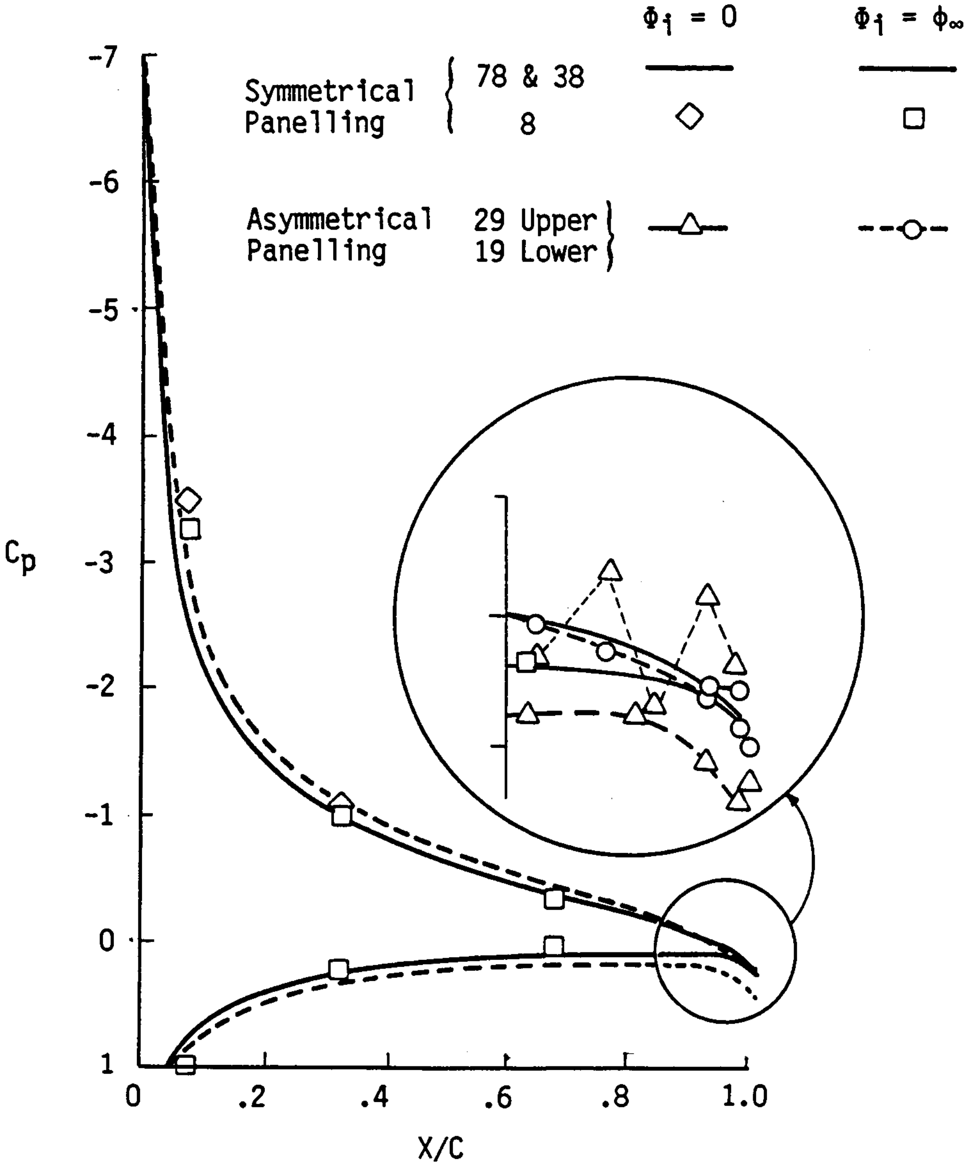


Figure 2.8. Calculated pressure distribution on NACA 0012 at 10 deg. angle of attack by Maskew [47].

author [50]. Reference [50] notes the difference between airfoils with nonzero lift and with zero lift is solely due to the vortex distribution. The source distribution makes no contribution to lift. Based on this idea, the onset velocity was split into two parts: one aligned along the zero-lift line and the other perpendicular to it (see Figure 2.9). The zero lift line velocity component is used to determine the source distribution and the perpendicular component in determining the vortex distribution. The reason for doing this is based on the following observation. Physically, since the component of the onset flow along the zero-lift line produces no lift, this component should be used to determine the source distribution. The other component of the onset flow, which is perpendicular to the zero-lift line, will then produce all the lift. This concept is in keeping with Hess' speculation that for airfoils with a cusped trailing edge, the source distribution must be inhibited from simulating a doublet at the trailing edge. Therefore, the proper vorticity distribution must be determined to completely remedy this problem. The artificial parabolic vorticity distribution suggested by Hess however is not appropriate. But by decomposing the flow into a component tangential to and one perpendicular to the zero lift line, a vortex distribution which will eliminate the simulated doublet distribution can be computed. The following paragraphs described the method of solution proposed in this study and demonstrate the improved form of solution. The method begins by distributing piecewise constant surface singularities (source and vortex) on flat panels over an airfoil with a cusped trailing edge. The computational scheme is different from Maskew [47] in that the external Neumann boundary value problem is solved as contrasted to the Dirichlet boundary value problem. For the external Neumann boundary value problem, the boundary condition of zero normal velocity on the surface of the body is used to derive the integral equations for the distribution of surface singularities. Applying

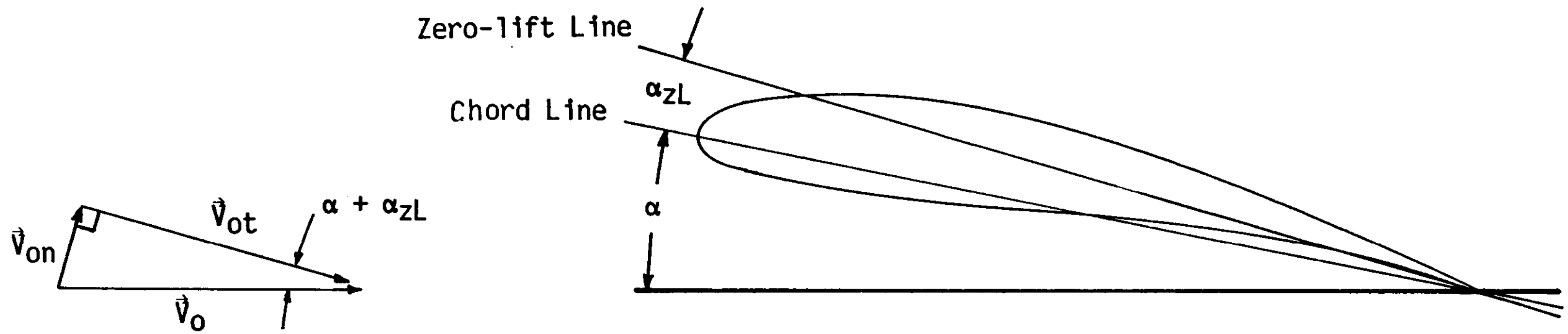


Figure 2.9. Geometry of airfoil and decomposition of onset velocity.

piecewise constant source and vortex distributions on each flat panel will result in the following discrete equations:

$$\mathbf{I}^n_{ij}\, \frac{\sigma_j}{2\pi} = -\vec{\mathbf{V}}_{0_t} \cdot \vec{\mathbf{n}}_i = \mathbf{R}_1 \tag{2.20}$$

$$\mathbf{D}^n_{ij}\, \frac{\gamma_j}{2\pi} = -\vec{\mathbf{V}}_{0_n} \cdot \vec{\mathbf{n}}_i = \mathbf{R}_2 \tag{2.21}$$

The matrices $\mathbf{I}^n$ and $\mathbf{D}^n$ are the influence coefficient matrices in the normal direction for the source and vortex distributions, respectively. The subscripts t and n refer to tangential and normal to the zero lift line, respectively.

From Equations (2.20) and (2.21), it is evident that the column matrices σ and γ are determinable only if the matrices $\mathbf{I}^n$ and $\mathbf{D}^n$ are invertable. The rows of $\mathbf{I}^n$ and $\mathbf{D}^n$ are linearly independent as long as all the surface panels are distinct. As the trailing-edge angle of the airfoil approaches zero, two panels (one from each surface) become almost identical in geometric location. In this case, the inversion process for $\mathbf{I}^n$ and $\mathbf{D}^n$ becomes ill-conditioned. However, appeal to the Kutta condition (applied so that surface pressures are equalized on the last panels of the upper and lower surfaces) causes the rows with the identical panels (that is, repeated rows) in the influence coefficient matrices to be replaced by two distinct rows (more details see Chapter IV).

As suggested in the works of Maskew [47] and Groh [49], the right hand sides of Equations (2.20) and (2.21) can also have significant effects on the accuracy of the overall solution process. Thus, amplifying a comment by Groh, this can be discussed in terms of the eigensystem of the matrices $\mathbf{I}^n$ or $\mathbf{D}^n$. The final discrete form for the velocity vector is

$$\vec{\mathbf{u}} = \mathbf{C}\, \vec{\mathbf{R}}_1 - \mathbf{C}^{-1}\, \vec{\mathbf{R}}_2, \tag{2.22}$$

where $\mathbf{C} = -\mathbf{D}^n\ (\mathbf{I}^n)^{-1}$ (see Appendix C) is the coefficient matrix for the contribution from the source distribution and $\mathbf{C}^{-1}$ the coefficient matrix for the vortex contribution.

Let $\{(\lambda_i, \vec{\mathbf{X}}_i)\}$ be the eigensystem of $\mathbf{C}$. Then if $\vec{\mathbf{R}}_1 = \sum \alpha_i \vec{\mathbf{X}}_i$, $\vec{\mathbf{R}}_2 = \sum \beta_i \vec{\mathbf{X}}_i$ and $\vec{u} = \sum \delta_i \vec{\mathbf{X}}_i$, it directly results that

$$\delta_i = \alpha_i \lambda_i - \beta_i / \lambda_i \tag{2.23}$$

is the expression for the coefficients in the expansion of the velocity vector $\vec{u}$. Since the numerical values of λ_i are widely spaced, poor results are obtained from Equation (2.23) because the importance of small changes to α_i or β_i may magnify their effects on δ_i. The coefficients α_i and β_i are dependent upon the form of the decomposition, $\vec{\mathbf{R}}_1$ and $\vec{\mathbf{R}}_2$, used to determine the source and vortex singularity strengths.

Recall that Hess [46] suggested a parabolic vortex distribution around the surface of the airfoil be used. From classical thin airfoil theory [51], the calculation of uniform flow over a cambered airfoil at incidence can be decomposed into a uniform flow without angle of attack over a symmetrical airfoil, a uniform flow without angle of attack over a camber line only, and a uniform flow with angle of attack over a flat plate (see Figure 2.10). In the case of a symmetric airfoil, it is only necessary to consider the thickness distribution and the flat plate incidence loading. In thin airfoil theory, the former can be expressed as a source distribution and the latter as a vortex distribution placed on the chord line. The analytical expression for the vortex distribution, the flat plate loading, is given as

$$\gamma(\theta) = 2\ V_o\ \alpha\ \frac{1 - cos(\theta)}{sin(\theta)}, \tag{2.24}$$

where $-\pi \leq \theta \leq \pi$. The argument θ is used to replace the x coordinate in such a way that, $x = (1 + cos(\theta))/2$.

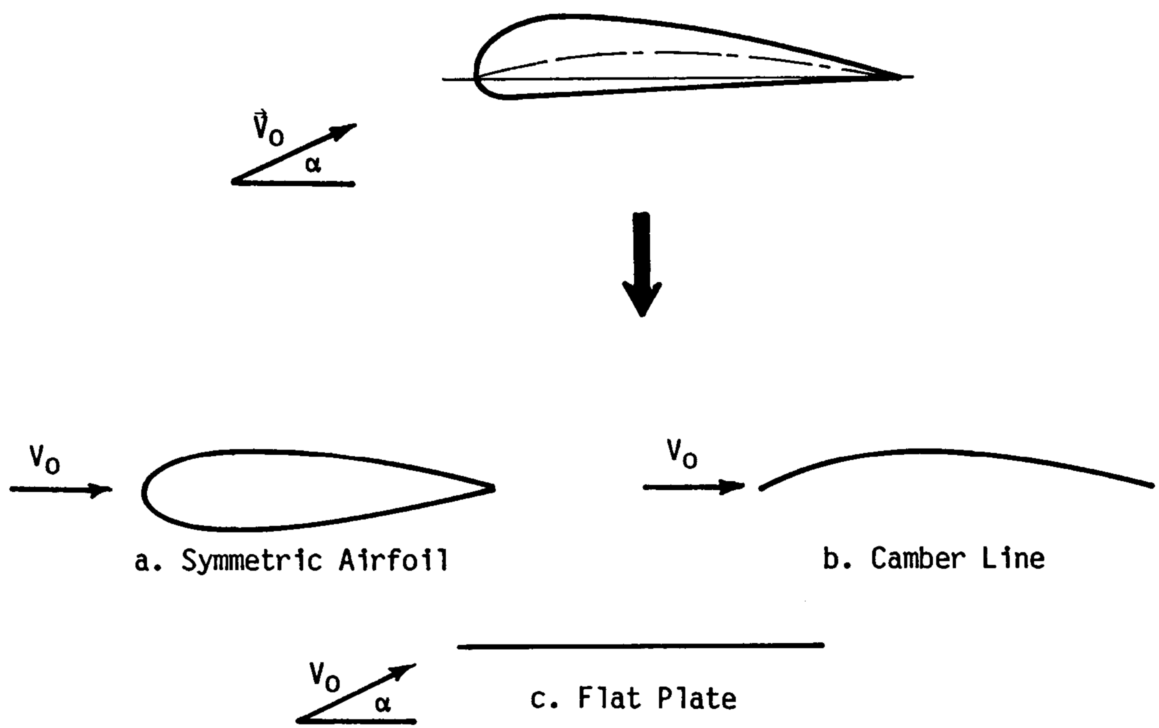


Figure 2.10. Illustration of the classical decomposition of the problem of an arbitrary thin airfoil into three simpler problems.

A computer program is developed for solving Equations (2.20) and (2.21), which is based on the proposed onset velocity split. More details about the computer program will be discussed in Chapter IV. The vortex distribution calculated from Equation (2.21) is plotted along the surface around a NACA 0012 airfoil at 10 degree angle of attack in Figure 2.11. It is evident that this is not the parabolic distribution suggested by Hess [46]. Figure 2.12 shows the total vortex distribution (sum of upper and lower surface distributions from Figure 2.11) compared with the analytical result for the flat plate, Equation (2.24). Both are for 10 degree angle of attack. The difference between them is due to the thickness effect, which is not included in the flat plate calculation; that is, the differences resulting from placing the singularities on the chord line rather than on the surface.

When an unequal number of panels on the lower and upper surfaces, (for example 19 and 29 panels, respectively) is used, inaccuracies can result from the dissimilar panel sizes. Figure 2.8 (page 30) shows one of Maskew's calculations [47] where a poor pressure distribution, especially near the trailing edge, is observed. Maskew uses two different internal velocity potentials for imposing boundary conditions, $\phi_i = 0$ and $\phi_i = \phi_\infty$. He thought that this was due to a severe mismatching of panels at the trailing edges. But, for a conventional airfoil like NACA 0012, there is no large curvature at the trailing edge. This means that the number of panels in that area should not cause the reported problem. On the contrary, because of the larger curvature change at the leading edge, fewer panels may lead to the calculation being inadequate at the leading edge. In order to show that the poor pressure distribution found by Maskew [47] is really due to inappropriate panelling, the same configuration as Maskew's was calculated by the author's method. The results are shown in Figure 2.13 and as expected this phenomenon does not occur

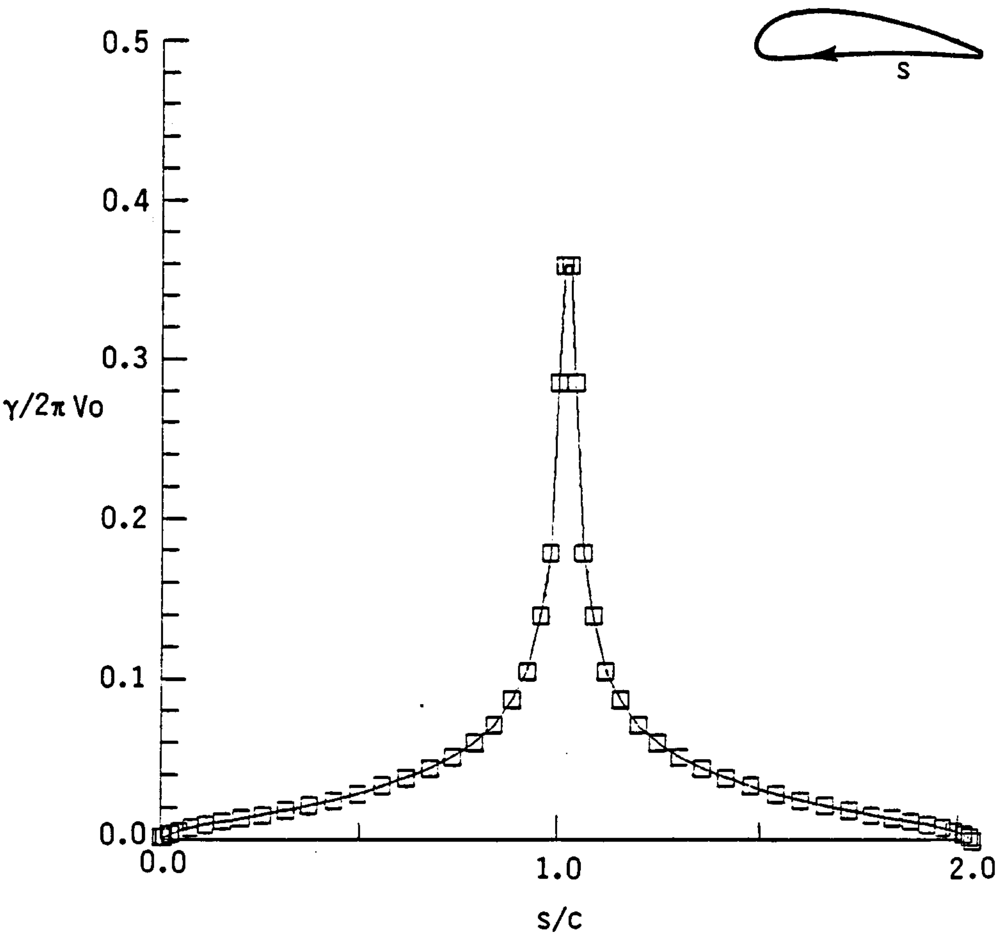


Figure 2.11. Vortex distribution along the surface around NACA 0012 at 10 deg. angle of attack with 50 panels.

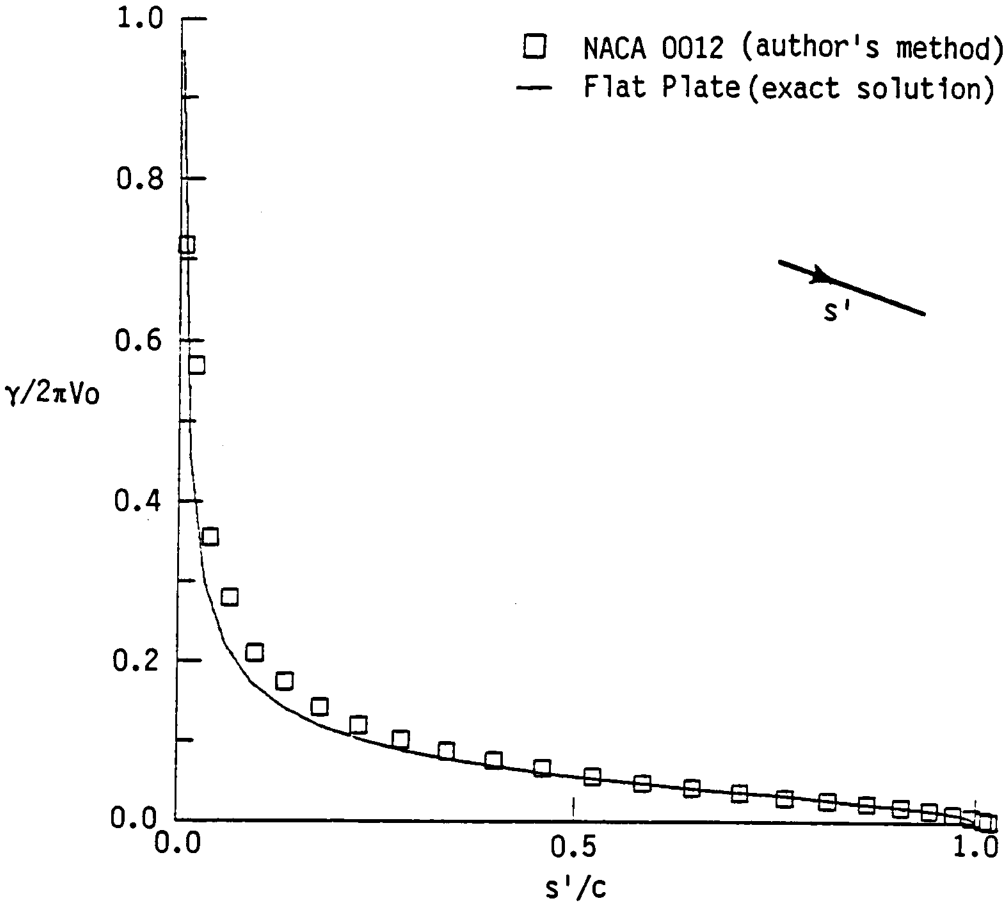


Figure 2.12. Comparison of vorticity distribution between calculated NACA 0012 result (50 panels) and exact flat plate result, both at 10 deg. angle of attack.

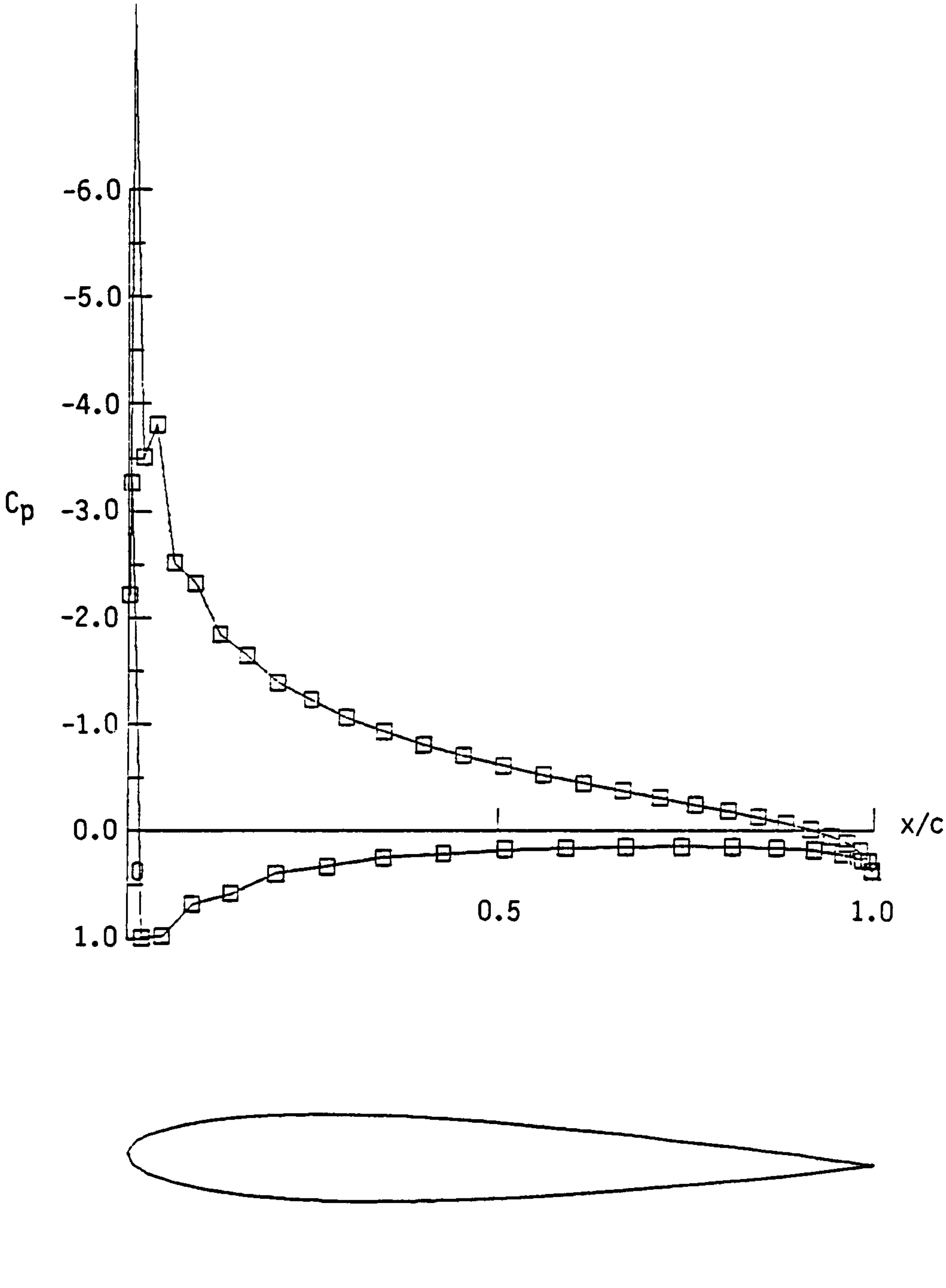


Figure 2.13. Calculated pressure distribution on NACA 0012 at 10 deg. angle of attack with 29(upper)/19(lower) panels.

at the trailing edge. There is, however, some oscillation near the leading edge which, most likely, is due to the inadequate panelling there.

Joukowski airfoils are the best type of airfoil for exploring the author's method as far as its ability to handle airfoils with cusped trailing edges. Figure 2.14 shows an excellent comparison between the analytical solution and the calculated results with various numbers of panels. These calculations are for a symmetric Joukowski airfoil at 10 degrees angle of attack. From Figure 2.14, it is obvious that the number of panels is not a critical factor as long as the number is great enough to describe the airfoil geometry. Figure 2.15 confirms this by comparing results for a range of panel number from 8 to 100 panels. It can be observed that going to more than 30 panels gives no further improvement in C_p distribution. But on the other hand, although the result can be obtained for only 8 panels, the geometry already lose its validity. Generally speaking, 30 panels are able to reveal the proper geometry of conventional airfoils such as NACA 0012. Figure 2.16 shows the calculated pressure distribution for a cambered Joukowski airfoil at 4 degree angle of attack. Again, good agreement with the analytical solution can be obtained.

The conclusions of this section are as follows. The present conceptual split of the onset flow into two components, $\vec{\mathbf{V}}_{0_n}$ normal and $\vec{\mathbf{V}}_{0_t}$ tangential to the zero lift line, is new. The proper vorticity distribution can be determined from $-\vec{\mathbf{V}}_{0_n} \cdot \vec{\mathbf{n}}$ (where $\vec{\mathbf{V}}_{0_n}$ is the component of the onset velocity normal to the zero lift line) directly without imposing an artificial vorticity distributor. This split also supplies a numerically superior form for the Neumann-type lower-order formulation. Higher-order formulations, such as using curved surface elements, piecewise parabolic source distributions, and extrapolated Kutta conditions, can improve a correct solution. However they cannot cure an incorrect solution, because of the

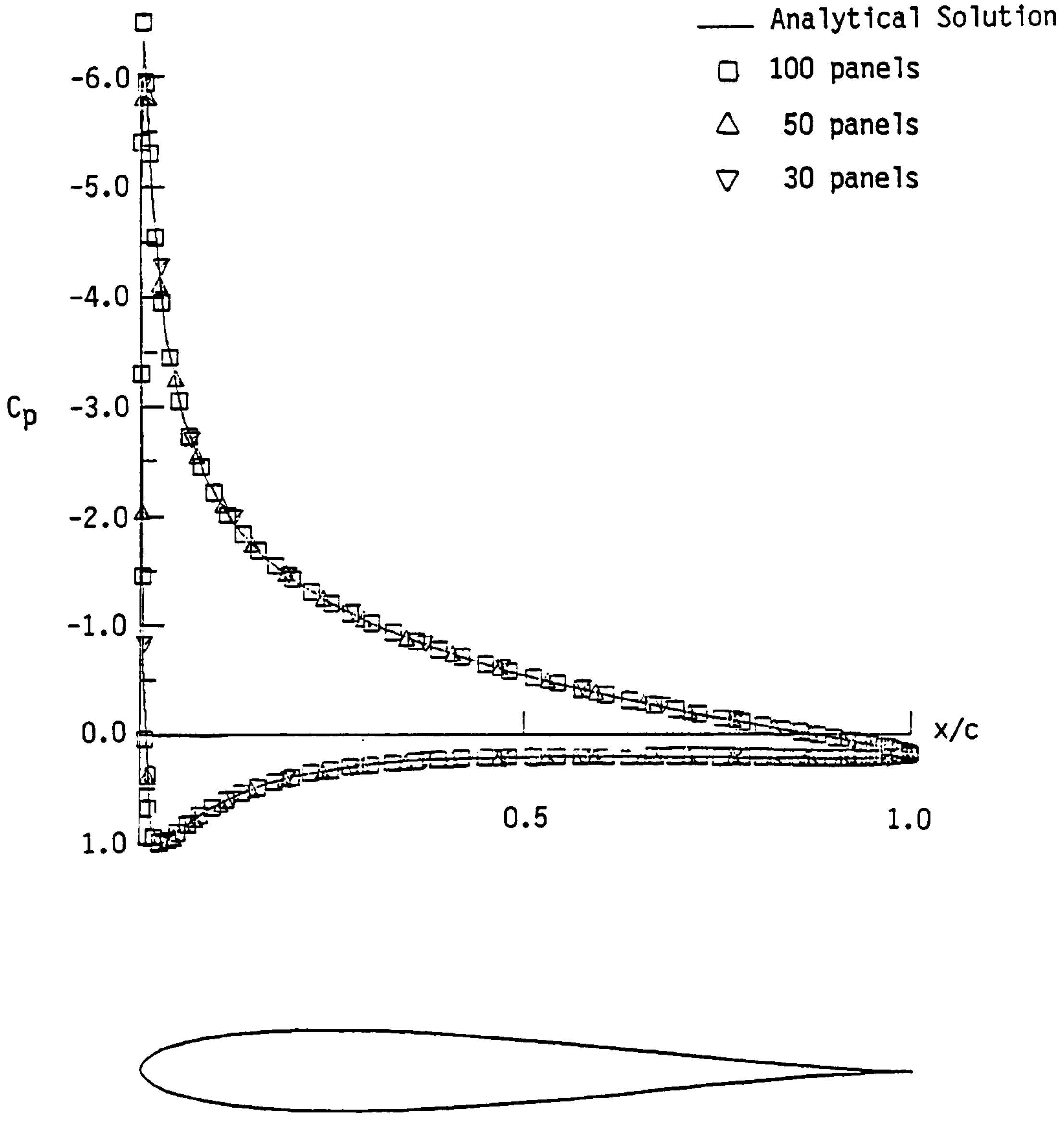


Figure 2.14. Comparison between analytical solution and calculated solution with various panel numbers for a symmetric Joukowski airfoil at 10 deg. angle of attack.

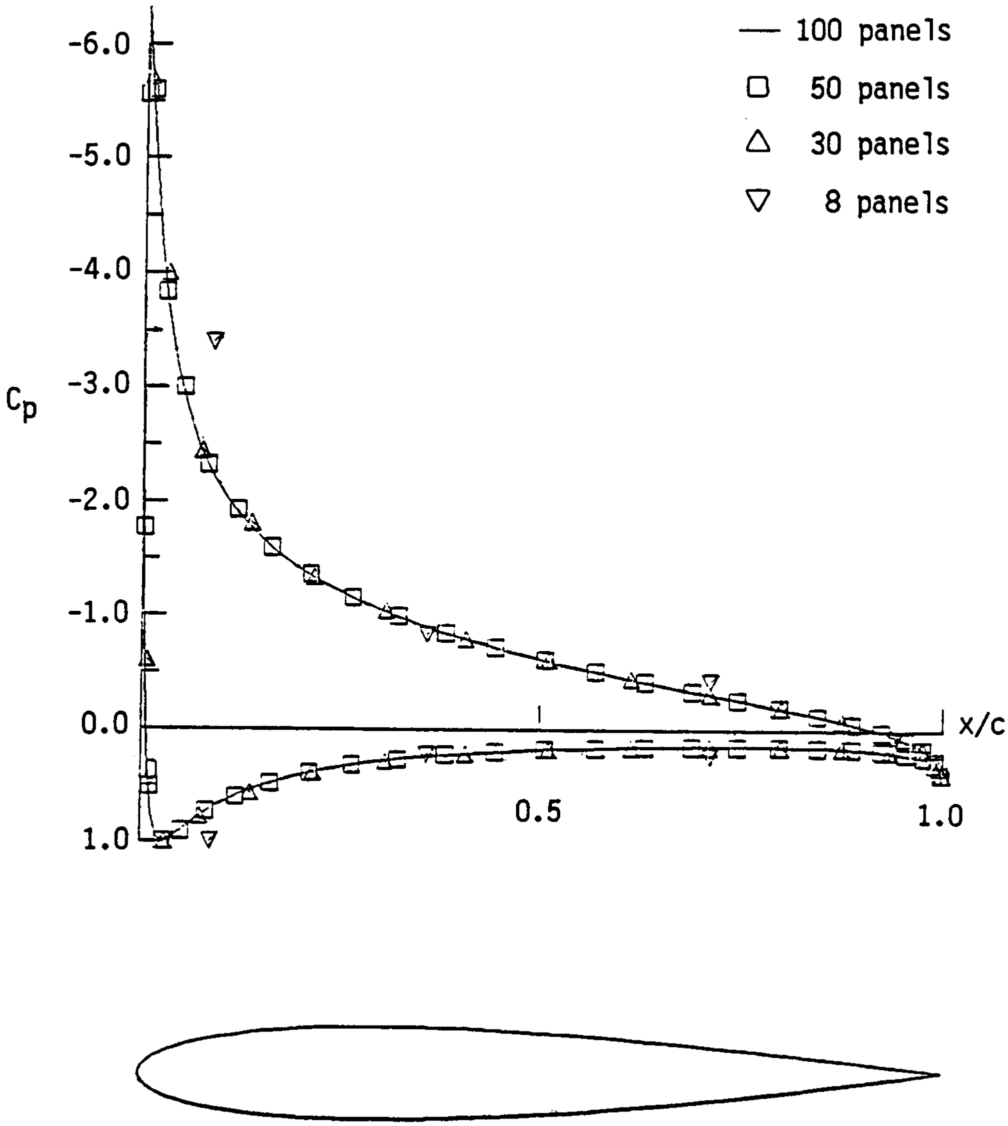


Figure 2.15. Calculated pressure distribution on NACA 0012 at 10 deg. angle of attack with various panel numbers.

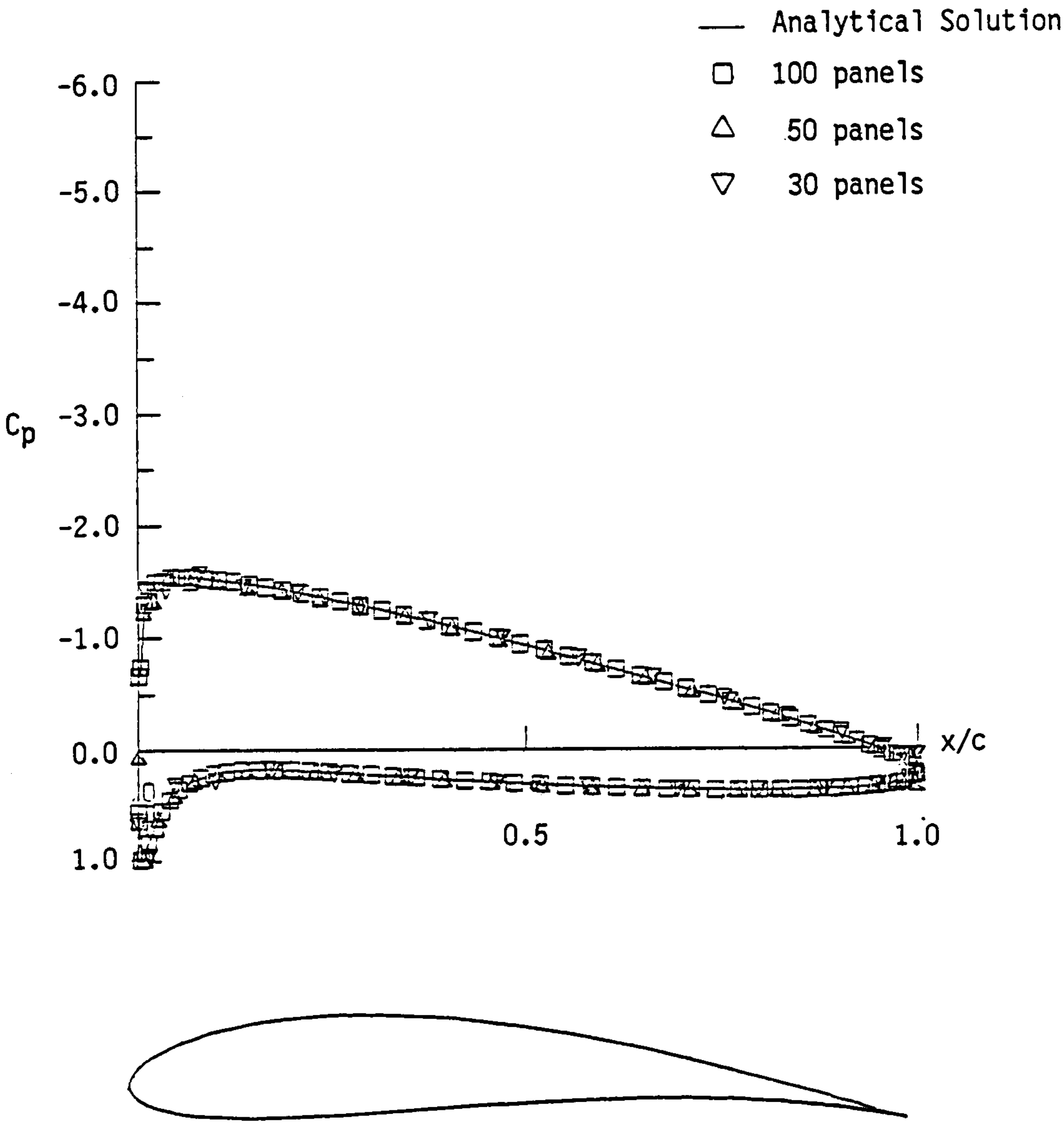


Figure 2.16. Comparison between analytical solution and calculated solution with various panel numbers for a cambered Joukowski airfoil at 4 deg. angle of attack.

induced "hump" on the C_p distribution at the trailing edge. A proper vortex distribution is the key to removing the unacceptable crossing phenomena on the C_p distribution, which is obtained from the author's formulation without specifying an artificial distribution.

CHAPTER III

THE SCALAR/VECTOR POTENTIAL PANEL METHOD

This chapter introduces the scalar/vector potential formulation of rotational flow over a rectangular wing. The panel method is thereby extended from irrotational flow to rotational flow. The airfoil/wing geometry and coordinate transformation will be specified. The types of surface singularities and their relationships with the numerical solution technique will be derived. The detailed formulations for a two dimensional airfoil and a three dimensional rectangular wing will also be presented in this chapter.

A. FORMULATION BASED ON SCALAR/VECTOR POTENTIALS

Assuming a three-dimensional, inviscid and incompressible flow field, the composite velocity field of Equation (2.4) can be extended as

$$\begin{aligned} \vec{\mathbf{V}} &= \vec{\mathbf{V}}_o + \vec{\mathbf{u}} \\ &= \vec{\mathbf{V}}_o + (\vec{\mathbf{u}}_1 + \vec{\mathbf{u}}_2), \end{aligned} \tag{3.1}$$

where $\vec{\mathbf{u}}_1$ is treated as an irrotational field and $\vec{\mathbf{u}}_2$ as a solenoidal field i.e.:

$$\nabla \times \vec{\mathbf{u}}_1 = 0$$

$$\nabla \cdot \vec{\mathbf{u}}_2 = 0$$

With the above assumptions the *divergence* and *curl* of Equation (3.1) results in the two equalities:

$$\nabla \cdot \vec{\mathbf{V}} = \nabla \cdot \vec{\mathbf{V}}_o + \nabla \cdot \vec{\mathbf{u}}_1 = 0 \tag{3.2a}$$

$$\nabla \times \vec{\mathbf{V}} = \nabla \times \vec{\mathbf{V}}_o + \nabla \times \vec{\mathbf{u}}_2$$

$$= \vec{\zeta}_o + \vec{\zeta}'$$

$$\equiv \vec{\zeta} \tag{3.2b}$$

where $\vec{\zeta}$ in general is not zero and $\vec{\zeta}_o$ is the onset vorticity vector. Equations (3.2) can be rewritten as follows:

$$\nabla \cdot \vec{\mathbf{u}}_1 = -\nabla \cdot \vec{\mathbf{V}}_o = 0 \tag{3.3a}$$

$$\begin{aligned} \nabla \times \vec{\mathbf{u}}_2 &= \nabla \times \vec{\mathbf{V}} - \nabla \times \vec{\mathbf{V}}_o \\ &= \vec{\zeta} - \vec{\zeta}_o \\ &= \vec{\zeta}' \end{aligned} \tag{3.3b}$$

Let the perturbed irrotational velocity field be expressed as

$$\vec{\mathbf{u}}_1 = \nabla \phi \tag{3.4}$$

Then

$$\nabla \cdot \vec{\mathbf{u}}_1 = \nabla^2 \phi = 0 \tag{3.5}$$

The above Laplace equation can be solved for ϕ by Green's formulation (see Hunt [44]). A source distribution is selected for $\vec{\mathbf{u}}_1$ in the present formulation such that

$$\begin{aligned} \phi(\vec{\mathbf{x}}) &= \frac{1}{4\pi} \int_S \int \frac{-\sigma(\vec{\mathbf{x}}_o)}{r(\vec{\mathbf{x}}, \vec{\mathbf{x}}_o)} \, ds \\ &= \int_S \int \theta_\phi \, ds \end{aligned} \tag{3.6}$$

where $\theta_\phi = -\sigma/(4\pi r)$. Let the perturbed solenoidal field be

$$\vec{\mathbf{u}}_2 = \nabla \times \vec{\psi} \tag{3.7}$$

where $\vec{\psi}$ is a divergentless quantity such that

$$\begin{aligned}
\nabla \times \vec{\mathbf{u}}_2 &= \nabla \times (\nabla \times \vec{\psi}) \\
&= \nabla(\nabla \cdot \vec{\psi}) - \nabla^2 \vec{\psi} \\
&= -\nabla^2 \vec{\psi} \\
&= \vec{\zeta} - \vec{\zeta}_o
\end{aligned} \tag{3.8}$$

or

$$\nabla^2 \psi_i = -\zeta_i' \tag{3.9}$$

where i represents the ith component of the vector $\vec{\psi}$. The above Poisson equation is again solved by Green's formulation. A source form is selected for $\vec{\mathbf{u}}_2$. The method of selection is explained later in the next section. The vector potential $\vec{\psi}$ can be expressed as

$$\psi_i(\vec{\mathbf{x}}) = \frac{1}{4\pi} \int_S \int \frac{-\sigma_i(\vec{\mathbf{x}}_o)}{r(\vec{\mathbf{x}}, \vec{\mathbf{x}}_o)} \, ds - \frac{1}{4\pi} \int \int_R \int \frac{-\sigma_i(\vec{\mathbf{x}}_o)}{r(\vec{\mathbf{x}}, \vec{\mathbf{x}}_o)} \, \zeta_i' \, dv$$

$$= \int_S \int \theta_{\psi_i} \, ds - \frac{1}{4\pi} \int \int_R \int \frac{-\sigma_i(\vec{\mathbf{x}}_o)}{r(\vec{\mathbf{x}}, \vec{\mathbf{x}}_o)} \, \zeta_i' \, dv \tag{3.10}$$

where $\theta_{\psi_i} = -\sigma_i/(4\pi r)$. The volume integral in Equation (3.10) is due to the onset vorticity. Therefore, the perturbed velocity field becomes

$$\vec{\mathbf{u}} = \nabla \phi + \nabla \times \vec{\psi} \tag{3.11}$$

or in cartesian tensor notation

$$u_i = \frac{\partial \phi}{\partial x_i} + \epsilon_{ijk} \frac{\partial \psi_j}{\partial x_k}$$

$$= \int_S \int [\frac{\partial \theta_\phi}{\partial x_i} + \epsilon_{ijk} \frac{\partial \theta_{\psi_j}}{\partial x_k}] \; ds$$

$$-\frac{1}{4\pi} \int \int_R \int \epsilon_{ijk} \frac{\partial}{\partial x_k} (\frac{-\sigma_j}{r} \zeta_j') \; dv \qquad (3.12)$$

where the integration is with respect to $\vec{\mathbf{x}}_o$ and the differentiation with respect to $\vec{\mathbf{x}}$. The first integral is determined by applying the usual normal boundary conditions and the second integral is determined by integrating iteratively. The iteration procedure first assumes $\vec{\zeta} = \vec{\zeta}_o$ and calculates a set of vector potentials, then updates the vorticity field and recalculate a new set of vector potentials. The iteration is stopped when the vector potential has converged in the given norm (for example $Max(\triangle\psi) \leq \pm 10^{-4}$).

It is easy to identify the type of singularity for the ϕ distribution, because, for irrotational flow, only the scalar potential ϕ is needed and its perturbed velocity is $\nabla\phi$. However, it is difficult to identify the singularity type associated with $\vec{\psi}$ because its perturbed velocity is given as $\nabla \times \vec{\psi}$. Therefore, in the next section, the relationship between a source and a vortex will be derived in order to identify the type of singularity required to express the vector potential $\vec{\psi}$.

B. SINGULARITIES ASSOCIATED WITH THE VECTOR POTENTIAL

It remains to be shown how singularity strengths can be determined for the vector potential formulation. The form of the vector potential is not readily available from the literature. Hence, in order to identify the type of singularity for the vector potential $\vec{\psi}$, the following derivation of the relationship between source and vortex is presented. It is shown that the *curl* of the source distribution θ_{ψ_i} is equal to the *gradient* of the vortex distribution Γ. Once this equivalence between source and vortex is found, it is possible to use a vortex distribution to simulate the vector potential $\vec{\psi}$ rather than a doublet. The *gradient* of the vortex distribution then gives the the perturbed velocity $\vec{\mathbf{u}}_2$.

For the three-dimensional case, first consider the vector potential component $\vec{\psi}_y$ in the $\hat{j}$ direction. Let the singularity be distributed on a flat plane in the xy plane and be expressed as

$$\vec{\psi}_y = \psi_y\,\hat{j} = \frac{1}{4\pi}\int_S\int \frac{-\sigma_y\,\hat{j}}{r}\,ds, \tag{3.13}$$

where σ_y is piecewise constant. Based on the nomenclature of Figure 3.1, the perturbed velocity due to the vector potential $\vec{\psi}_y$ can then be expressed as

$$\begin{aligned}\vec{\mathbf{u}}_{2y} &= \nabla\times\vec{\psi}_y \\ &= \frac{-\sigma_y}{4\pi}\nabla\times\left(\int_{x_1}^{x_2}\int_{y_1}^{y_2}\frac{\hat{j}}{\sqrt{(x-x_o)^2+(y-y_o)^2+(z-z_o)^2}}\,dy_o\,dx_o\right)\end{aligned} \tag{3.14}$$

where x, y and z are the coordinates for the field point p, and x_o, y_o and z_o are the coordinates of the dummy variable on the singularity surface. Upper limits x_2, y_2 and lower limits x_1, y_1 are the extreme values of the singularity surface. Since the singularity surface is flat on xy plane, the variable z_o does not enter the integration. Note that the subscript y on $\vec{\mathbf{u}}_{2y}$ does not stand for a y component but a perturbation velocity vector due to $\vec{\psi}_y$. Since

$$\nabla\times\left(\frac{\hat{j}}{\sqrt{(x-x_o)^2+(y-y_o)^2+(z-z_o)^2}}\right)$$

$$= \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ \partial_x & \partial_y & \partial_z \\ 0 & 1/\sqrt{(x-x_o)^2+(y-y_o)^2+(z-z_o)^2} & 0 \end{vmatrix}$$

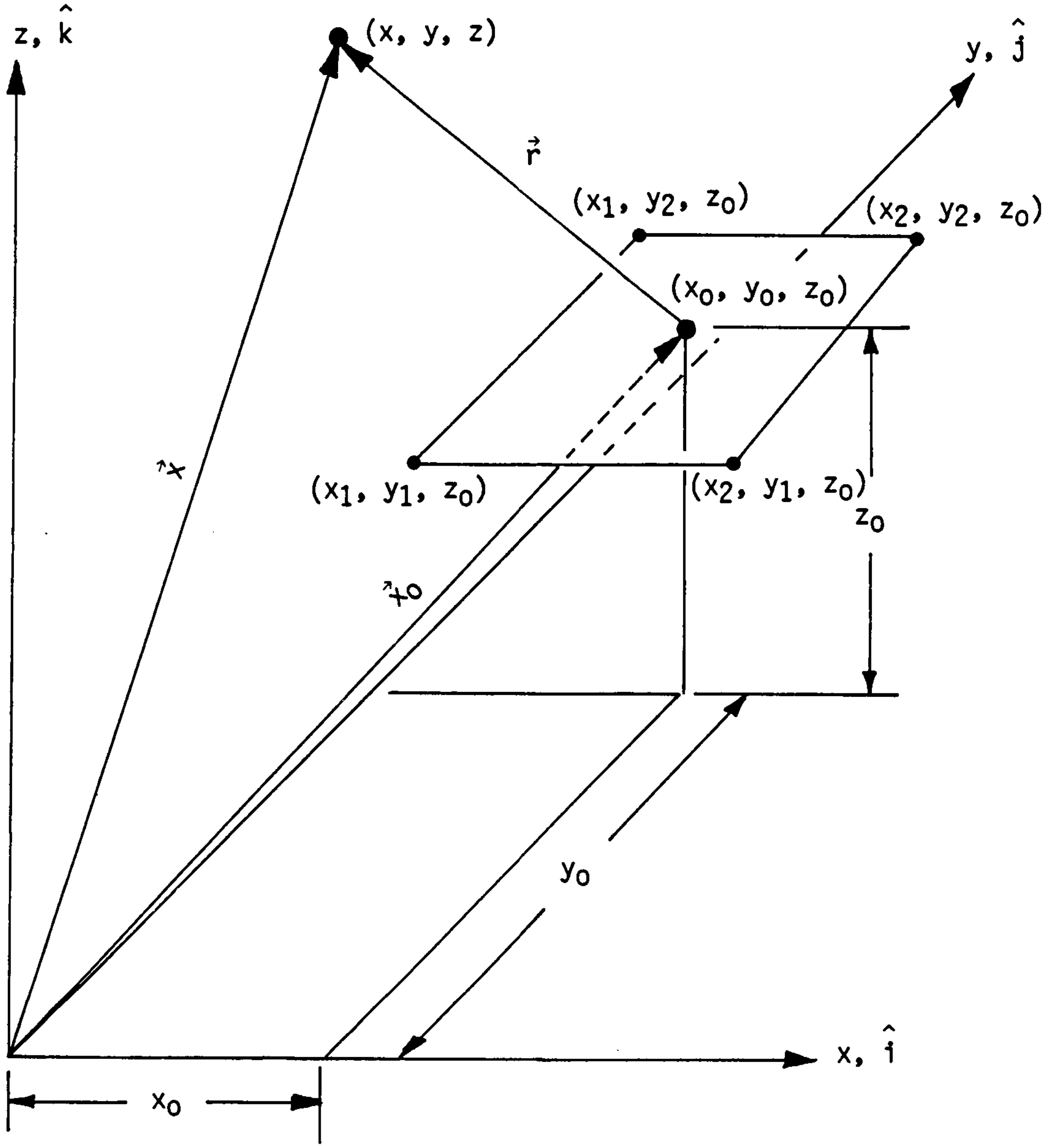


Figure 3.1. Illustration of the coordinate system for a flat singularity surface.

$$= -\hat{\imath}\ \frac{\partial}{\partial z}\left(\frac{1}{\sqrt{(x-x_o)^2+(y-y_o)^2+(z-z_o)^2}}\right)+$$

$$\hat{k}\ \frac{\partial}{\partial x}\left(\frac{1}{\sqrt{(x-x_o)^2+(y-y_o)^2+(z-z_o)^2}}\right)$$

$$= \frac{1}{[(x-x_o)^2+(y-y_o)^2+(z-z_o)^2]^{\frac{3}{2}}}\ [(z-z_o)\ \hat{\imath}-(x-x_o)\ \hat{k}], \tag{3.15}$$

Equation (3.14) can be expressed as

$$\vec{\mathbf{u}}_{2y} = -\frac{\sigma_y}{4\pi}\int_{x_1}^{x_2}\Bigg([(z-z_o)\ \hat{\imath}-(x-x_o)\ \hat{k}]$$
$$\int_{y_1}^{y_2}\frac{dy_o}{[(x-x_o)^2+(y-y_o)^2+(z-z_o)^2]^{\frac{3}{2}}}\Bigg)\ dx_o. \tag{3.16}$$

Since

$$\int_{y_1}^{y_2}\frac{dy_o}{[(x-x_o)^2+(y-y_o)^2+(z-z_o)^2]^{\frac{3}{2}}} =$$

$$\frac{y-y_1}{[(x-x_o)^2+(z-z_o)^2]\sqrt{(x-x_o)^2+(y-y_1)^2+(z-z_o)^2}}$$

$$-\frac{y-y_2}{[(x-x_o)^2+(z-z_o)^2]\ \sqrt{(x-x_o)^2+(y-y_2)^2+(z-z_o)^2}} \tag{3.17}$$

Equation(3.14) can be integrated once as

$$\vec{\mathbf{u}}_{2y} = -\frac{\sigma_y}{4\pi}\int_{x_1}^{x_2}\frac{1}{\sqrt{(x-x_o)^2+(z-z_o)^2}}\ \left(\frac{y-y_1}{\sqrt{(x-x_o)^2+(y-y_1)^2+(z-z_o)^2}}-\right.$$
$$\left.\frac{y-y_2}{\sqrt{(x-x_o)^2+(y-y_2)^2+(z-z_o)^2}}\right)\ \left(\frac{z-z_o}{\sqrt{(x-x_o)^2+(z-z_o)^2}}\hat{\imath}-\right.$$
$$\left.\frac{x-x_o}{\sqrt{(x-x_o)^2+(z-z_o)^2}}\hat{k}\right)\ dx_o \tag{3.18}$$

To show the relationship between a source and a vortex, compare the integrand of Equation (3.18) with the induced velocity due to a finite, discrete vortex filament:

$$\vec{\mathbf{u}}(\ \vec{\mathbf{x}}\ ) = \frac{\Gamma_y}{4\pi h}\ (\ cos(\theta_1) - cos(\theta_2)\ )\ \hat{e} \tag{3.19}$$

Equation (3.19) is the Biot-Savart law where $h = \sqrt{(x-x_o)^2 + (z-z_o)^2}$ and if $-\sigma_y = \Gamma_y$, the integrand of Equation (3.18) is equivalent to that of Equation (3.19) where $cos(\theta_1)$, $cos(\theta_2)$ and $\hat{\mathbf{e}}$ are expressed as function of (x, y, z) and (x_o, y_o, z_o) as can be seen from Figure 3.2. This means that the y component of the vector potential $\vec{\psi}$ can be simulated by a piecewise constant vortex filament distribution as shown in Figure 3.3.

Similarly, the x component of the vector potential can be expressed as

$$\vec{\psi}_x = \psi_x\ \hat{i} = \frac{1}{4\pi}\int_S\int \frac{-\sigma_x\ \hat{i}}{r}\ ds \tag{3.20}$$

and its perturbed velocity as

$$\begin{aligned}\vec{\mathbf{u}}_{2x} &= \nabla \times \vec{\psi}_x \\ &= \frac{-\sigma_x}{4\pi}\nabla\times\left(\int_{y_1}^{y_2}\int_{x_1}^{x_2} \frac{\hat{i}}{\sqrt{(x-x_o)^2+(y-y_o)^2+(z-z_o)^2}}\ dx_o\ dy_o\right).\end{aligned} \tag{3.21}$$

Since

$$\int_{x_1}^{x_2} \frac{dx_o}{[(x-x_o)^2+(y-y_o)^2+(z-z_o)^2]^{\frac{3}{2}}} =$$

$$\frac{x-x_1}{[(y-y_o)^2+(z-z_o)^2]\sqrt{(x-x_1)^2+(y-y_o)^2+(z-z_o)^2}}$$

$$-\frac{x-x_2}{[(y-y_o)^2+(z-z_o)^2]\ \sqrt{(x-x_2)^2+(y-y_o)^2+(z-z_o)^2}} \tag{3.22}$$

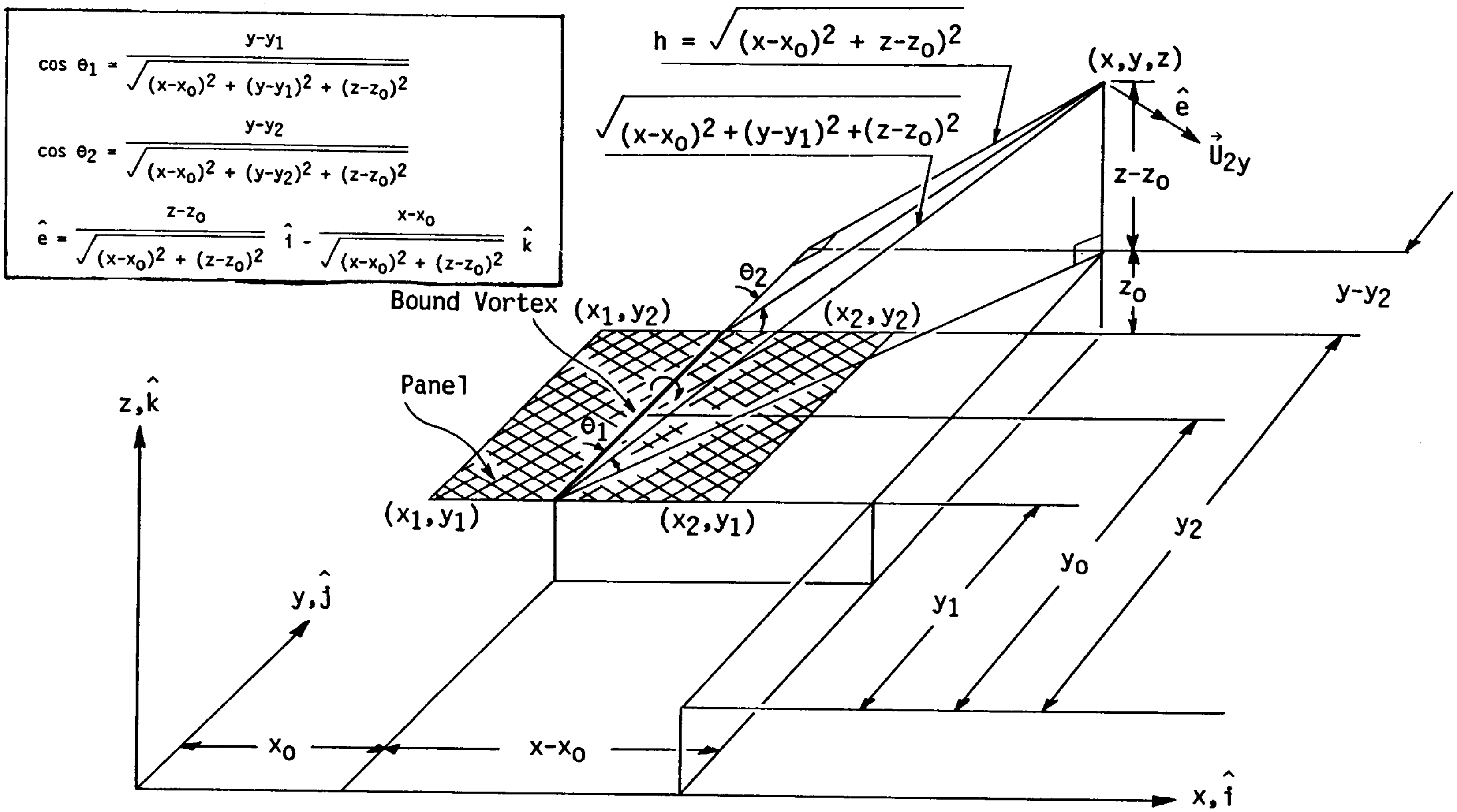


Figure 3.2. Illustration of the induced velocity of a bound vortex filament.

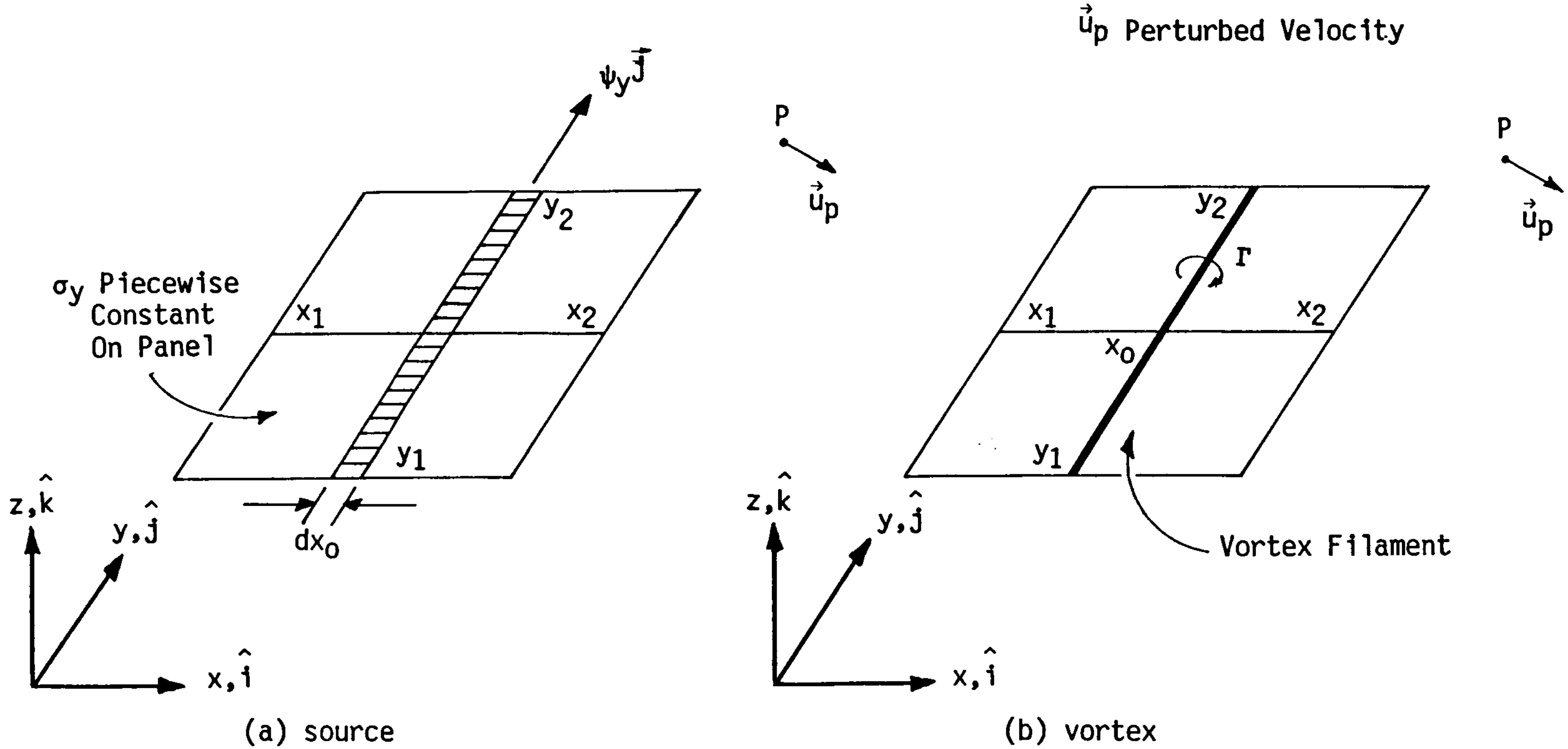


Figure 3.3. Equivalence of a source and a vortex filament used for vector potential $\vec{\psi}_y$.

Equation (3.21) can be integrated once as

$$\vec{\mathbf{u}}_{2x} = -\frac{\sigma_x}{4\pi}\int_{y_1}^{y_2} \Big(\frac{x - x_1}{[(y-y_o)^2 + (z-z_o)^2]\sqrt{(x-x_1)^2 + (y-y_o)^2 + (z-z_o)^2}} -$$

$$\frac{x - x_2}{[(y-y_o)^2 + (z-z_o)^2]\sqrt{(x-x_2)^2 + (y-y_o)^2 + (z-z_o)^2}}\Big)\times$$

$$[-(z-z_o)\hat{j} + (y-y_o)\hat{k}]\ dy_o \tag{3.23}$$

Thus for an individual panel the relationship is equivalent to Equation (3.19), $\sigma_x = -\Gamma$. In addition, the wake can be simulated by letting $x_2 \rightarrow \infty$ for a panel at the wing trailing edge. (See Figure 3.4)

$$\lim_{x_2\rightarrow\infty} \frac{x - x_2}{[(y-y_o)^2 + (z-z_o)^2]\sqrt{(x-x_2)^2 + (y-y_o)^2 + (z-z_o)^2}}$$

$$= \lim_{x_2\rightarrow\infty} \frac{(x - x_2)/x_2}{[(y-y_o)^2 + (z-z_o)^2]\sqrt{[(x-x_2)/x_2]^2 + [(y-y_o)^2 + (z-z_o)^2]/x_2^2}}$$

$$= \frac{-1}{(y-y_o)^2 + (z-z_o)^2} \tag{3.24}$$

Substituting Equation (3.24) into Equation (3.23) gives

$$\vec{\mathbf{u}}_{2x} = \frac{\sigma_x}{4\pi}\int_{y_1}^{y_2} \frac{1}{\sqrt{(y-y_o)^2 + (z-z_o)^2}}\ \Big(1 + \frac{x - x_1}{\sqrt{(x-x_1)^2 + (y-y_o)^2 + (z-z_o)^2}}\Big)$$

$$\times\ \Big(\ \frac{z - z_o}{\sqrt{(y-y_o)^2 + (z-z_o)^2}}\hat{j} - \frac{y - y_o}{\sqrt{(y-y_o)^2 + (z-z_o)^2}}\hat{k}\ \Big)\ dy_o, \tag{3.25}$$

The integrand of Equation (3.25) can also be compared with the induced velocity due to a semi-infinite discrete vortex filament,

$$\vec{\mathbf{u}}(\ \vec{\mathbf{x}}\ ) = \frac{\Gamma_x}{4\pi h}\ (\ 1 + cos(\theta_o)\ )\ \hat{e} \tag{3.26}$$

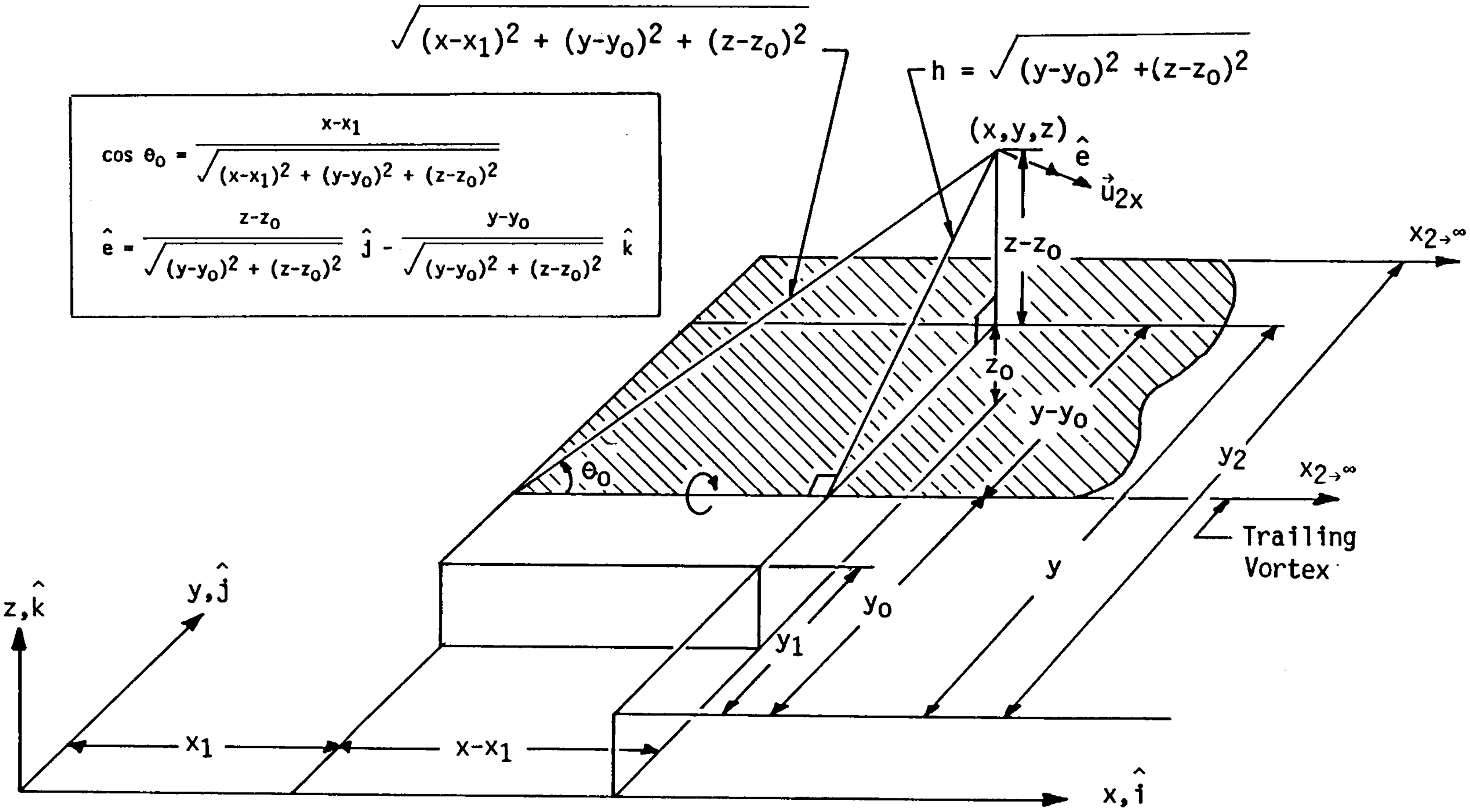


Figure 3.4. Illustration of the induced velocity of a trailing vortex filament making up the wake.

obtained by the Biot-Savart law, where θ_o and $\hat{\mathbf{e}}$ are defined in Figure 3.4. This also means that the contribution from the wake simulated by a semi-infinite strip source distribution for the vector potential $\vec{\psi}_x$ can be replaced by a semi-infinite vortex strip, as shown in Figure 3.5.

The contribution of the vector potential $\vec{\psi}_z$ to the induced velocity can be derived by a similar procedure and the composition of the vector potentials $\vec{\psi}_x$, $\vec{\psi}_y$, and $\vec{\psi}_z$ can be simulated as shown in Figure 3.6, that is, as vortex panels on the wing surface and as vortex filaments on the wake surface. In Figure 3.6, the wake roll-up phenomena is neglected in order to avoid wake relaxation calculation. According to Hess [44] the wake geometry may be assumed known and parallel to the chord surface of the wing in the near wing region and then parallel to the direction of the oncoming flow for the further downstream.

C. GEOMETRY AND NUMERICAL MODELS

The global and local coordinates of the present formulation are shown on Figure 3.7 where b is the wing span. The global coordinate is situated on the mid span of the wing with the x and y axes along the chord and span directions respectively. The local coordinate system on each panel is oriented in a similar manner as the global coordinate. A field point $p(x_i, y_i, z_i)$, either the control point on the panels or the grid point in the domain, can be expressed in terms of the local coordinate variables $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$ as follows:

$$\begin{bmatrix} \bar{x}_i \\ \bar{y}_i \\ \bar{z}_i \end{bmatrix} = \begin{pmatrix} \hat{\imath} \cdot \hat{e}_x & \hat{\jmath} \cdot \hat{e}_x & \hat{k} \cdot \hat{e}_x \\ \hat{\imath} \cdot \hat{e}_y & \hat{\jmath} \cdot \hat{e}_y & \hat{k} \cdot \hat{e}_y \\ \hat{\imath} \cdot \hat{e}_z & \hat{\jmath} \cdot \hat{e}_z & \hat{k} \cdot \hat{e}_z \end{pmatrix} \begin{bmatrix} x_i - x_{io} \\ y_i - y_{io} \\ z_i - z_{io} \end{bmatrix} \tag{3.27}$$

where (x_{io}, y_{io}, z_{io}) is the origin of the local system measured in the global coordinates (where $\hat{\imath}, \hat{\jmath}, \hat{k}$ are the unit vectors in global coordinates and $\hat{e}_x, \hat{e}_y, \hat{e}_z$ are the unit vectors in the local coordinates as shown in Figure 3.7).

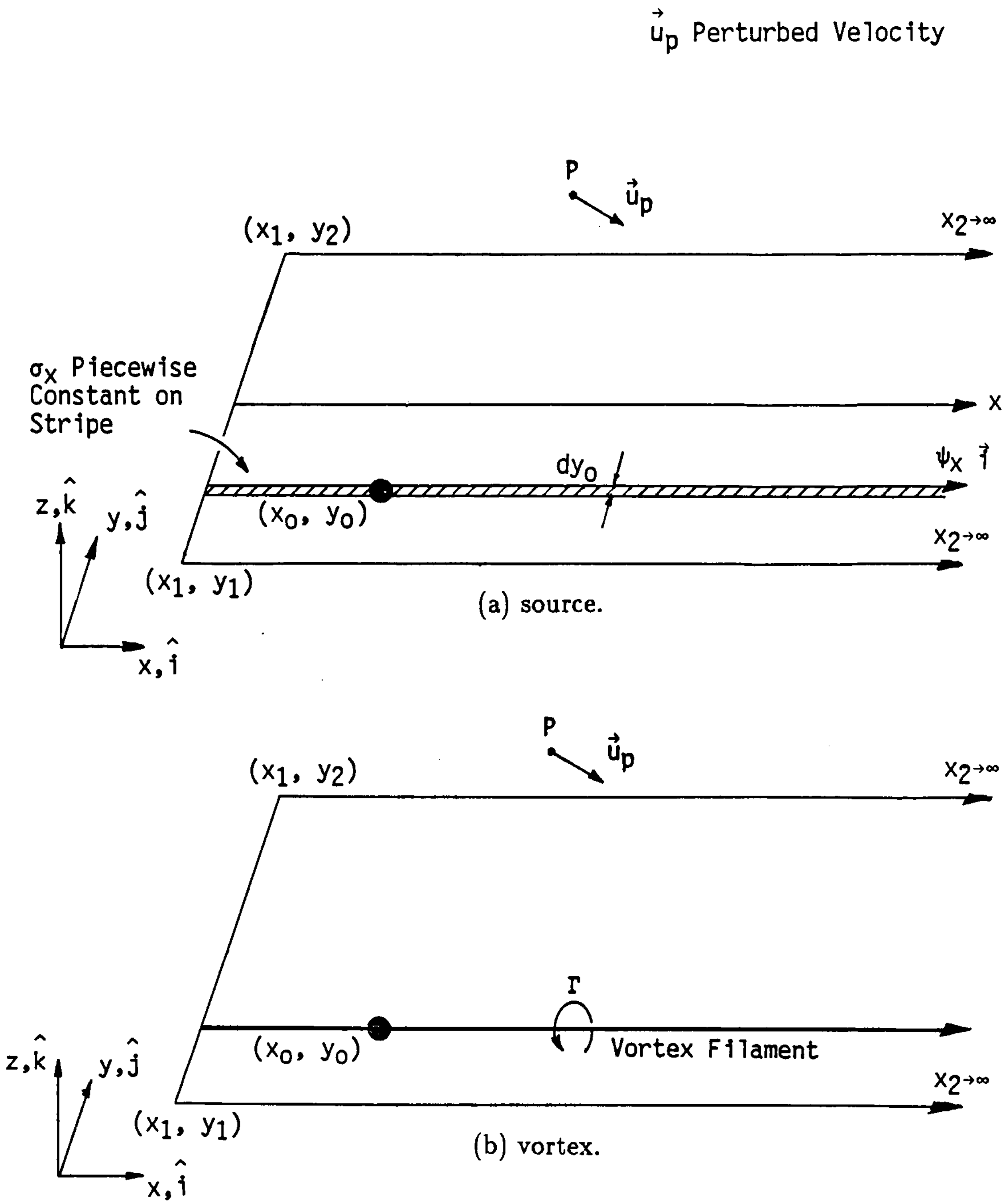


Figure 3.5. Equivalence of a source and a vortex filament used for vector potential $\vec{\psi}_x$.

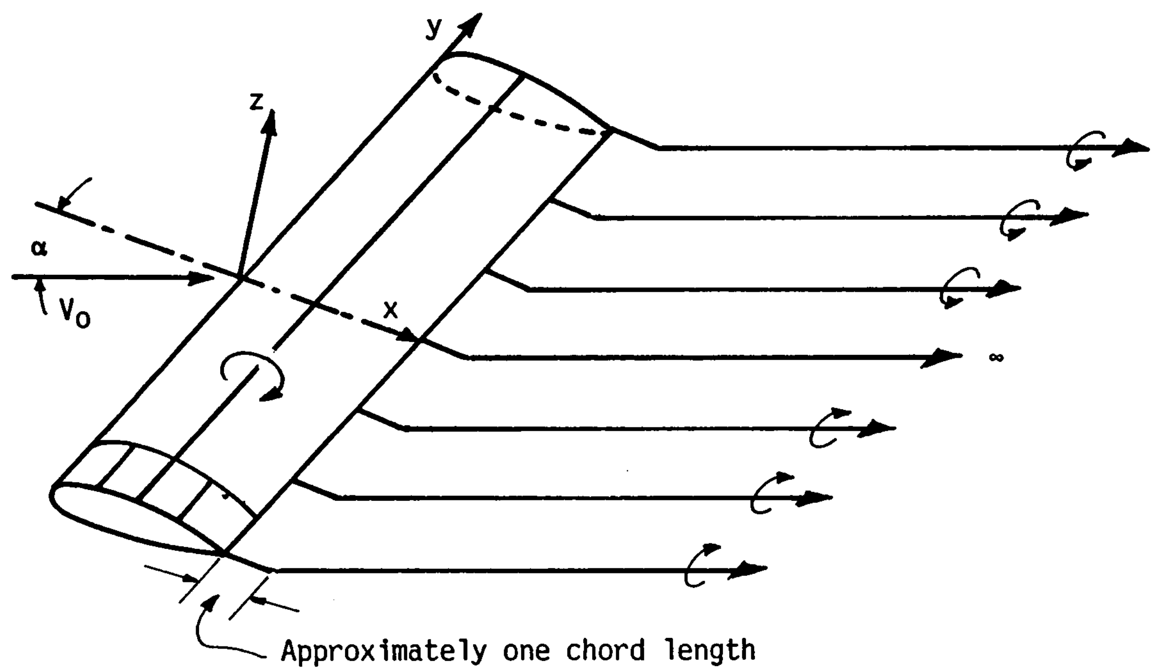


Figure 3.6. Vortex models for vector potential $\vec{\psi}$.

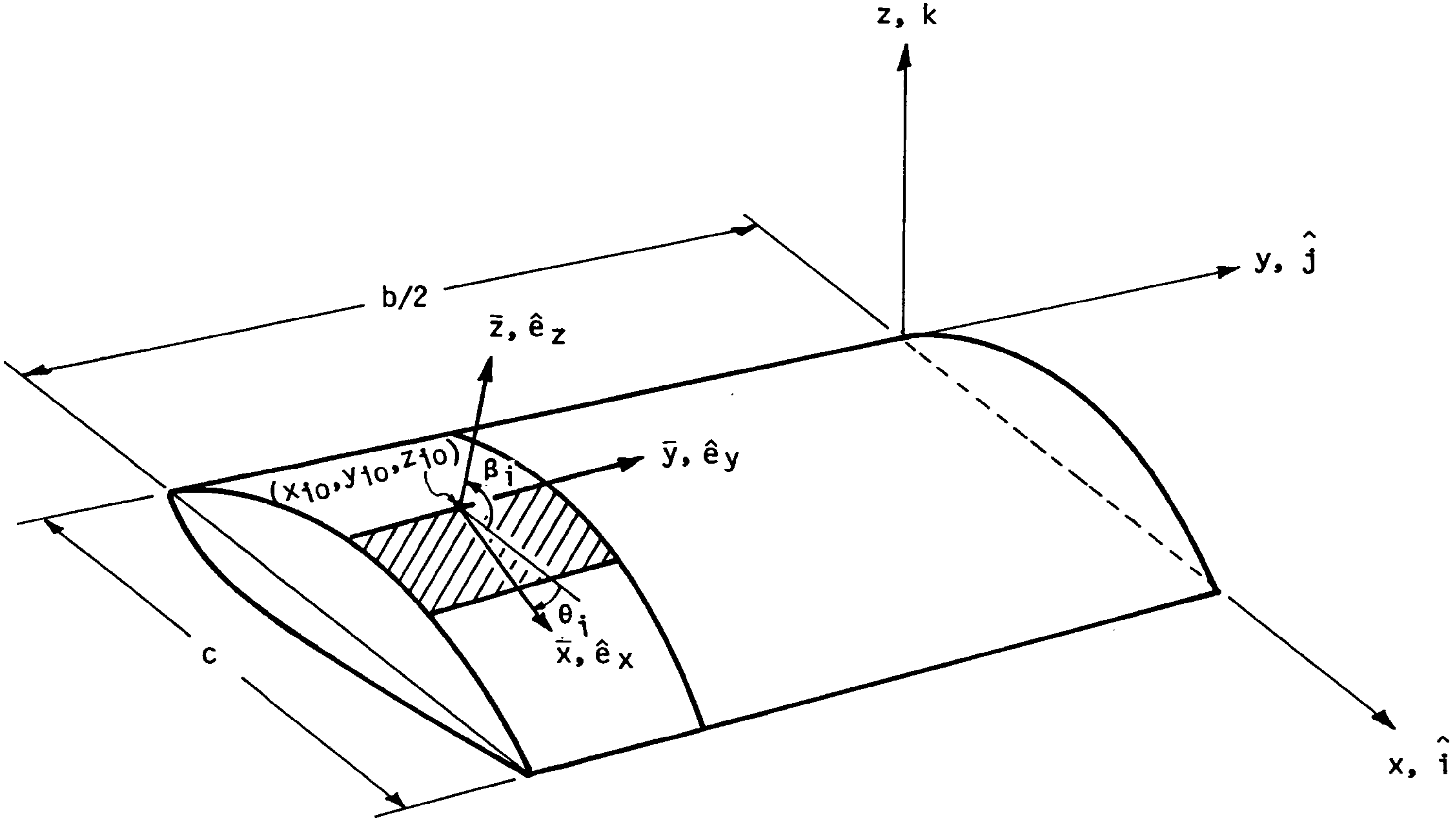


Figure 3.7. Global and local coordinate systems.

A rectangular finite wing will be used as the model. According to the previous derivation, a piecewise constant source distribution is used for the scalar potential ϕ and a piecewise constant vortex distribution for the vector potential $\vec{\psi}$ (see Figure 3.8). The wake of the wing is assumed to be fixed and known (parallel to the chord line for one to two chord lengths and then parallel to the free stream) in order to avoid the wake relaxation calculations. For a two-dimensional airfoil, piecewise constant source and vortex distributions are distributed on each flat panel. No wake specification is needed for the two-dimensional flow problems and the Kutta condition is applied to render a unique solution for the two-dimensional airfoil as mentioned in Chapter II. For a three-dimensional wing the Kutta condition is imposed at each trailing edge section.

In the next two sections, the mathematical formulation for the rectangular finite wing and the two-dimensional airfoil, based on the numerical models introduced in this section, is presented. Since there are many similarities in procedures between the two-dimensional airfoil and the rectangular finite wing, the simpler two-dimensional airfoil is introduced first.

D. TWO-DIMENSIONAL AIRFOIL

For two-dimensional flow, there is only one vector potential $\vec{\psi}$ in the $\hat{j}$ direction. Through a similar Green's formulation as that for the three-dimensional case, Equations (3.6) and (3.10), the two-dimensional scalar and vector potentials can be expressed as

$$\phi_p = \frac{1}{2\pi} \int_C \sigma \ln(r)\, dl, \tag{3.28}$$

and

$$\psi = \frac{1}{2\pi} \left(\int_C \sigma_y \ln(r)\, dl - \int_S \int \ln(r)(\zeta_y - \zeta_{yo})\, ds \right) \tag{3.29}$$

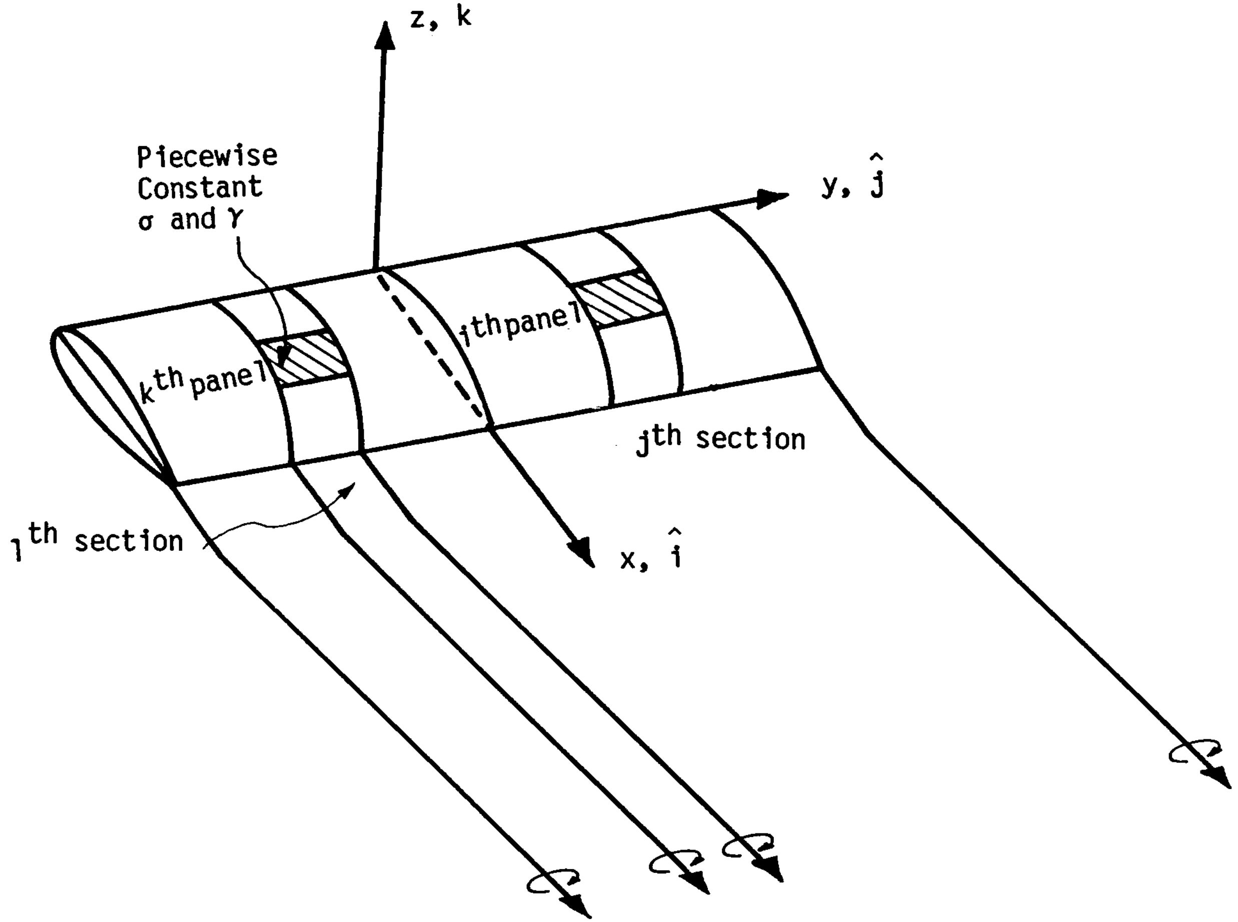


Figure 3.8. Piecewise constant source and vortex distribution for scalar and vector potentials.

where σ and σ_y are piecewise constant distributions on each flat panel, and ζ_y and ζ_{yo} are the present and onset vorticity, respectively. The C and S of the integration stand for the integration path along the airfoil surface and throughout the two-dimensional domain respectively. The perturbed velocities associated with ϕ and ψ can be expressed as

$$\vec{\mathbf{u}}_1 = \frac{1}{2\pi}\, \nabla\Big(\int_C \sigma \ln(r)\, dl\Big) \tag{3.30}$$

and

$$\vec{\mathbf{u}}_2 = \nabla \times \frac{\hat{j}}{2\pi}\Big(\int_C \sigma_y \ln(r)\, dl - \int_S\int \ln(r)(\zeta_y - \zeta_{yo})\, ds\Big). \tag{3.31}$$

According to the derived relationship between a source and a vortex, Equations (3.18) and (3.19), $\vec{\mathbf{u}}_2$ can be further expressed as

$$\vec{\mathbf{u}}_2 = \frac{-1}{2\pi}\, \nabla\Big(\int_C \gamma\Theta\, dl\Big) - \nabla \times \Big(\frac{\hat{j}}{2\pi}\int_S\int \ln(r)(\zeta_y - \zeta_{yo})\, ds\Big). \tag{3.32}$$

in which a piecewise constant vortex distribution (γ) is used to simulate the vector potential $\vec{\psi}_y$, and Θ is the expression for the vortex distribution given as $tan^{-1}((z_i - z_j)/(x_i - x_j))$ [51]. The perturbed velocities $\vec{\mathbf{u}}_1$ and $\vec{\mathbf{u}}_2$ can be discretized as

$$\vec{\mathbf{u}}_1 = \sum_j \Big(\frac{\sigma_j}{2\pi}\int_j \nabla_i(\ln(r_{ij}))\, dl_j\Big) \tag{3.30$'$}$$

and

$$\vec{\mathbf{u}}_2 = -\sum_j \Big(\frac{\gamma_j}{2\pi}\int_j \nabla_i(\Theta_{ij})\, dl_j\Big) - \nabla \times \Big(\frac{\hat{j}}{2\pi}\int_S\int \ln(r)(\zeta_y - \zeta_{yo})\, ds\Big) \tag{3.32$'$}$$

where subscript i stands for the ith control point (field point p) and subscript j stands for the jth control point (dummy variable q, see Figure 3.9).

Applying the normal boundary condition,

$$\vec{\mathbf{V}}_o \cdot \vec{\mathbf{n}}_i + (\vec{\mathbf{u}}_1 + \vec{\mathbf{u}}_2) \cdot \vec{\mathbf{n}}_i = 0 \tag{3.33}$$

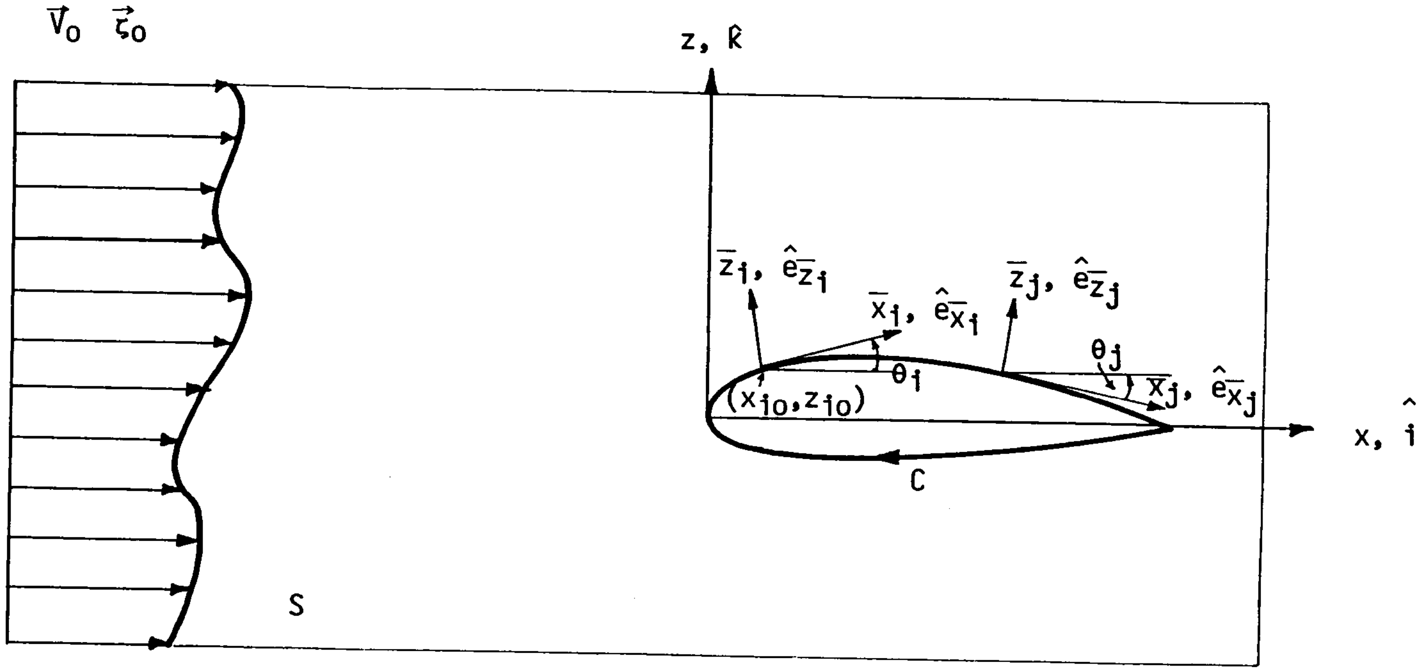


Figure 3.9. An airfoil in shear flow.

where $\vec{\mathbf{n}}_i$ stands for the unit outward normal on the ith panel, then Equation (3.33) can be split as follows

$$\sum_j \left[\frac{\sigma_j}{2\pi} \int_j \nabla_i(\ln(r_{ij})) \cdot \vec{\mathbf{n}}_i \; dl_j \right] = -\vec{\mathbf{V}}_{0_t} \cdot \vec{\mathbf{n}}_i \tag{3.34}$$

and

$$-\sum_j \left[\frac{\gamma_j}{2\pi} \int_j \nabla_i(\Theta_{ij}) \cdot \vec{\mathbf{n}}_i \; dl_j \right] = -\vec{\mathbf{V}}_{0_n} \cdot \vec{\mathbf{n}}_i + \nabla_i \times \left(\frac{\hat{j}}{2\pi} \int_S \int \ln(r)(\zeta_y - \zeta_{yo}) \; ds \right) \cdot \vec{\mathbf{n}}_i \tag{3.35}$$

where $\vec{\mathbf{V}}_{0_t}$ and $\vec{\mathbf{V}}_{0_n}$ are defined as in Figure 2.9 page 32 and Equations (2.20) and (2.21). Equations (3.34) and (3.35) can be further expressed in matrix forms,

$$\left(\mathbf{I}^n_{ij} \right) \left[\frac{\sigma_j}{2\pi} \right] = \left[\mathbf{R}_{i1} \right] \tag{3.36}$$

$$\left(\mathbf{D}^n_{ij} \right) \left[\frac{\gamma_j}{2\pi} \right] = \left[\mathbf{R}_{i2} \right] \tag{3.37}$$

where $\mathbf{I}^n_{ij}$ and $\mathbf{D}^n_{ij}$ are the influence matrices for the normal components of the source and the vortex distributions and $\mathbf{R}_{i1} = -\vec{\mathbf{V}}_{0_t} \cdot \vec{\mathbf{n}}_i$ and $\mathbf{R}_{i2} = -\vec{\mathbf{V}}_{0_n} \cdot \vec{\mathbf{n}}_i + \nabla_i \times [\hat{j}/(2\pi) \int_S \int \ln(r)(\zeta_y - \zeta_{yo}) \; ds] \cdot \vec{\mathbf{n}}_i$ The influence matrices $\mathbf{I}^n_{ij}$ and $\mathbf{D}^n_{ij}$ can be expressed and integrated analytically as follows (Note :dl_j replaced by ds_j):

$$\begin{aligned}
\mathbf{I}^n_{ij} &= \int_j \nabla_i(\ln r_{ij}) \cdot \mathbf{n}_i \; ds_j \\
&= \int_j \frac{\partial}{\partial n_i}(\ln r_{ij}) \; ds_j \\
&= \int_j \frac{\bar{x}_i sin(\theta_j - \theta_i) + \bar{z}_i cos(\theta_j - \theta_i) - s_j sin(\theta_j - \theta_i)}{(\bar{x}_i^2 + \bar{z}_i^2) - 2\bar{x}_i s_j + s_j^2} \; ds_j \\
&= \int_j \frac{d + e s_j}{a + b s_j + c s_j^2} \; ds_j \qquad (i \neq j)
\end{aligned} \tag{3.38}$$

In Equation (3.38), $\bar{x}_i$ and $\bar{z}_i$ are transformed in terms of local coordinates (see Equation (3.27)), and θ_i and θ_j are defined as in Figure 3.9. The definitions of the quantities a through e are

$$\begin{aligned}
a &= \bar{x}_i^2 + \bar{z}_i^2 \\
b &= -2\bar{x}_i \\
c &= 1 \\
d &= \bar{x}_i sin(\theta_j - \theta_i) + \bar{z}_i cos(\theta_j - \theta_i) \\
e &= -sin(\theta_j - \theta_i)
\end{aligned} \tag{3.39}$$

Also

$$\begin{aligned}
R &= a + bs_j + cs_j^2 \\
\Delta &= 4ac - b^2 = 4\bar{z}_i^2
\end{aligned}$$

For $\Delta > 0$,

$$\begin{aligned}
\mathbf{I}_{ij}^n &= \frac{e}{2c}[\ln(R)]_0^{l_j} + (d - \frac{be}{2c})\frac{2}{\sqrt{\Delta}} tan^{-1}[\frac{b + 2cs_j}{\sqrt{\Delta}}]_0^{l_j} \\
&= \frac{1}{2} sin(\theta_i - \theta_j)\ln\left(1 + \frac{l_j^2 - 2\bar{x}_i l_j}{\bar{x}_i^2 + \bar{z}_i^2}\right) \\
&+ cos(\theta_i - \theta_j)\left[tan^{-1}(\frac{l_j - \bar{x}_i}{\bar{z}_i}) - tan^{-1}(\frac{-\bar{x}_i}{\bar{z}_i})\right] \quad (i \neq j)
\end{aligned} \tag{3.40a}$$

and for $\Delta = 0$,

$$\begin{aligned}
\mathbf{I}_{ij}^n &= \frac{e}{2c}[\ln(R)]_0^{l_j} + (d - \frac{be}{2c})[\frac{-2}{b + 2cs_j}]_0^{l_j} \\
&= \frac{1}{2} sin(\theta_i - \theta_j)\ln\left(1 + \frac{l_j^2 - 2\bar{x}_i l_j}{\bar{x}_i^2 + \bar{z}_i^2}\right) \\
&= sin(\theta_i - \theta_j)\ln\left|\frac{l_j - \bar{x}_i}{\bar{x}_i}\right| \quad (i \neq j)
\end{aligned} \tag{3.40b}$$

where l_j is the panel length for the jth panel. For $i = j$, $\mathbf{I}_{ii}^n = \pi$, see Appendix B. Similarly, for the influence matrix $\mathbf{D}_{ij}^n$,

$$
\begin{aligned}
\mathbf{D}^n_{ij} &= -\int_j \nabla_i(\Theta_{ij}) \cdot \mathbf{n}_i \, ds_j \\
&= -\int_j \nabla_i \left[tan^{-1}(\frac{z_i - z_j}{x_i - x_j}) \right] \cdot \mathbf{n}_i \, ds_j \\
&= -\int_j \frac{\partial}{\partial n_i} \left[tan^{-1}(\frac{z_i - z_j}{x_i - x_j}) \right] ds_j \\
&= -\int_j \frac{\bar{x}_i cos(\theta_j - \theta_i) - \bar{z}_i sin(\theta_j - \theta_i) - s_j cos(\theta_j - \theta_i)}{(\bar{x}_i^2 + \bar{z}_i^2) - 2\bar{x}_i s_j + s_j^2} ds_j \\
&= \int_j \frac{d + e s_j}{a + b s_j + c s_j^2} ds_j \qquad (i \neq j) \qquad (3.41)
\end{aligned}
$$

where

$$
\begin{aligned}
a &= \bar{x}_i^2 + \bar{z}_i^2 \\
b &= -2\bar{x}_i \\
c &= 1 \\
d &= -\bar{x}_i cos(\theta_j - \theta_i) + \bar{z}_i sin(\theta_j - \theta_i) \\
e &= cos(\theta_j - \theta_i) \qquad (3.42)
\end{aligned}
$$

For $\Delta > 0$,

$$
\begin{aligned}
\mathbf{D}^n_{ij} &= \frac{e}{2c}[\ln(R)]_0^{l_j} + (d - \frac{be}{2c})\frac{2}{\sqrt{\Delta}} tan^{-1}[\frac{b + 2cs_j}{\sqrt{\Delta}}]_0^{l_j} \\
&= \frac{1}{2} cos(\theta_i - \theta_j) \ln\left(1 + \frac{l_j^2 - 2\bar{x}_i l_j}{\bar{x}_i^2 + \bar{z}_i^2}\right) \\
&\quad - sin(\theta_i - \theta_j)\left[tan^{-1}(\frac{l_j - \bar{x}_i}{\bar{z}_i}) - tan^{-1}(\frac{-\bar{x}_i}{\bar{z}_i})\right] \quad (i \neq j) \qquad (3.43a)
\end{aligned}
$$

and for $\Delta = 0$,

$$
\begin{aligned}
\mathbf{D}^n_{ij} &= \frac{e}{2c}[\ln(R)]_0^{l_j} + (d - \frac{be}{2c})[\frac{-2}{b + 2cs_j}]_0^{l_j} \\
&= \frac{1}{2} cos(\theta_i - \theta_j) \ln\left(1 + \frac{l_j^2 - 2\bar{x}_i l_j}{\bar{x}_i^2 + \bar{z}_i^2}\right) \\
&= cos(\theta_i - \theta_j) \ln \mid \frac{l_j - \bar{x}_i}{\bar{x}_i} \mid \quad (i \neq j) \qquad (3.43b)
\end{aligned}
$$

For $i = j$, the singular integrations of $\mathbf{D}^n_{ii}$ gives a value of zero (see Appendix B).

The discrete expressions for $\nabla_i \times [\hat{j}/(2\pi) \int_S \int \ln(r)(\zeta_y - \zeta_{yo})\, ds] \cdot \vec{\mathbf{n}}_i$ in $\mathbf{R}_{i2}$ is as follows:

$$\nabla_i \times \left(\frac{\hat{j}}{2\pi} \int_S \int \ln(r)(\zeta_y - \zeta_{yo}) ds \right) \cdot \vec{\mathbf{n}}_i = \frac{-1}{2\pi} \int_S \int \frac{(x_i - \tilde{x})cos(\theta_i) + (y_i - \tilde{y})sin(\theta_i)}{(x_i - \tilde{x})^2 + (y_i - \tilde{y})^2}$$
$$(\zeta_y(\tilde{x}, \tilde{y}) - \zeta_{yo})\, d\tilde{x} d\tilde{y} \tag{3.44}$$

In solving for σ and γ in Equations (3.36) and (3.37), an iteration procedure is used. The first iteration assumes that $\vec{\zeta}^0$ equals to $\vec{\zeta}_o$ for which the surface integral (see Equation (3.35)) is zero. Then, the matrix of Equations (3.36) and (3.37) is inverted for the values of σ and γ^1. The calculated σ and γ^1 are used to update the velocity field and then the updated velocity field (implying an updated vorticity field) is substituted into the right-hand side of Equation (3.37) to obtain a γ^2 value. The iteration procedure is repeated until $Max(\gamma^{n+1} - \gamma^n)$ does not change significantly, say by no more than $\pm 10^{-4}$. The convergence criterion of $\pm 10^{-4}$ has been compared with more strict convergence criterion such as $\pm 10^{-6}$. It is found that the results for C_p do not change beyond the $4th$ decimal place and the iteration converges approximately 10 times faster based on the former criterion than the latter.

Once the strengths of σ and γ are determined, the magnitude of the tangential velocity on the airfoil surface can be obtained as

$$\begin{aligned} V_{si} &= \vec{\mathbf{V}}_o \cdot \vec{\mathbf{t}}_i + (\vec{\mathbf{u}}_1 + \vec{\mathbf{u}}_2) \cdot \vec{\mathbf{t}}_i \\ &= \vec{\mathbf{V}}_o \cdot \vec{\mathbf{t}}_i + \left(\mathbf{I}^t_{ij} \right) \left[\frac{\sigma_j}{2\pi} \right] + \left(\mathbf{D}^t_{ij} \right) \left[\frac{\gamma_j}{2\pi} \right] \\ &+ \nabla \times \left(\frac{\hat{j}}{2\pi} \int_S \int \ln(r)(\zeta_y - \zeta_{yo}) ds \right) \cdot \vec{\mathbf{t}}_i \end{aligned} \tag{3.45}$$

where

$$\mathbf{I}^t_{ij} = \int_j \nabla_i(\ln r_{ij}) \cdot \mathbf{t}_i \; ds_j$$
$$= \int_j \frac{\partial}{\partial t_i}(\ln r_{ij}) \; ds_j \tag{3.46}$$

and

$$\mathbf{D}^t_{ij} = -\int_j \nabla_i(\Theta_{ij}) \cdot \mathbf{t}_i \; ds_j$$
$$= -\int_j \nabla_i \left[tan^{-1}(\frac{z_i - z_j}{x_i - x_j}) \right] \cdot \mathbf{t}_i \; ds_j$$
$$= -\int_j \frac{\partial}{\partial t_i} \left[tan^{-1}(\frac{z_i - z_j}{x_i - x_j}) \right] ds_j \tag{3.47}$$

For two-dimensional calculations, the following symmetric relationships hold (see Appendix C):

$$\mathbf{I}^t_{ij} = -\mathbf{D}^n_{ij}$$
$$\mathbf{D}^t_{ij} = \mathbf{I}^n_{ij} \tag{3.48}$$

Once the magnitude of the tangential velocity is obtained, the pressure distribution can be computed from Equations (2.9) or (2.10).

E. FINITE WING

In this section, the formulation for a rectangular finite wing will be presented. Piecewise constant source and vortex distributions are used to simulate the scalar potential ϕ and vector potential $\vec{\psi}$ respectively. Similar procedures to those in the last section are performed here.

First, the perturbed velocities $\vec{\mathbf{u}}_1$ and $\vec{\mathbf{u}}_2$ can be expressed as

$$\vec{\mathbf{u}}_1 = \nabla\phi \tag{3.49}$$

$$\vec{\mathbf{u}}_2 = \nabla \times \vec{\psi} \tag{3.50}$$

The normal boundary condition is applied to $\vec{\mathbf{u}}_1$ as usual such that

$$\begin{aligned}
\vec{\mathbf{u}}_1 \cdot \vec{\mathbf{n}}_{kl} &= \sum_j \sum_i \nabla_{kl}(\phi) \cdot \vec{\mathbf{n}}_{kl} \\
&= \sum_j \sum_i \left[\frac{\sigma_{ij}}{4\pi} \int_j \int_i \nabla_{kl}\left(\frac{-1}{r_{kl,ij}}\right) \cdot \vec{\mathbf{n}}_{kl}\, ds_{ij} \right] \\
&= \sum_j \sum_i \left(\mathbf{I}^n_{kl,ij}\right) \left[\frac{\sigma_{ij}}{4\pi}\right] \qquad (3.51)
\end{aligned}$$

that is,

$$\left(\mathbf{I}^n_{kl,ij}\right) \left[\frac{\sigma_{ij}}{4\pi}\right] = -\vec{\mathbf{V}}_{0_t} \cdot \vec{\mathbf{n}}_{kl} \qquad (3.52)$$

Subscripts k, l stand for the kth panel in the lth section for field point p and subscripts i, j stand for the ith panel in the jth section for point q on the singularity surface (see Figure 3.8, page 62). The influence matrix $\mathbf{I}^n_{kl,ij}$ can be expressed as

$$\begin{aligned}
\mathbf{I}^n_{kl,ij} &= \int_j \int_i \nabla_{kl}\left(\frac{-1}{r_{kl,ij}}\right) \cdot \mathbf{n}_{kl}\, ds_{ij} \\
&= \int_j \int_i \frac{(\bar{x}_{kl} - s_{ij}) sin(\theta_{ij} - \theta_{kl}) + \bar{z}_{kl} cos(\theta_{ij} - \theta_{kl})}{[(\bar{x}_{kl} - s_{ij})^2 + (\bar{y}_{kl} - t_{ij})^2 + \bar{z}^2_{kl}]^{\frac{3}{2}}}\, ds_{ij}\, dt_{ij} \qquad (3.53)
\end{aligned}$$

where

$$r_{kl,ij} = \sqrt{(x_{kl} - x_{ij})^2 + (y_{kl} - y_{ij})^2 + (z_{kl} - z_{ij})^2}$$

The *bar* on top of coordinates indicate that they have been transformed in terms of local coordinates. After lengthy and complicated integrations, the analytical expression for the influence matrix $\mathbf{I}^n_{kl,ij}$ is as follows. For $\bar{z}_{kl} \neq 0$,

$$\begin{aligned}
\mathbf{I}^n_{kl,ij} = {} & sign(\bar{x}_{kl} - \bar{s}_{ij})\ sign(\bar{z}_{kl})\ cos(\theta_{ij} - \theta_{kl})\ [\\
& tan^{-1}(\bar{y}_{kl,r} \sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 / (\bar{z}^2_{kl}[(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,r} + \bar{z}^2_{kl}])}\,) \\
& - tan^{-1}(\bar{y}_{kl,l} \sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 / (\bar{z}^2_{kl}[(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,l} + \bar{z}^2_{kl}])}\,)\ ] - \ sin(\theta_{ij} - \theta_{kl}) \\
& \ln[\ (\bar{y}_{kl,r} + \sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,r} + \bar{z}^2_{kl}}\,) / (\bar{y}_{kl,l} + \sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,l} + \bar{z}^2_{kl}}\,)\ ]
\end{aligned}$$

$$- sign(\bar{x}_{kl})\ sign(\bar{z}_{kl})\ cos(\theta_{ij} - \theta_{kl})\ [tan^{-1}(\ \bar{y}_{kl,r}\sqrt{\bar{x}_{kl}^2/[\bar{z}_{kl}^2(\bar{x}_{kl}^2 + \bar{y}_{kl,r}^2 + \bar{z}_{kl}^2)]}\ )$$

$$- tan^{-1}(\ \bar{y}_{kl,l}\sqrt{\bar{x}_{kl}^2/[\bar{z}_{kl}^2(\bar{x}_{kl}^2 + \bar{y}_{kl,l}^2 + \bar{z}_{kl}^2)]}\ )\ \Big] +\ sin(\theta_{ij} - \theta_{kl})$$

$$\ln[\ (\ \bar{y}_{kl,r} + \sqrt{\bar{x}_{kl}^2 + \bar{y}_{kl,r}^2 + \bar{z}_{kl}^2}\ )/(\ \bar{y}_{kl,l} + \sqrt{\bar{x}_{kl}^2 + \bar{y}_{kl,l}^2 + \bar{z}_{kl}^2}\ )\ ] \tag{3.54}$$

and for $\bar{z}_{kl} = 0$,

$$\mathbf{I}_{kl,ij}^n = -sin(\theta_{ij} - \theta_{kl}) \ln\Big[$$

$$(\ \bar{y}_{kl,r} + \sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}_{kl,r}^2}\ )/(\ \bar{y}_{kl,l} + \sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}_{kl,l}^2}\ )\ \Big]$$

$$+\ sin(\theta_{ij} - \theta_{kl}) \ln\Big[\ (\ \bar{y}_{kl,r} + \sqrt{\bar{x}_{kl}^2 + \bar{y}_{kl,r}^2}\ )/(\ \bar{y}_{kl,l} + \sqrt{\bar{x}_{kl}^2 + \bar{y}_{kl,l}^2}\ )\ \Big] \tag{3.55}$$

where

$$\bar{y}_{kl,r} = \bar{y}_{kl} - \bar{t}_{ij,r}$$

$$\bar{y}_{kl,l} = \bar{y}_{kl} - \bar{t}_{ij,l}$$

and the subscripts r and l stand for the integration limits in the y direction on each panel. The symbol *sign* is defined as follows:

$$a = sign(a)\sqrt{a^2}$$

Similarly, the normal boundary condition is applied to $\vec{\mathbf{u}}_2$ such that

$$\begin{aligned}\vec{\mathbf{u}}_2 \cdot \vec{\mathbf{n}}_{kl} &= \sum_j \sum_i\ \nabla_{kl} \times (\vec{\psi}) \cdot \vec{\mathbf{n}}_{kl} \\ &= \sum_j \sum_i \left(\mathbf{D}_{kl,ij}^n\right) \left[\frac{\gamma_{ij}}{4\pi}\right] + \nabla \times \vec{\mathbf{R}} \cdot \vec{\mathbf{n}}_{kl}\end{aligned} \tag{3.56}$$

that is,

$$\left(\mathbf{D}_{kl,ij}^n\right) \left[\frac{\gamma_{ij}}{4\pi}\right] = -\vec{\mathbf{V}}_{0_n} \cdot \vec{\mathbf{n}}_{kl} - \left(\nabla \times \vec{\mathbf{R}}\right) \cdot \vec{\mathbf{n}}_{kl} \tag{3.57}$$

The influence matrix $\mathbf{D}^n_{kl,ij}$ for $\bar{z}_{kl} \neq 0$, can be expressed as

$$
\begin{aligned}
\mathbf{D}^n_{kl,ij} = \; & sign(\bar{y}_{kl,l})\; sign(\bar{z}_{kl})\; sin(\theta_{ij} - \theta_{kl})\; [\\
& tan^{-1}(\,(\bar{x}_{kl} - \bar{s}_{ij})\sqrt{(\bar{y}_{kl,l})^2/(\bar{z}^2_{kl}[(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,l} + \bar{z}^2_{kl}])}\,) \\
& - tan^{-1}(\,\bar{x}_{kl}\sqrt{(\bar{y}_{kl,l})^2/(\bar{z}^2_{kl}[\bar{x}^2_{kl} + \bar{y}^2_{kl,l} + \bar{z}^2_{kl}])}\,)\,] \\
& - sign(\bar{y}_{kl,l})\; cos(\theta_{ij} - \theta_{kl})/2\{ \\
& \ln[\,(\,\sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,l} + \bar{z}^2_{kl}} - \sqrt{\bar{y}^2_{kl,l}}\,)/(\,\sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,l} + \bar{z}^2_{kl}} + \sqrt{\bar{y}^2_{kl,l}}\,)\,] \\
& - \ln[\,(\,\sqrt{\bar{x}^2_{kl} + \bar{y}^2_{kl,l} + \bar{z}^2_{kl}} - \sqrt{\bar{y}^2_{kl,l}}\,)/(\,\sqrt{\bar{x}^2_{kl} + \bar{y}^2_{kl,l} + \bar{z}^2_{kl}} + \sqrt{\bar{y}^2_{kl,l}}\,)\,]\} \\
& + sign(\bar{y}_{kl,r})\; sign(\bar{z}_{kl})\; sin(\theta_{ij} - \theta_{kl})[\\
& tan^{-1}(\,(\bar{x}_{kl} - \bar{s}_{ij})\sqrt{(\bar{y}_{kl,r})^2/(\bar{z}^2_{kl}[(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,r} + \bar{z}^2_{kl}])}\,) \\
& - tan^{-1}(\,\bar{x}_{kl}\sqrt{(\bar{y}_{kl,r})^2/(\bar{z}^2_{kl}[\bar{x}^2_{kl} + \bar{y}^2_{kl,r} - \bar{z}^2_{kl}])}\,)\,] \\
& - sign(\bar{y}_{kl,r})\; cos(\theta_{ij} - \theta_{kl})/2.0\{ \\
& \ln[\,(\,\sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,r} + \bar{z}^2_{kl}} - \sqrt{\bar{y}^2_{kl,r}}\,)/(\,\sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,r} + \bar{z}^2_{kl}} + \sqrt{\bar{y}^2_{kl,r}}\,)\,] \\
& - \ln[\,(\,\sqrt{\bar{x}^2_{kl} + \bar{y}^2_{kl,r} + \bar{z}^2_{kl}} - \sqrt{\bar{y}^2_{kl,r}}\,)/(\,\sqrt{\bar{x}^2_{kl} + \bar{y}^2_{kl,r} + \bar{z}^2_{kl}} + \sqrt{\bar{y}^2_{kl,r}}\,)\,]\}
\end{aligned}
\tag{3.58}
$$

and for $\bar{z}_{kl} = 0$,

$$
\begin{aligned}
\mathbf{D}^n_{kl,ij} = & -cos(\theta_{ij} - \theta_{kl}) \ln\Big[\\
& (\,\bar{y}_{kl,l} + \sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,l}}\,)/(\,\bar{y}_{kl,r} + \sqrt{(\bar{x}_{kl} - \bar{s}_{ij})^2 + \bar{y}^2_{kl,r}}\,)\Big] \\
& + \; cos(\theta_{ij} - \theta_{kl}) \ln\Big[\,(\,\bar{y}_{kl,l} + \sqrt{\bar{x}^2_{kl} + \bar{y}^2_{kl,l}}\,)/(\,\bar{y}_{kl,r} + \sqrt{\bar{x}^2_{kl} + \bar{y}^2_{kl,r}}\,)\Big]
\end{aligned}
\tag{3.59}
$$

The discrete expressions for $-\nabla \times \mathbf{R} \cdot \vec{\mathbf{n}}_{kl}$ in Equation (3.57) is as follows:

$$-\nabla \times \vec{\mathbf{R}} \cdot \vec{\mathbf{n}}_{kl} = \frac{1}{4\pi} \int\int_R\int \frac{-[(y_k - \tilde{y})\zeta_z' - (z_k - \tilde{z})\zeta_y']sin(\theta_k) + [(x_k - \tilde{x})\zeta_y' - (y_k - \tilde{y})\zeta_x']cos(\theta_k)}{[(x_k - \tilde{x})^2 + (y_k - \tilde{y})^2 + (z_k - \tilde{z})^2]^{3/2}} d\tilde{x}d\tilde{y}d\tilde{z} \tag{3.60}$$

which is integrated numerically.

For inviscid and incompressible flow, the vorticity transport equation becomes

$$\frac{D\vec{\zeta}}{Dt} = \vec{\zeta} \cdot \nabla\vec{\mathbf{V}} \tag{3.61}$$

If the flow is two-dimensional, the right-hand-side of Equation (3.61) is zero, which means that the vorticity is just advected by the flow with no vortex stretching. The vortex stretching is not zero, for three-dimensional flow, but that case is not considered herein. The implementation of the vorticity transport equation (without vortex stretching) into the iteration technique is similar to the approach of Chow *et al.* [18]. Since there is no vortex stretching, vorticity is conserved along each stream line. After updating the velocity field from Equation 3.1, the vorticity field is updated by tracing the stream function on each grid point to its far upstream location. The onset vorticity, $\vec{\zeta}_o$, at that point is then used to update $\zeta' = \zeta - \zeta_o$. This iteration technique usually converges in less than 10 iterations.

CHAPTER IV

NUMERICAL ALGORITHM

In this section the computer program for solving the scalar and vector potentials, or source and vortex strengths for a two-dimensional airfoil, is described. The program uses an iteration procedure to solve for σ and γ in two-dimensional nonuniform shear flow. The area integral due to the onset vorticity field in Equation (3.31) is determined by integrating numerically with each iteration. The iteration first assumes $\vec{\zeta} = \vec{\zeta}_o$ and calculates a set of piecewise vortex values γ from Equation (3.37), The velocity at all grid point is then calculated from Equations (3.30′) and (3.32′). The stream function for each grid point is then determined from

$$\psi(x,y) - \psi_{ref} = \int_{y_{ref}}^{y} u(x,\tilde{y})\, d\tilde{y} - \int_{x_{ref}}^{x} v(\tilde{x},y)\, d\tilde{x} \tag{4.1}$$

where x_{ref} and y_{ref} are the coordinates at which the stream function has the value ψ_{ref}. The computation next updates the vorticity field by assigning to the grid point the upstream value of $\zeta = \zeta_o + \zeta'$ which corresponds to the grid point value of ψ and recalculates a set of values of γ for the new vortex distribution. The iteration is stopped when the vortex distribution has converged in the selected norm (in this case for $Max(\Delta\gamma) \leq \pm 10^{-4}$).

An outline of the computational procedures for the two-dimensional airfoil is presented here. The computer program is composed of seven major sub-programs, i.e. FORCE, FIELD, GEOMTY, JAWS, LEQN, MAIN, and ONSET. A brief description of these sub-programs is as follows:

FORCE – This sub-program is for calculating the aerodynamic coefficients, such as C_p, C_l, and C_m.

FIELD – This sub-program is for calculating the velocity and vorticity fields.

GEOMTY – This sub-program is for determining the airfoil shape and grid size in the domain.

JAWS – This sub-program is specifically for calculating the necessary information from JAWS wind shear data.

LEQN – This sub-program is the core of the computation, which calculates the coefficient matrices for the normal boundary condition and the singularity strengths.

MAIN – This sub-program is the main program for initializing the computation and for connecting all the sub-programs.

ONSET – This sub-program is for determining the onset velocity field.

In the following paragraphs, the introduction of each sub-program will be based on the order of the computational procedures. First, the MAIN program initializes the control parameters to begin the calculation. The parameters are input by NAMELIST in AIRFOIL.DAT. An example file for a given test case of the input and output formats is included in Appendix E. As an example of the control parameters, a specific head wind can be selected with the following parameter IFLOW :

IFLOW = 1,	uniform flow
2,	linear shear flow
3,	Gaussian shear flow
4,	JAWS shear flow

and with or without different downdrafts

IDBL = 0,	without downdraft
1,	uniform downdraft
2,	linear shear downdraft
3,	Gaussian shear downdraft

	4,	Cosine-type shear downdraft
	5,	Sinousoidal type shear downdraft

The parameter IAIR is used for selecting different airfoil shape

IAIR =	1,	NACA 0012
	2,	NACA 2412
	3,	NACA 4-digits
	4,	symmetric Joukowski
	5,	cambered Joukowski

After initialization, the MAIN program calls GEOMTY to determine the geometry of the airfoil and grid size. More details about the grid size which is computed by the subroutine GRDCORD is given in the following paragraphs.

In the sub-program GEOMTY, the coordinates for the airfoil shape are calculated first. There are two subroutines, NACA4 and JOUKAR, for calculating NACA 4-digit and Joukowski airfoils respectively. The formula for the NACA 4-digit airfoil are as follows [53]:

$$\begin{aligned} \pm z_t = \frac{t}{0.20}(0.29690\ \sqrt{x} - 0.12600\ x - 0.35160\ x^2 \\ + 0.28430\ x^3 - 0.10150\ x^4) \end{aligned} \tag{4.2}$$

for constructing the thickness distribution, z_t, and

$$\begin{aligned} z_c = \frac{m}{p^2}(2p\ x - x^2)\ \ (forward\ of\ max.\ ordinate) \\ \frac{m}{(1-p)^2}[(1-2p) - 2px - x^2]\ \ (aft\ of\ max.\ ordinate) \end{aligned} \tag{4.3}$$

for constructing the camber distribution, z_c, where t stands for the maximum thickness of the airfoil, m for the maximum ordinate of the mean line, and p for the chordwise position of the maximum ordinate. Values of x, z_t, and z_c from the above equations are substituted into Equations (4.4a) and (4.4b), for calculating the upper and lower coordinates of the NACA 4-digit airfoil.

$$x_u = x - z_t\, sin(\delta)$$
$$z_u = z_c + z_t\, cos(\delta) \tag{4.4a}$$

$$x_l = x + z_t\, sin(\delta)$$
$$z_l = z_c - z_t\, cos(\delta) \tag{4.4b}$$

The definitions of x_u, z_u, x_l, z_l, and δ can be found in Figure 4.1. Similar procedures are used for generating the coordinates of the Joukowski airfoils except the classical Joukowski transformation is used instead of a standard formula. The panelling technique used in the program for generating both NACA 4-digit and Joukowski airfoils is as follows:

$$x = 0.5 \times (1.0 + cos(\theta)) \tag{4.5}$$

where $-\pi \leq \theta \leq \pi$ and x is the coordinate for determining the z coordinate in Equation (4.2) and (4.3). Both x and z are the coordinates for the airfoil (see Figure 4.1). The range $-\pi \leq \theta \leq 0$ is for the lower surface of the airfoil and the range $0 \leq \theta \leq \pi$ is for the upper surface. The increments of θ is just $\pi/(NPAN/2)$ where the parameter NPAN stands for the total number of panels.

After the coordinates of the airfoil are determined, subroutines BDYCORD, PANEL and CNTCORD are used to calculate the control point, panel length and panel orientation. The control point is a position on each panel for imposing the no penetration boundary condition and for calculating the velocity (or pressure) value. The location of the control point is not necessarily situated in the middle of each panel because this will cause the diagonal elements of the vortex coefficient

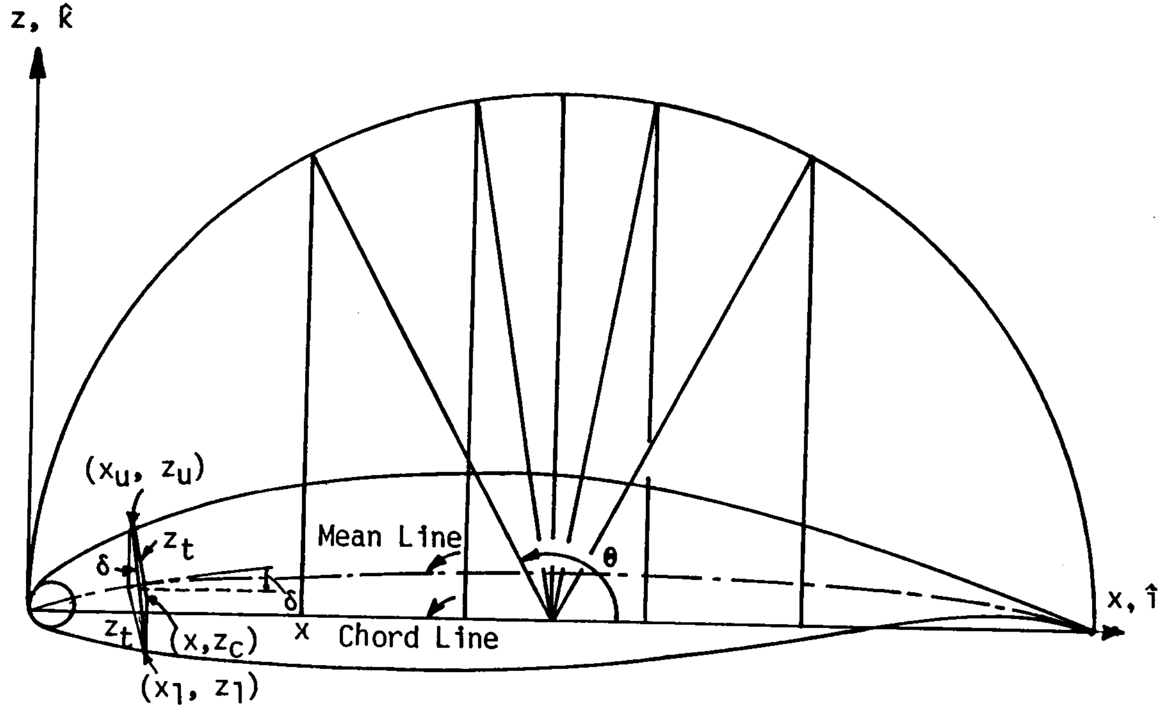


Figure 4.1. Illustration of the panelling scheme and airfoil nomenclature.

matrix to become zero (for more discussion see Appendix B). The parameter RCP is designed to control the location of the control point, which is also input by NAMELIST. The range $0.55l \leq RCP \leq 0.75l$ is recommended for calculation and $RCP = 0.60l$ is used in the present study, where l is the length of the panel.

Subroutine GRDCORD is for calculating the coordinates of the grids in the computational domain. In the present algorithm the computational domain is taken a 10×10 chord length square region and a uniform rectangular mesh is used. The number of grid points used have been 21×21, 41×41 and 51×51 (see Figure 4.2). The grid points are used only in the iteration for nonuniform shear flow. For the 10×10 chord length computational domain, the difference in computed values of C_l for a grid number of 21×21 and for 41×41 is less than 1.5%. However, the grid number of 41×41 has been the primary grid size used in the present study.

Subroutines TRACORD and TRANXY are used for determining the transformed coordinates with respect to local coordinates. These local coordinates are substituted into the Equations (3.40) and (3.43) for calculating the coefficient matrix $\mathbf{I}_{ij}^n$ and $\mathbf{D}_{ij}^n$. The formula for the transformation is:

$$\bar{x}_i = cos(\theta_i)\ (x - x_{io}) + sin(\theta_i)\ (z - z_{io})$$

$$\bar{z}_i = -sin(\theta_i)\ (x - x_{io}) + cos(\theta_i)\ (z - z_{io}) \tag{4.6}$$

where θ is the panel orientation angle and x_{io} and z_{io} are the origin of each local coordinate (see Figure 3.9 page 64).

After all the geometrical quantities have been determined, the onset velocity is calculated based on the control parameter IFLOW the value of which selects various subroutines such as:

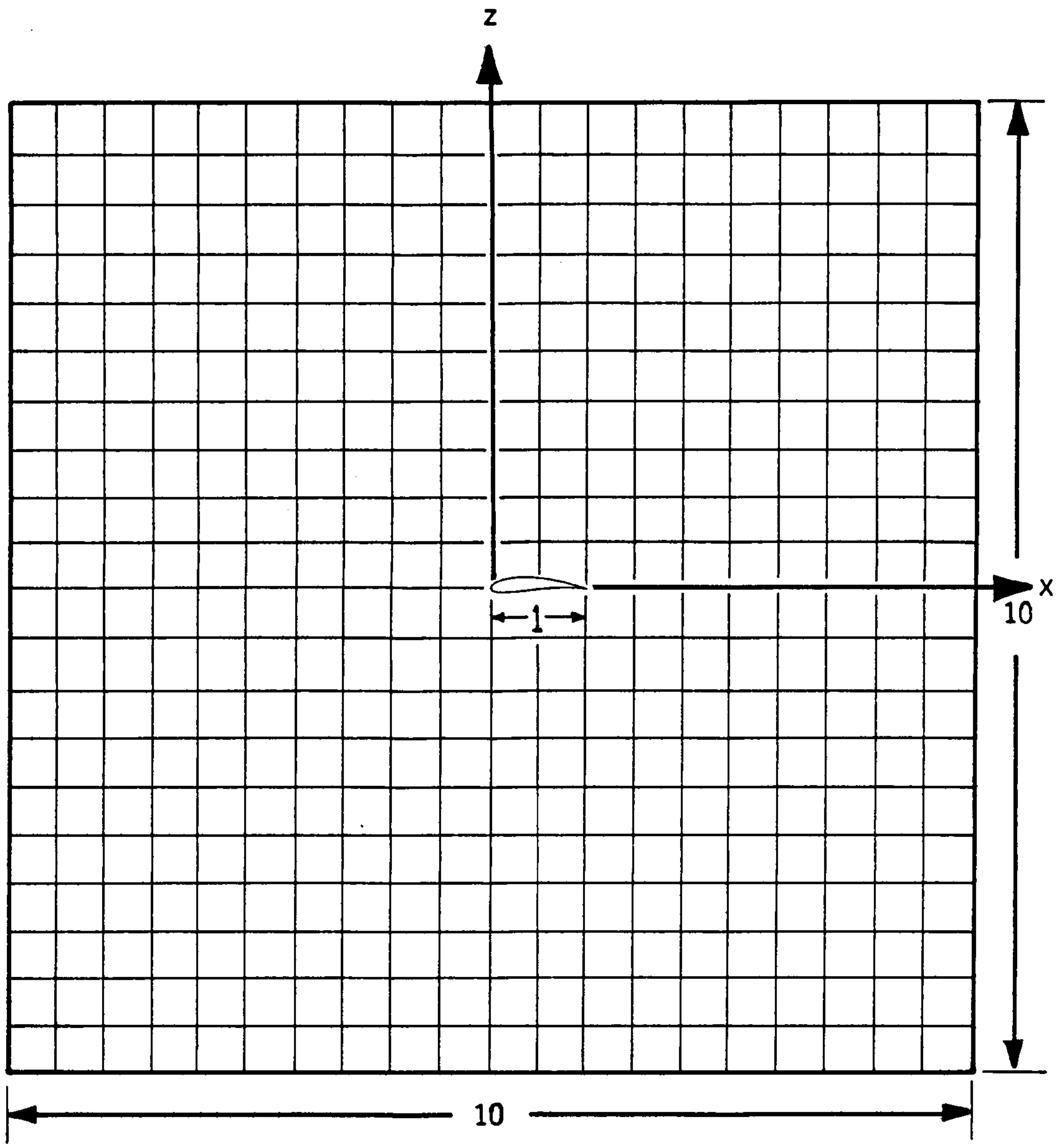


Figure 4.2. Computational domain for iterating the forcing term.

IFLOW = 1,	call subroutine UNISFL
2,	call subroutine UNISFL
3,	call subroutine EXPSFL
4,	call subroutine JAWSFL

Again the parameter IFLOW is input by NAMELIST in AIRFOIL.DAT.

The sub-program JAWS is special for the JAWS wind shear data set. The JAWS wind shear data set provides three wind speed components: longitudinal, latitudinal, and vertical, at each grid point, which are on a $500ft \times 500ft$ spacing. For two-dimensional calculations, a corridor (or vertical plane through the microburst flow field) of grid point wind speeds from the JAWS data, 5AU1847AB. COR [5], is used. The wind velocity at a specific point is calculated from each grid point of this corridor by area-weighted interpolation.

$$W_p = \frac{\sum_i W_i \times A_i}{\sum_j A_j}, \qquad (i, j = 1, 4) \tag{4.7}$$

where W_p is the wind velocity at any point p of this corridor and W_i is the input wind velocity at the four corners of each computational domain which is 50×50 referenced to the airfoil chord length. Because the JAWS wind field is taken $500ft \times 500ft$, the airfoil chord length is fixed to $10ft$ long. The areas A_1 to A_4 are the areas diagonal opposite to the corner points 1 to 4.

In sub-program LEQN, the subroutine COEF calculates the coefficient matrices for the source and the vortex. Equations (3.40) and (3.43) are the expressions for the coefficient matrices of the normal components of perturbed velocities. Equation (3.48) shows the relationship between the coefficient matrices of the normal and the tangential components of perturbed velocities. Algorithms for calculating these expressions are programmed in the subroutine COEF.

The Kutta condition is based on Criterion (2) page 26, which requires the surface pressures (velocity magnitudes) on the upper and lower surfaces to have a common value. The Kutta condition is imposed by replacing the last row of the normal coefficient matrix, $\mathbf{D}^n_{ij}$, for the vortex by the following expression (see Figure 4.3).

$$\left[\vec{\mathbf{V}}_{0_1} + \sum_j \frac{\sigma_j}{2\pi}\int_j \nabla_1(\log(r_{1j}))\, dl_j - \sum_j \frac{\gamma_j}{2\pi}\int_j \nabla_1(\Theta_{1j})\, dl_j \right.$$

$$\left. -\nabla_1 \times \left(\frac{\hat{j}}{2\pi}\int_S\int \ln(r)(\zeta_y - \zeta_{yo})\, ds\right)\right] \cdot (-\vec{\mathbf{t}}_1)$$

$$= \left[\vec{\mathbf{V}}_{0_M} + \sum_j \frac{\sigma_j}{2\pi}\int_j \nabla_M(\log(r_{Mj}))\, dl_j - \sum_j \frac{\gamma_j}{2\pi}\int_j \nabla_M(\Theta_{Mj})\, dl_j \right.$$

$$\left. -\nabla_M \times \left(\frac{\hat{j}}{2\pi}\int_S\int \ln(r)(\zeta_y - \zeta_{yo})\, ds\right)\right] \cdot (\vec{\mathbf{t}}_M)$$

such that

$$\sum_j \left(\mathbf{D}^t_{Mj} + \mathbf{D}^t_{1j}\right) \frac{\gamma_j}{2\pi} =$$

$$-\left[\vec{\mathbf{V}}_{0_1} \cdot \vec{\mathbf{t}}_1 + \sum_j \mathbf{I}^t_{1j}\, \frac{\sigma_j}{2\pi} - \nabla_1 \times \left(\frac{\hat{j}}{2\pi}\int_S\int \ln(r)(\zeta_y - \zeta_{yo})\, ds\right)\cdot \vec{\mathbf{t}}_1\right.$$

$$\left.+\vec{\mathbf{V}}_{0_M} \cdot \vec{\mathbf{t}}_M + \sum_j \mathbf{I}^t_{Mj}\, \frac{\sigma_j}{2\pi} - \nabla_M \times \left(\frac{\hat{j}}{2\pi}\int_S\int \ln(r)(\zeta_y - \zeta_{yo})\, ds\right)\cdot \vec{\mathbf{t}}_M\right] \quad (4.8)$$

The subroutines RHS1 and RHS2 calculate $-\vec{\mathbf{V}}_{0_t} \cdot \vec{\mathbf{n}}_i$ and $-\vec{\mathbf{V}}_{0_n} \cdot \vec{\mathbf{n}}_i$ respectively. The subroutine RHS2 implements $\nabla_i \times [\hat{j}/(2\pi)\int_S\int \log(r)(\zeta_y - \zeta_{yo})\, ds] \cdot \vec{\mathbf{n}}_i$ for calculating nonlinear shear flow problems by calling sub-function RESL. The area integral is calculated by calling the bicubic spline quadrature DBCQDU from International Mathematical and Statistical Libraries (IMSL). The matrix inversions of Equations (3.36) and (3.37) are calculated by calling LINV2F from IMSL, which has the features: row equilibration, partial pivoting, and iterative improvement. Once the inverse matrices are obtained the singularity strengths σ

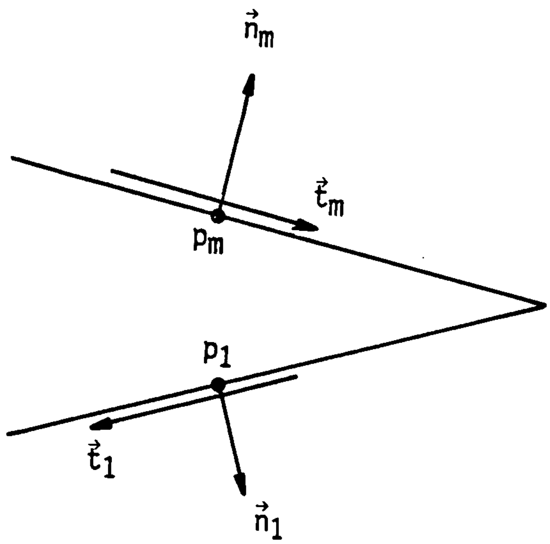


Figure 4.3. Two rearmost panels on the airfoil for imposing Kutta condition.

and γ can be calculated by multiplying the inverse matrices with $\mathbf{R}_{i1}(= -\vec{\mathbf{V}}_{0_t} \cdot \vec{\mathbf{n}}_i)$ and $\mathbf{R}_{i2}(= -\vec{\mathbf{V}}_{0_n} \cdot \vec{\mathbf{n}}_i + \nabla_i \times [\hat{j}/(2\pi) \int_S \int \log(r)(\zeta_y - \zeta_{yo})\ ds] \cdot \vec{\mathbf{n}}_i)$, with exception of the $\mathbf{R}_{M2}$ element which is given by the right-hand-side of Equation (4.8), respectively.

Based on the calculated singularity strengths σ and γ, the velocity and vorticity fields are updated in sub-program FIELD. First, the velocity field is updated in subroutine VELFLD by summing the onset velocity field and the perturbed velocity field as follows:

$$\vec{\mathbf{V}} = \vec{\mathbf{V}}_o + \mathbf{I}\,(\frac{\sigma}{2\pi}) + \mathbf{D}\,(\frac{\gamma}{2\pi}) \\ -\nabla \times \left(\frac{\hat{j}}{2\pi} \int_S \int \ln(r)(\zeta_y - \zeta_{yo})\ ds\right) \tag{4.9}$$

where $\mathbf{I}$ and $\mathbf{D}$ are the second order tensors (square matrices) and σ and γ are the column vectors of the strength of each panel.

In solving for γ in Equation (3.37), an iteration procedure is needed. First, it is assumed that $\vec{\zeta}^0$ equals $\vec{\zeta}_o$ which causes the surface integral (see Equation (3.35)) to be zero. The first iterative step is carried out and initial values of σ and γ are calculated. These values are then used to determine the velocity field and stream function in order to update the vorticity field in subroutine VORFLD. The vorticity field is updated by tracing the stream function far upstream in order to satisfy the vorticity transport equation. The present vorticity value at each grid point is determined by searching for the equivalent stream function value far upstream. The onset vorticity value with the equivalent stream function value is used as the present vorticity value. A similar approach is also used by Chow *et al* [18] and van der Vooren and Labrujere [20].

The onset vorticity ζ_o is calculated from $\zeta_o = \partial v/\partial x - \partial u/\partial y$. Fourth order accurate finite differencing schemes [53] are applied to calculate the vorticity ζ_o, for example,

$$\partial f_i = \frac{1}{12h}(f_{i-2} - 8\ f_{i-1} + 8\ f_{i+1} - f_{i+2}) \tag{4.10a}$$

for central differencing and

$$\partial f_i = \frac{1}{12h}(-25\ f_i + 48\ f_{i+1} - 36\ f_{i+2} + 16\ f_{i+3} - 3\ f_{i+4}) \tag{4.10b}$$

for one-sided differencing. The accuracy of the above finite difference techniques is h^4 where h is the grid size. Typically $0.2 < h < 0.5$ hence the accuracy is on the order of 5×10^{-3}.

Subroutines STREAM and STFN are designed for calculating the stagnation stream function and the stream function at each grid point respectively. The formula for calculating the stream function is given by Equation (4.1) (also see Figure 4.4), where x_{ref} and y_{ref} are the reference coordinate, and, ψ_{ref} equals zero in the lower left corner of the grid. The line integrals in Equation (4.1) are calculated by calling the cubic spline quadrature DCSQDU from the IMSL.

At last the criterion for checking convergence is $Max(\Delta\gamma)$ within $\pm 10^{-4}$. Originally a convergence criterion of $Max(\Delta\gamma) \leq \pm 10^{-6}$ was used for the calculation, however, the converged results were equal within at least 4 to 5 decimal for either criterion, therefore $Max(\Delta\gamma) \leq \pm 10^{-4}$ was used for the remainder of the study. Usually the iteration scheme converges in less than ten iterations.

Once the iteration converges, the sub-program FORCE is called by the MAIN program to calculate the aerodynamic coefficients C_p, C_l, and C_m. Again the numerical integration of the pressure distribution (from Bernoulli Equation, Equation (2.9)) along the airfoil surface is done by calling the cubic spline quadrature DCSQDU from the IMSL. Finally the subroutine OUTPUT outputs results to the output file AIRFOIL.OUT.

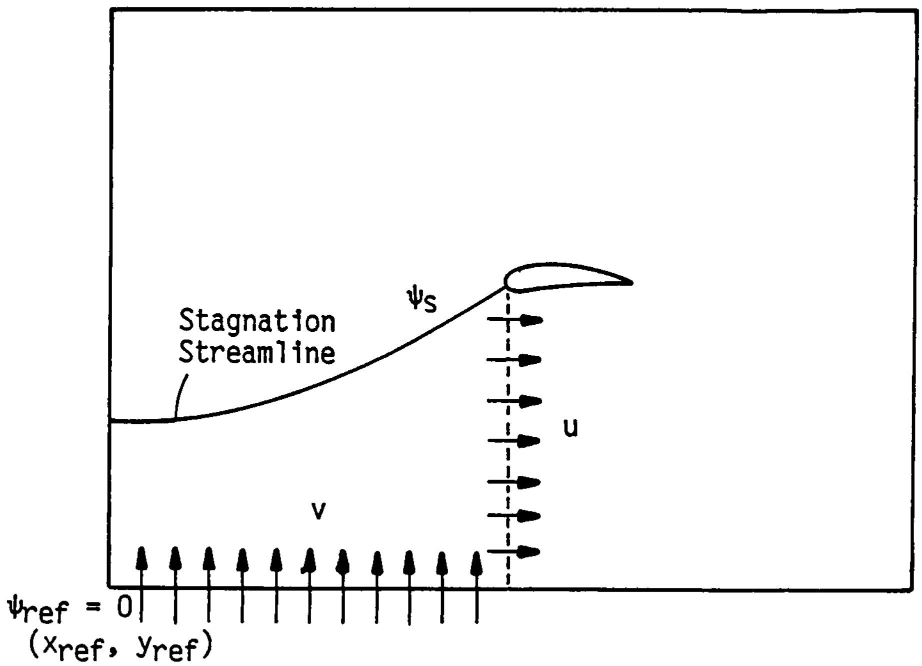


Figure 4.4. Illustration of integrating the stream function.

The calculation for nonpotential flow problems requires about 1.5×10^{-3} sec per iteration per grid point or panel on a VAX-11/785. The CPU times required for finite difference which solve formulations such as given by Chaderjian *et al.* [41] and Rao *et al.* [42], are 1×10^{-6} sec and 5×10^{-6} sec per iteration per grid point respectively on a CRAY XMP vector processor. It should be noted that because the VAX-11/785 is 100 times slower than the CRAY XMP, and because the present program is intended for method validation and has not been optimized, the calculations do not show any advantage in CPU time.

CHAPTER V

RESULTS AND DISCUSSION

In this chapter the calculated results using the present method are presented and compared with theoretical solutions such as two-dimensional uniform shear and two-dimensional nonuniform shear flows. Also comparisons with other numerical results based on either finite-difference or finite-element method are made. Final results from studies of downdraft and microburst wind shear effects on two-dimensional airfoils are presented.

A. TWO-DIMENSIONAL UNIFORM SHEAR FLOW

For a two-dimensional linear shear flow, the onset vorticity is constant everywhere inside the domain and there is no vortex stretching. Therefore the perturbed vorticity $\zeta'(= \zeta - \zeta_o)$ is identically zero such that the forcing term in Equation (3.9) is also zero. Thus, no iteration is necessary for two-dimensional linear shear flow. Tsien [10] attacked the linear shear flow problem depicted in Figure 5.1 analytically. Figures 5.2 and 5.3 compare Tsien's theoretical results with the present numerical calculations. Figure 5.2 shows the variation of lift with onset vorticity for a symmetric Joukowski airfoil at zero angle of attack. The computational model begins to departs from the theoretical model with increasing values of $\mid \zeta_o \mid$. According to thin airfoil theory, only camber and angle of attack cause lift in uniform flow. Since angle of attack and chamber are zero for the airfoil analyzed in Figure 5.2, the effect of the onset vorticity in nonuniform flow produces the induced lift.

Figure 5.2 shows that: for a symmetric airfoil at zero angle of attack, C_l is linearly proportional to the onset vorticity ζ_o. This relationship can be found from

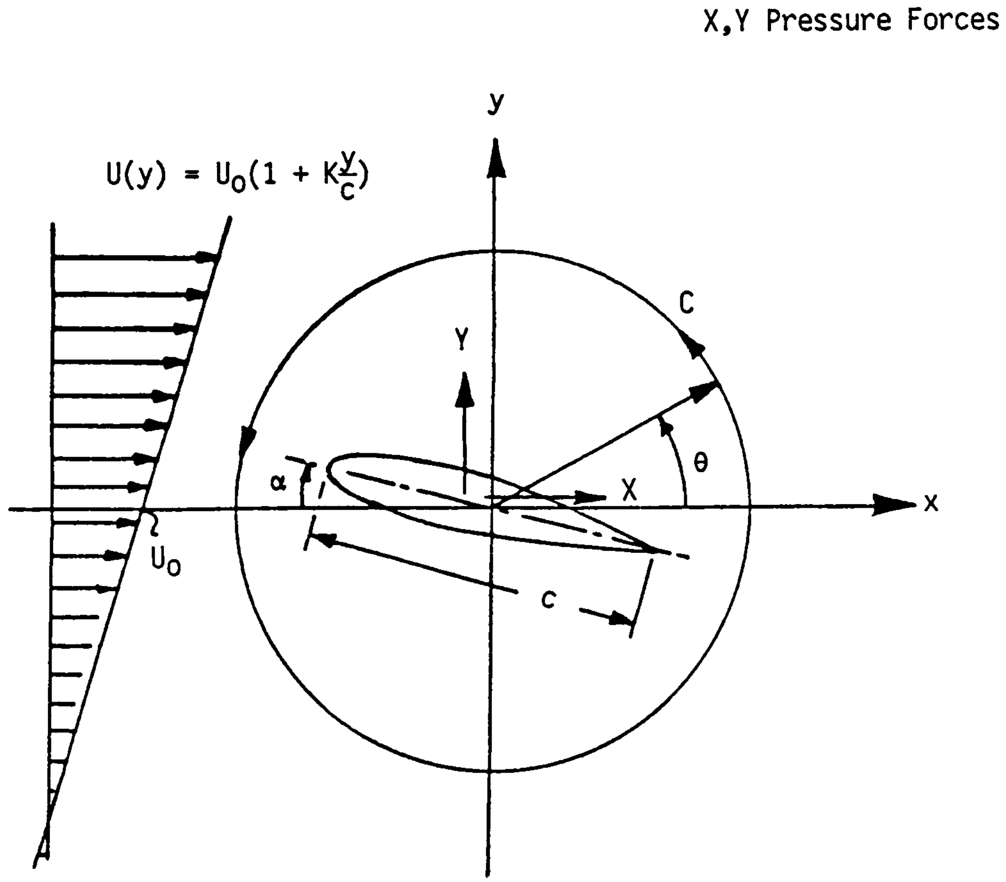


Figure 5.1. A symmetric Joukowski airfoil in stream with uniform shear.

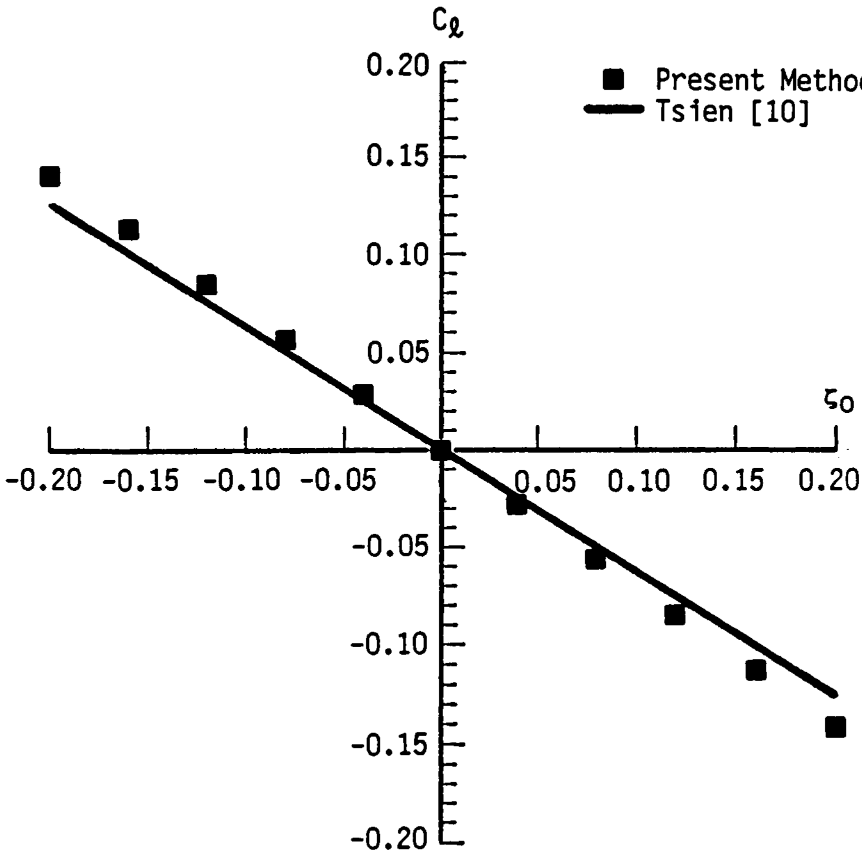


Figure 5.2. Lift vs onset vorticity for zero angle of attack, $\zeta_o = -K$.

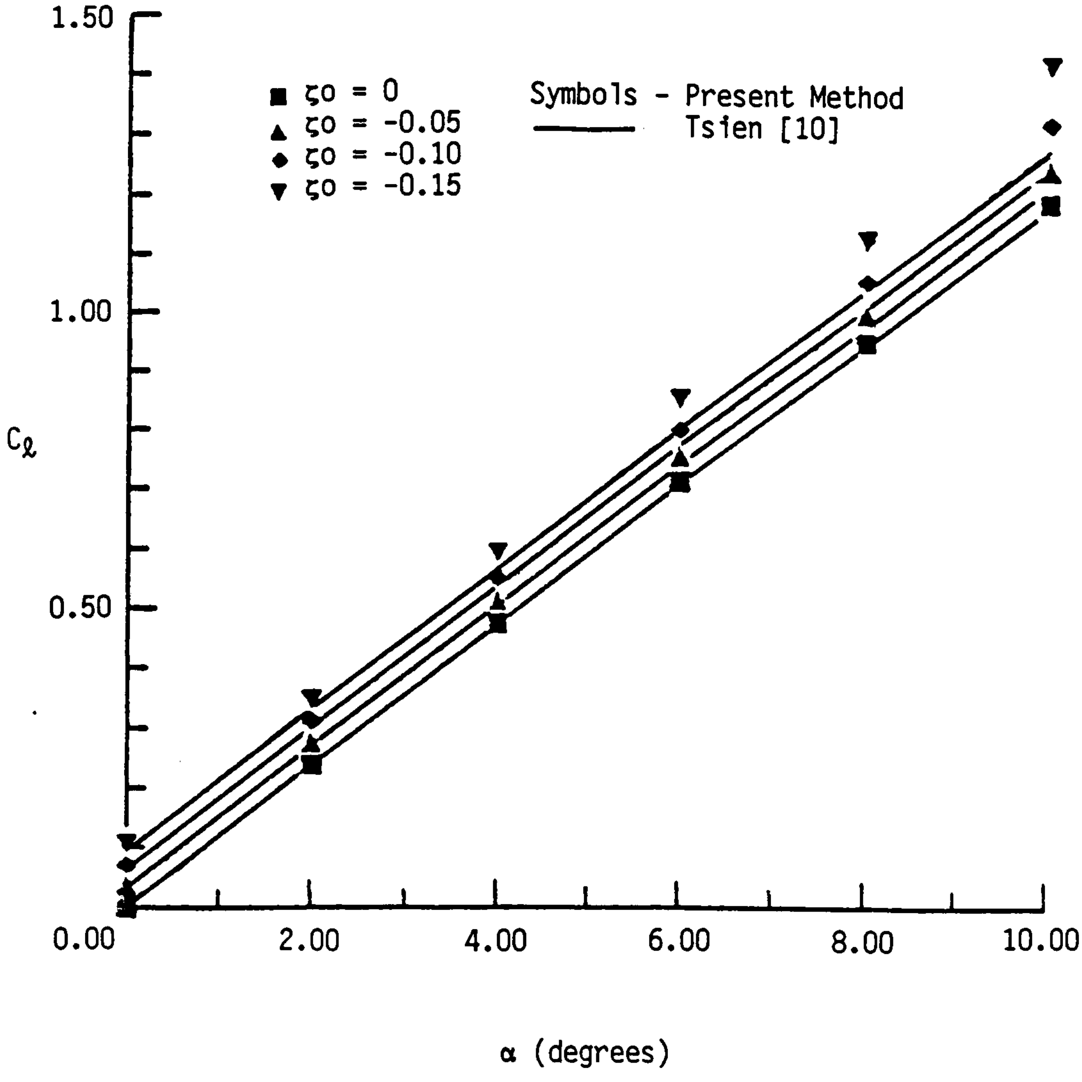


Figure 5.3. Lift vs angle of attack for various uniform shear, $\zeta_o = -K$.

Tsien's [10] Equation (60), i.e. :

$$C_l = 2\pi[l_o \ sin\alpha + K \ (l_1 + l_2 cos2\alpha) +$$
$$K^2 \ (l_3 sin\alpha + l_4 sin3\alpha)] \tag{5.1}$$

where l_o, l_1, l_2, l_3 and l_4 are the constants related to the geometry of the airfoil and $K = -\zeta_o$. At zero angle of attack ($\alpha = 0$) lift becomes linearly related to the shear, i.e. :

$$C_l = 2\pi \ K \ (l_1 + l_2) \tag{5.2}$$

This is consistent with two-dimensional uniform flow in that the Kutta-Joukowski theorem predicts

$$L = \rho U_o \Gamma \tag{5.3}$$

which indicates that lift is linearly proportional to the bound vortex circulation Γ. The present calculation also gives a linear variation but has a 12% steeper slope. The difference may be due to the numerical model for example, the panels at the trailing edge cannot simulate the cusped trailing edge as well as the theoretical model, that may induce large numerical error. Although the theoretical derivation shows a linear relationship there is presently no physical interpretation of this linear relationship between lift and onset vorticity.

Figure 5.3 shows a comparison of lift at various angles of attack and values of onset vorticity ζ_o predicted by Tsien's theory with the numerical results of the present method. Agreement with Tsien's theory is good for smaller onset vorticity. For larger onset vorticity the numerical results depart from the theory by as much as 13% at $\alpha = 10^o$ and $\zeta_o = -0.15$. Again, this error may be due to the numerical (panel) model itself. According to thin airfoil theory, lift is induced by airfoil camber and angle of attack in uniform flow. But for nonuniform flow such as linear shear flow, onset vorticity may induce extra lift if it is in a favorable sense, or vice versa (see Figure 5.2). The onset vorticity in the same direction as that of the bound vorticity is considered as favorable.

B. TWO-DIMENSIONAL NONUNIFORM SHEAR FLOW

Figure 5.4 depicts the flow problem of a parabolic onset velocity profile over a Joukowski airfoil studied by Jones [12,13]. For nonuniform shear flow, the linear relationship, Equation (5.2), mentioned in the last section does not hold. The forcing term in Equation (3.9) is not zero because neither the onset vorticity ζ_o nor the present vorticity ζ is constant everywhere inside the computational domain. Therefore an iteration is required for nonuniform shear flow. Figure 5.5 shows the lift coefficient versus the angle of attack with q as a parameter defining the magnitude of the nonuniform shear. The figure shows that, again, additional lift is gained because of the favorable onset vorticity. The results obtained by Jones [12,13] can only be considered as small perturbation results because the small perturbation assumption is applied to linearize the governing equation as mentioned in Chapter I. The present result agrees within 12% at $\alpha = 10^0$ with Jones' small perturbation result for the case considered.

Another example of the influence of shear flow is the interference effect of propellers on an airfoil. It is usually assumed that the disturbance of an airfoil on the jet stream behind the propeller is small. Chow *et al* [18] and van der Vooren and Labrujere [20] use finite-difference and finite element methods respectively to solve this problem. In order to compare the present method with Chow *et al*, van der Vooren and Labrujere's results, the scalar/vector potential panel method was applied to an airfoil submerged in a free stream with a Gaussian onset velocity profile as shown in Figure 5.6.

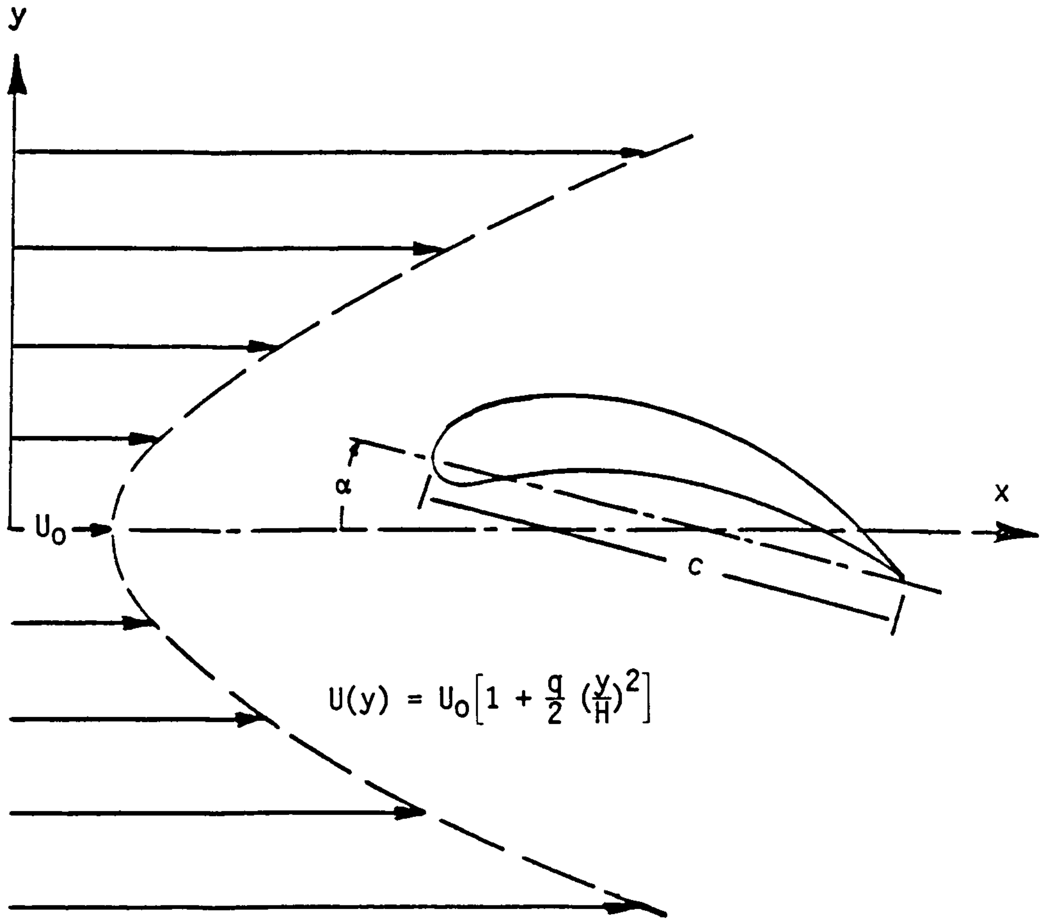


Figure 5.4. Theoretical model for a cambered Joukowski airfoil in stream with nonuniform shear [55].

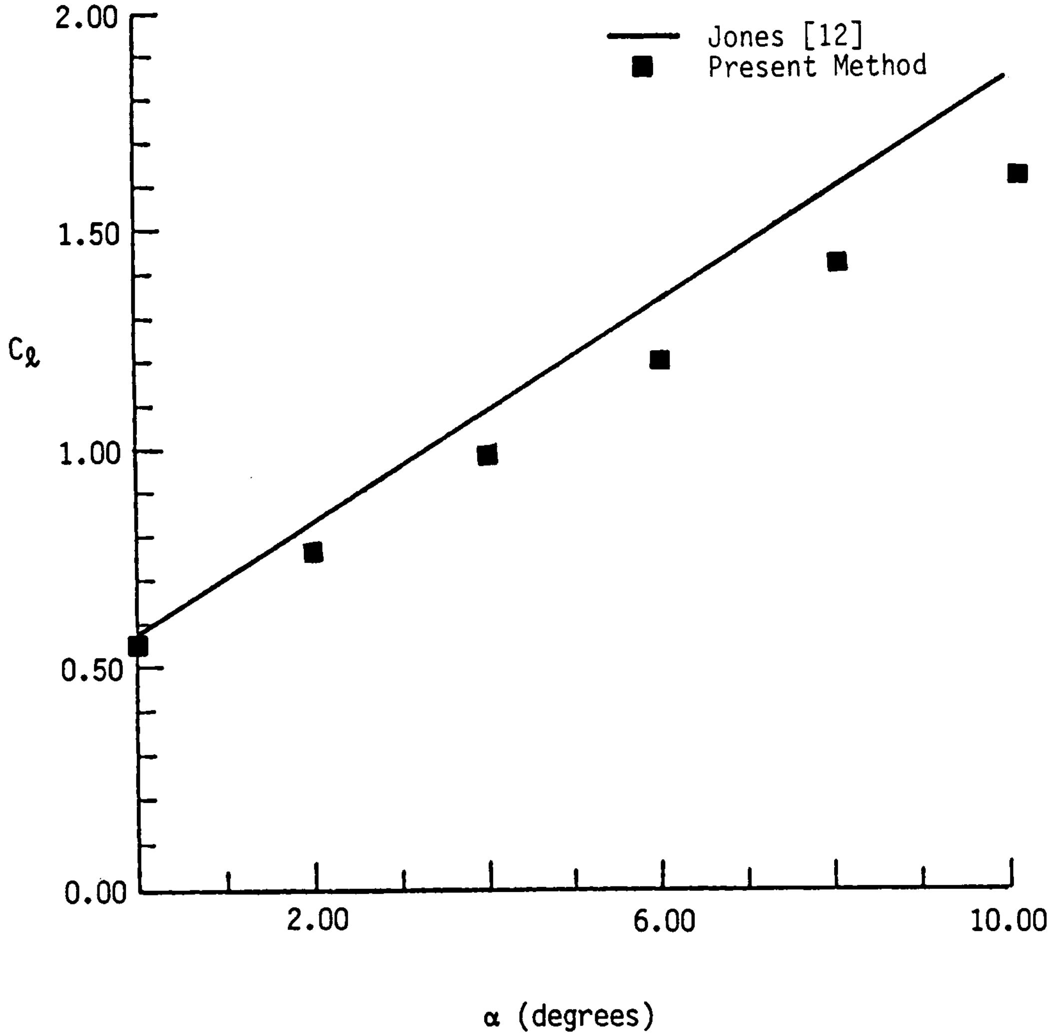


Figure 5.5. Lift vs angle of attack for nonuniform shear with $q = 0.02$, the parameter of parabolic nonuniform flow (100 panels on a cambered Joukowski airfoil).

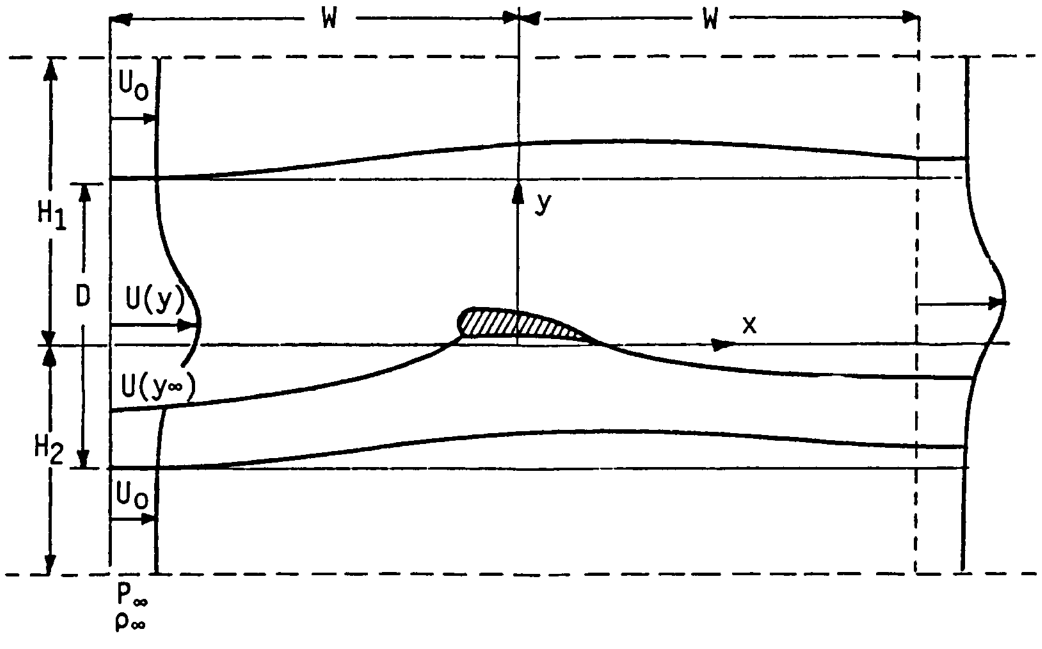


Figure 5.6. A cambered Joukowski airfoil in a stream with a Gaussian upstream wind profile, [18].

Figure 4.2 page 80 depicts the computational domain for iteration of the volume integral (area integral in two-dimensional flow). The present algorithm utilizes a rectangular mesh for the iteration and chord length as the reference length for normalizing the grid. This square computational domain (10×10) is wider but not as long as the strip-type computational domain (36×6.4) used by Chow *et al* and van der Vooren. For uniform flow, an analytical solution can be obtained and is used to justify the present result and van der Vooren's result. Figure 5.7 compares the present calculation with the analytical solution and finite element solution respectively [20]. Van der Vooren's calculation tends to overestimated the analytical result, whereas, the present calculation is very close to the analytical solution. The present method predicts C_l within 0.4% of the exact solution while the finite element method is only within 4.3%. The overestimation may be due to fictitous "wall effect" caused by the slender computational domain used in the finite element method. Figures 5.8 - 5.10 compare the scalar/vector panel method results with Chow *et al* and van der Vooren's results for different magnitudes of Gaussian shear flow. The general form of the velocity profile is $U(y) = 1+a\ exp(-y^2/d^2)$ where the parameter a represents the difference between the maximum velocity and the constant value i.e. $U_o(y = H) = 1$, and the parameter d represents the spread of the velocity nonuniformity. The C_l values of the present, Chow *et al*, van der Vooren and Labrujere's results are included in the figures. The fictitious "wall effect" observed in Figure 5.7 is not very clear in Figures 5.8, 5.9 and 5.10 which are for various Gaussian shear flows. Because the C_l values from the present computation are now very close to, from 0.5% to 2.61%, Chow *et al* and van der Vooren's results. The C_l values of Figures 5.9 and 5.10 are lower than their results, but C_l value of Figure 5.8 is slight higher (2%).

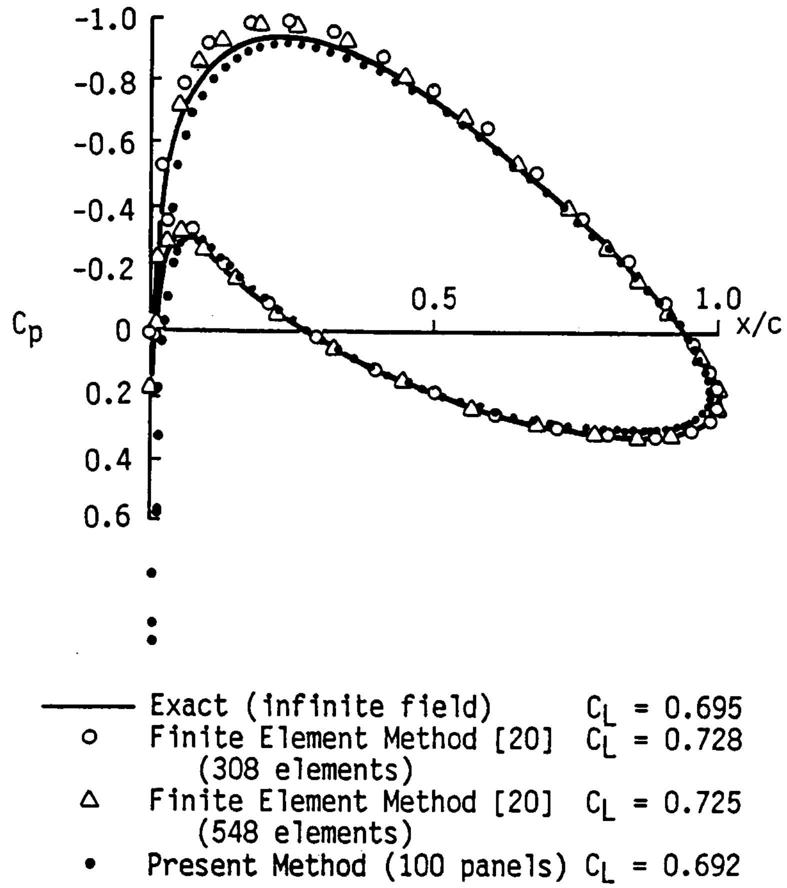


Figure 5.7. Pressure distribution of an airfoil in uniform flow, $U(y) = U_o$.

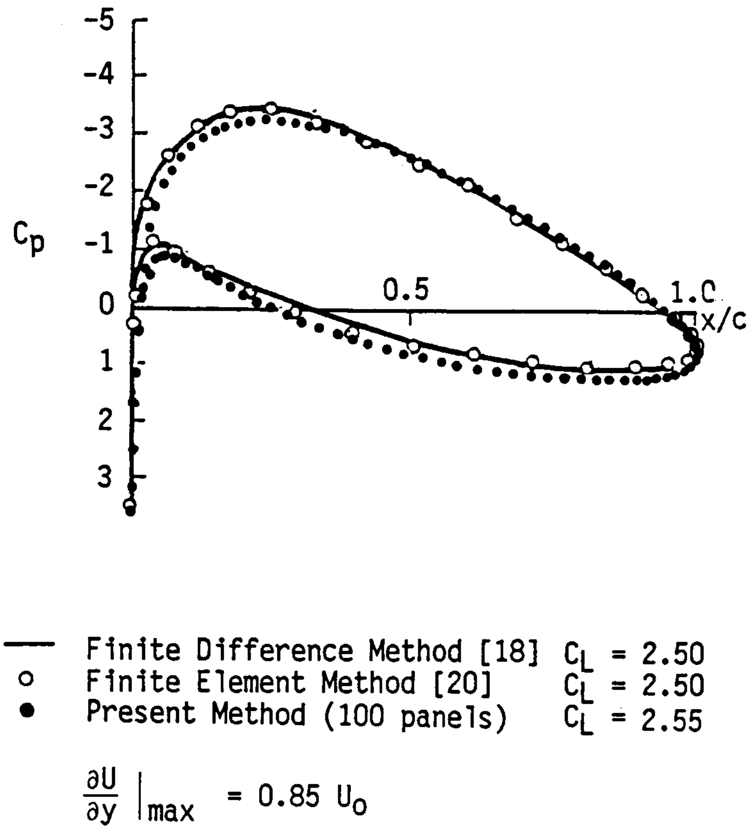
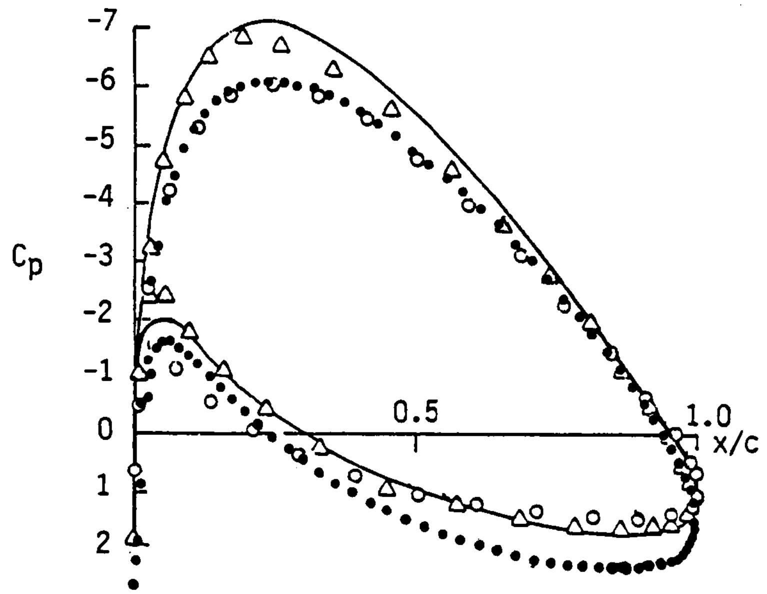


Figure 5.8. Pressure distribution of an airfoil in nonuniform stream, $U(y) = U_o(1 + e^{-y^2})$, U_o reference velocity.



— Finite Difference Method [18] $C_L = 4.98$
△○ Finite Element Method [20] $C_L = 4.68, 4.35$
• Panel Method (100 panels) $C_L = 4.85$

$$\left.\frac{\partial U}{\partial y}\right|_{max} = 1.72\ U_0$$

Figure 5.9. Pressure distribution of an airfoil in nonuniform stream, $U(y) = U_o(1 + 2e^{-y^2})$.

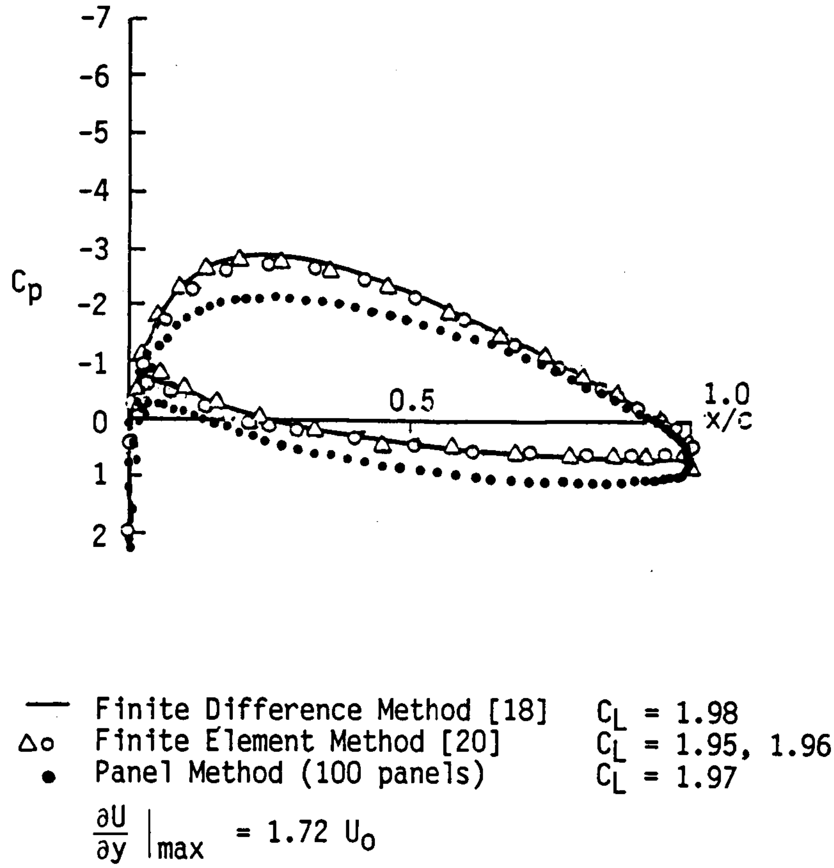


Figure 5.10. Pressure distribution of an airfoil in nonuniform stream, $U(y) = U_o(1 + e^{-(y/0.5)^2})$.

Experimental or other numerical results are needed to justify the present results and Chow *et al* and van der Vooren's results.

Figure 5.11 shows the pressure distribution for three different onset flows, uniform, linear, and nonlinear. In comparing the effects of different shear profiles on the lift a common velocity reference values is difficult to define. It is argues herein that U_o should be the mean velocity which corresponds to the equivalent momentum in each profile i.e.

$$U_o = \left[\int_{-L/2}^{L/2} U^2(y)dy/L \right]^{1/2}$$

It is observed that the C_p distributions in Figure 5.11 for the three different onset flows are about the same if the dynamic pressure $\rho U(y_\infty)^2/2$ is used to nondimensionalize the pressure distribution. Based on the dynamic pressure $\rho U(y_\infty)^2/2$, the C_l values for uniform, linear and nonlinear shear flows are 0.6921, 0.7374 and 0.7290 respectively, which are also very close values. But, the pressure forces for the three different onset flows are quite different because $U(y_\infty)$ values for uniform, linear and nonlinear shear flows are 6.0, 5.26 and 13.17 respectively. That means the nonlinear onset flow induces the highest lift and uniform has the lowest lift because $L = C_l \ \rho U^2(y_\infty) \ c/2$.

A similar analysis was carried-out which compared shear profiles on an equivalent mass bases, i.e.

$$U_o = \int_{-L/2}^{L/2} U(y)dy/L$$

The result is shown in Figure 5.12. The values of C_l for the uniform, linear shear and exponential shear in this case are 0.6921, 0.7317 and 0.7420 (if based on $\rho U(y_\infty)^2/2$), respectively. Again, the exponential shear flow with the strongest onset vorticity has the highest lift.

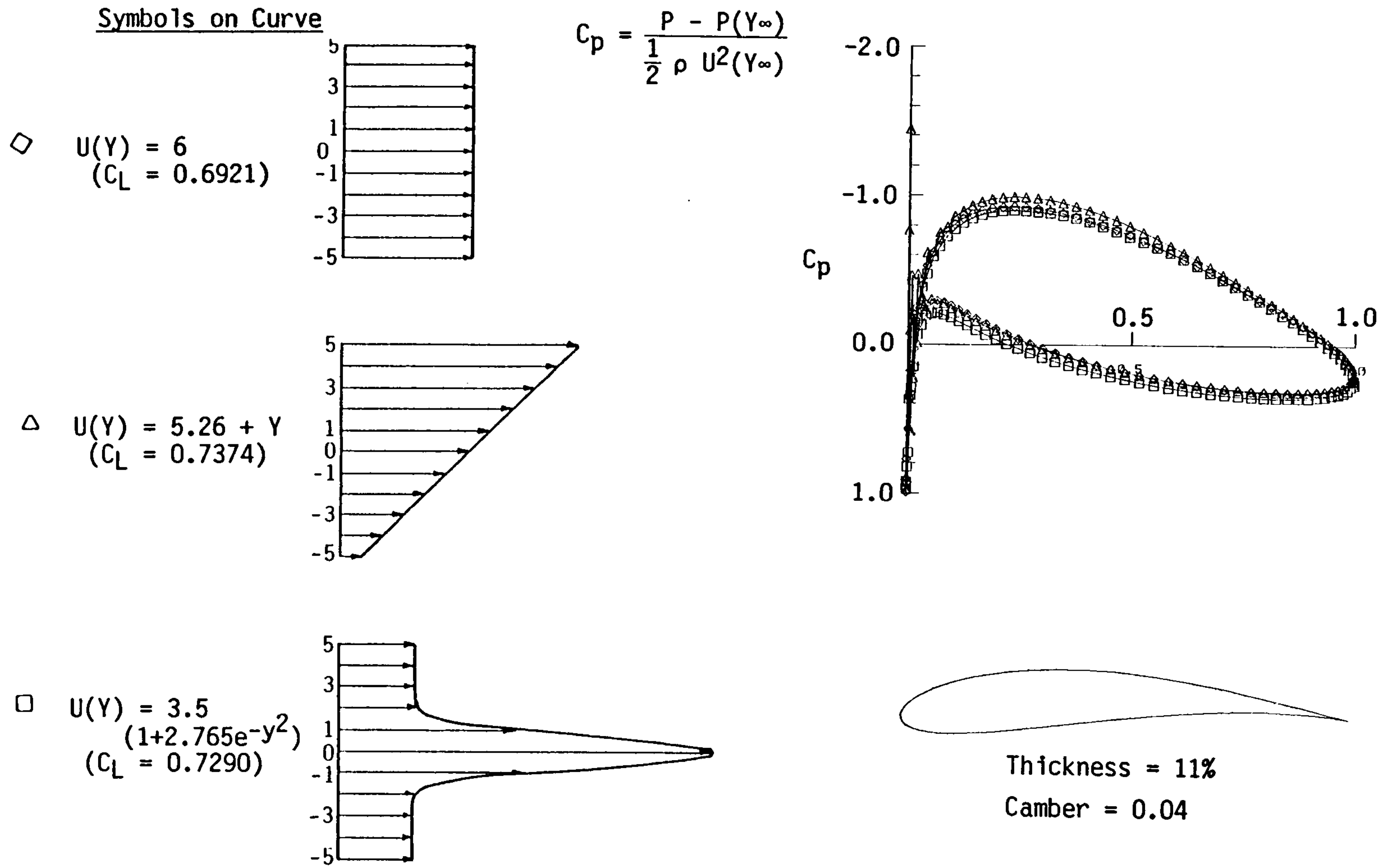


Figure 5.11. C_p diagram of various head wind profiles based on a common value of $U_o = [\int_{-L/2}^{L/2} U^2(y)\ dy/L]^{1/2}$, L the dimension of the domain.

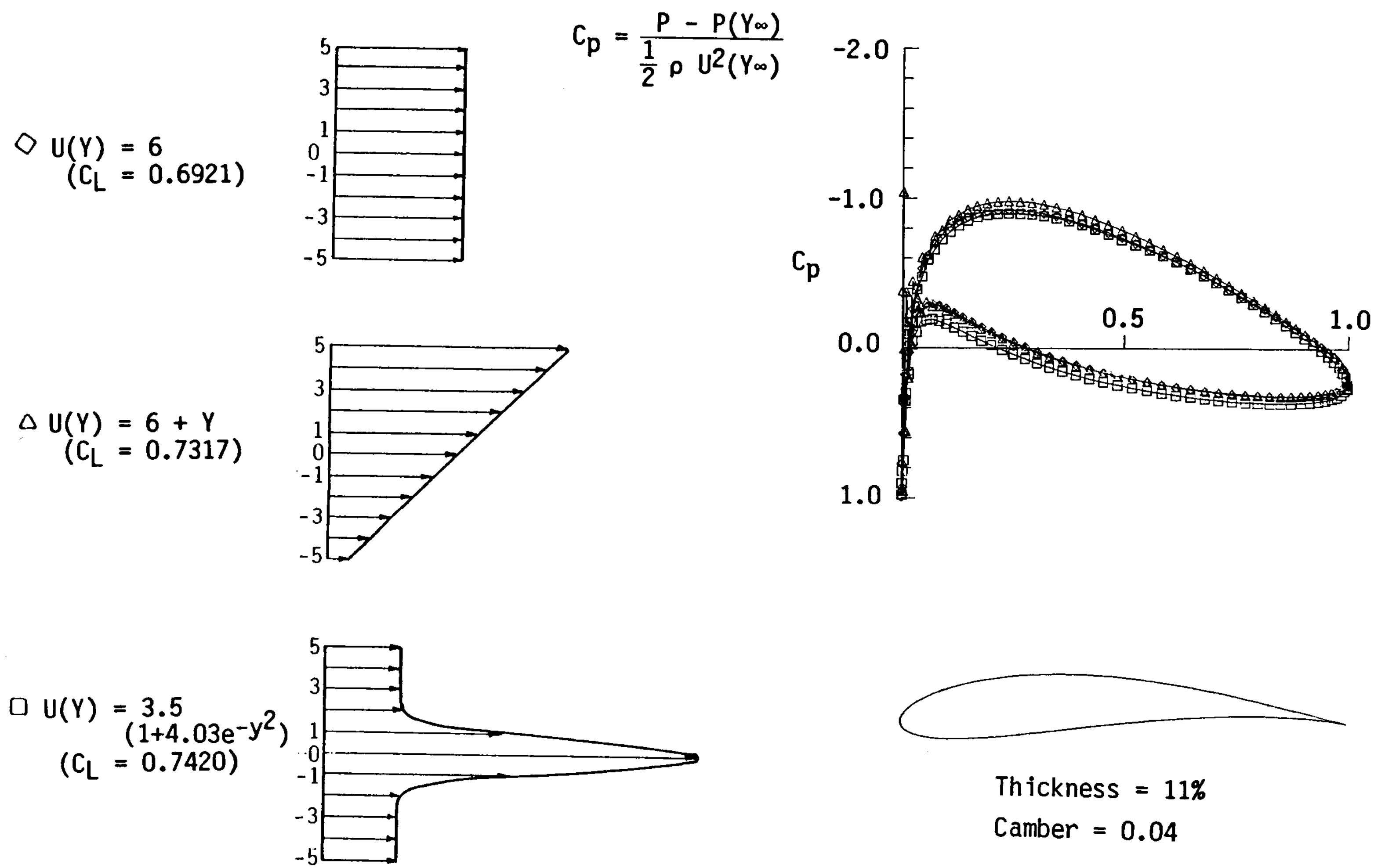


Figure 5.12. C_p diagram of various head wind profiles based on a common value of $U_o = \int_{-L/2}^{L/2} U(y)\, dy/L$, L the dimension of the domain.

The present method does not incur fictitious wall effects because the computational domain is wider than that of Chow and van der Vooren, and also because the influences due to the domain is indirect. For both the finite-difference and the finite-element method all the grid points in the computational domain need to be calculated directly. But for the scalar/vector potential panel method only the scalar and vector potentials on the body surface need to be calculated directly. The flow properties (such as velocity) on the grid points will only be updated by the perturbation of the scalar and vector potentials. These features of the scalar/vector potential panel method make it possible to do a preliminary calculation of a three-dimensional flow problem on a relatively small and slow computer system such as VAX-11/785.

C. TWO-DIMENSIONAL DOWNDRAFTS

In this section the influence of a downdraft on the lift of an airfoil is studied. The microburst wind fields are severe downdrafts accompanied by intense outflows and thus an understanding of the downdraft effect on airfoil lift is important. Figure 5.13 shows the influences of downdraft on a camber Joukowski airfoil for a uniform onset flow with three different kinds of downdraft: uniform, linear shear and nonlinear (cosine) shear. C_l values decrease from 0.6921 to -0.2768 ,-0.4468 and -0.5470 respectively where $\rho U^2(y_\infty)$ is the dynamic pressure used to nondimensionalize the lift. The major cause of the reduction is primarily effective decrease in the angle of attack. The concept of angle of attack however loses its meaning in nonuniform flow, because the angle of attack varies from point to point around airfoil. The angle of attack in the present study is general achieved by rotating the onset flow at each grid point in the flow field (see section D). This is equivalent to rotating the airfoil relative to the onset flow field. For the

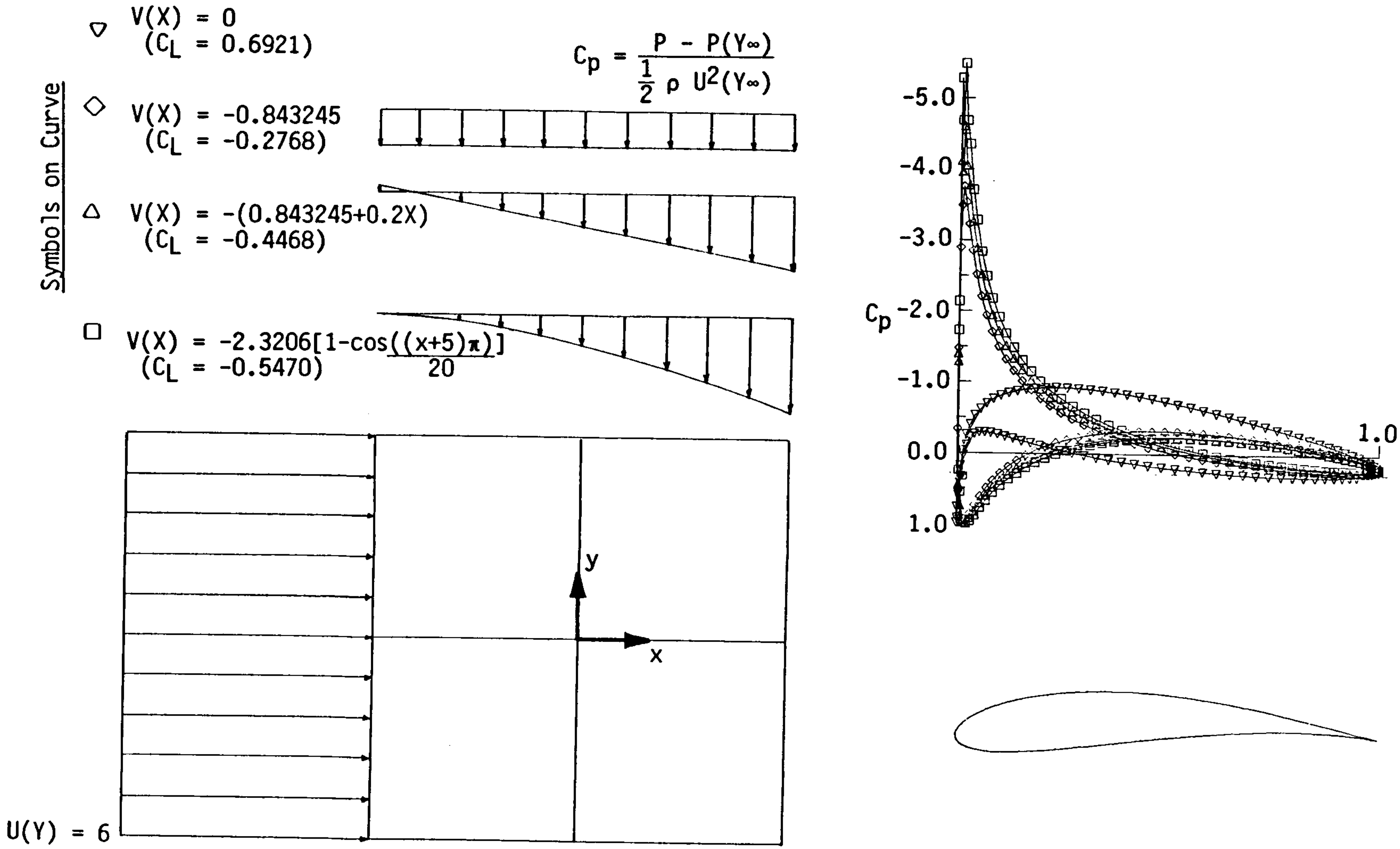


Figure 5.13. C_p diagram of a uniform head wind and various downdraft profiles based on a profile with a common value of $V_o = \int_{-L/2}^{L/2} V(x)\, dx/L$.

downdraft study no angle of attack was imposed other than occurring due to variations in the flow field. A similar decrease in C_l is shown in Figure 5.14, in which the downdraft flow shears in the reversed sense from that of Figure 5.13. Both results use a velocity profile with a common mass flow rate as a reference criterion. This criterion will induce stronger downdraft shear than that based on a common momentum profile.

The interesting thing is that the value of C_l in Figure 5.14 does not decrease as much as that in Figure 5.13. C_l drops from 0.6921 to -0.2768, -0.1067 and -0.1675 respectively in Figure 5.14. This phenomenon is contradictory to the previously stated concept that for onset vorticity in the same direction as that of the bound vorticity, a positive contribution is made to C_l. This is true for the onset shear flow without downdraft shown in Figure 5.11, but from the results in Figures 5.13 and 5.14, the effect is reversed. A theoretical explanation of the fact that downdraft induced vorticity with a conceptually unfavorable direction having less drop in C_l than a favorable vorticity is not readily available. The common and straightforward concept from uniform onset flow can be applied to nonuniform onset flow without downdrafts (Figure 5.11 and 5.12) but in this case not to the uniform flow with downdrafts (Figure 5.13 and 5.14). This probably occurs because in the latter case the lift is influenced by both the local angle of attack and the onset vorticity, whereas in the former case probably just the onset vorticity influences the lift (i.e. at the center of the flow field). As an estimate the local angle of attack in Figure 5.13 and 5.14 is -13^o. Since angle of attack changes from point to point, it is hard to separate the contribution to lift from angle of attack and that from vorticity. In Figure 5.13 and 5.14, the C_p distribution uses $\rho U^2(y_\infty)$ as reference dynamic pressure which is believed by Chow *et al* [18] to isolate the effect of shear. The $U(y_\infty)$ values for three different downdrafts

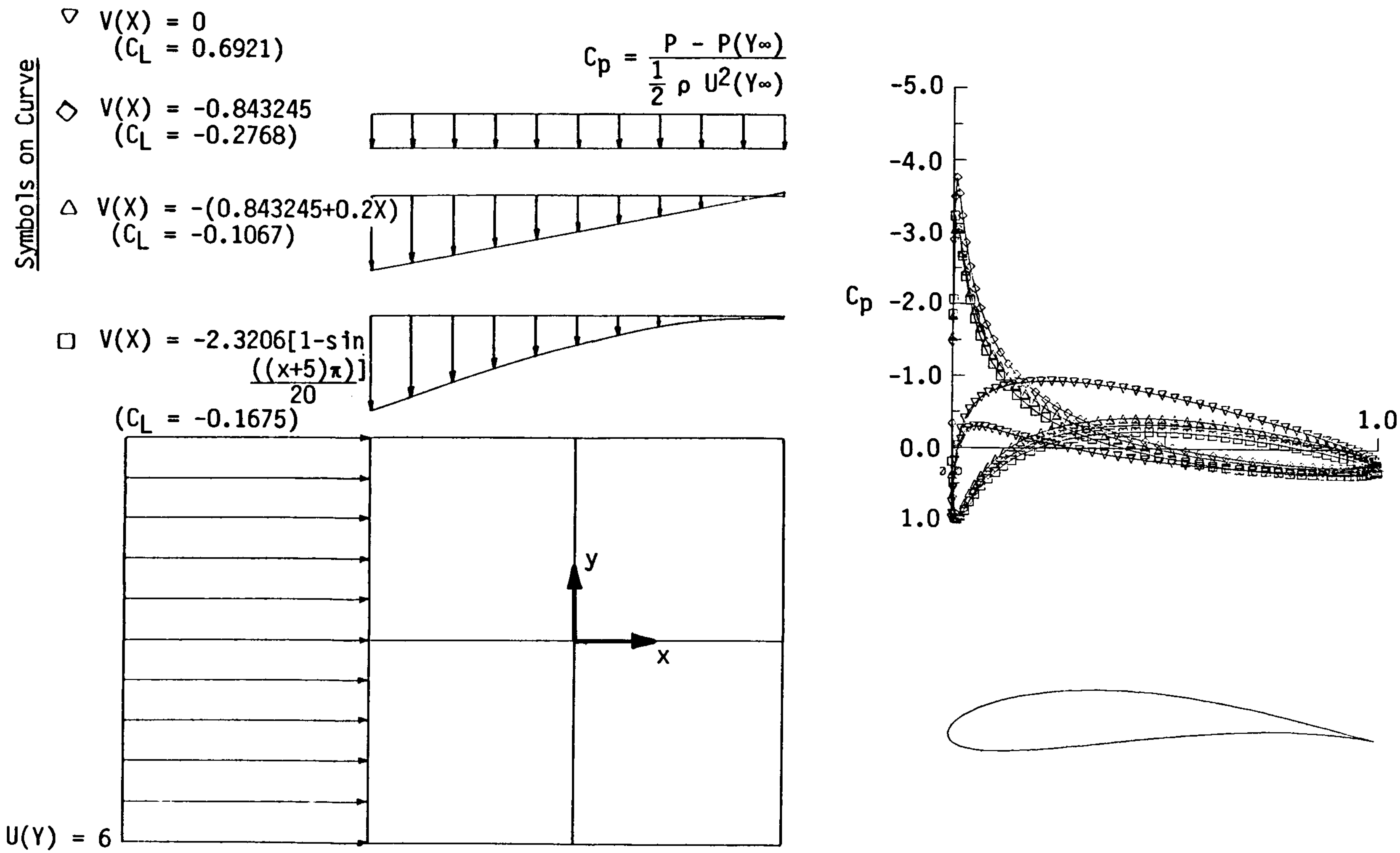


Figure 5.14. C_p diagram of a uniform head wind and various downdraft profiles based on a common value of $V_o = \int_{-L/2}^{L/2} V(x)\, dx/L$.

(uniform, linear shear and nonlinear shear) in Figure 5.13 are 6.059, 6.000 and 6.000 respectively (Note: $U(y_\infty) = \sqrt{U^2(-5,y) + V^2(-5,y)}$ unless the streamline moves to the upper boundary which was apparently not the case here). The $U(y_\infty)$ values in Figure 5.14 are 6.059, 6.277 and 6.433 respectively.

D. A TWO-DIMENSIONAL AIRFOIL IN JAWS WIND SHEAR

The forgoing results are obtained for simple forms of nonuniform approach flow. In order to investigate the influence of more realistic wind shears, measured data from the JAWS (Joint Airport Weather Studies) program [5] was input to the present calculation procedure. The JAWS wind shear data set provided the three wind speed components: longitudinal, latitudinal, and vertical, at each point of a 492 ft $\times$ 492 ft grid. The wind speed employed in the calculation was obtained by applying an area-weighted interpolation between grid points. Ten stations along a corridor of the Jaws data set were selected (see Figure 5.15) and the C_l' and C_m' of the airfoil with fixed attitude were calculated at each station. Quasi-steady state was assumed at each station.

The calculated forces will be quite different from those associated with the actual dynamic flight with transient effects and of a complete aircraft with fuselage and tail, but still interesting results can be observed from the airfoil model alone. Figure 5.16 (a) and (b) shows that the C_l' and C_m' variations, referenced to absolute airfoil velocity, for four different altitudes, 250 ft, 750 ft, 1250 ft and 1750 ft. For very large aircraft, ground effect will become important when the altitude is lower than 250 ft. The ground effect is not included in the present model however it would not be hard to implement the ground effect into the present model. Figure 5.17 (a) and (b) shows similar calculations except at a different angle of attack. As mentioned in the last section, the angle of attack is defined as rotating the

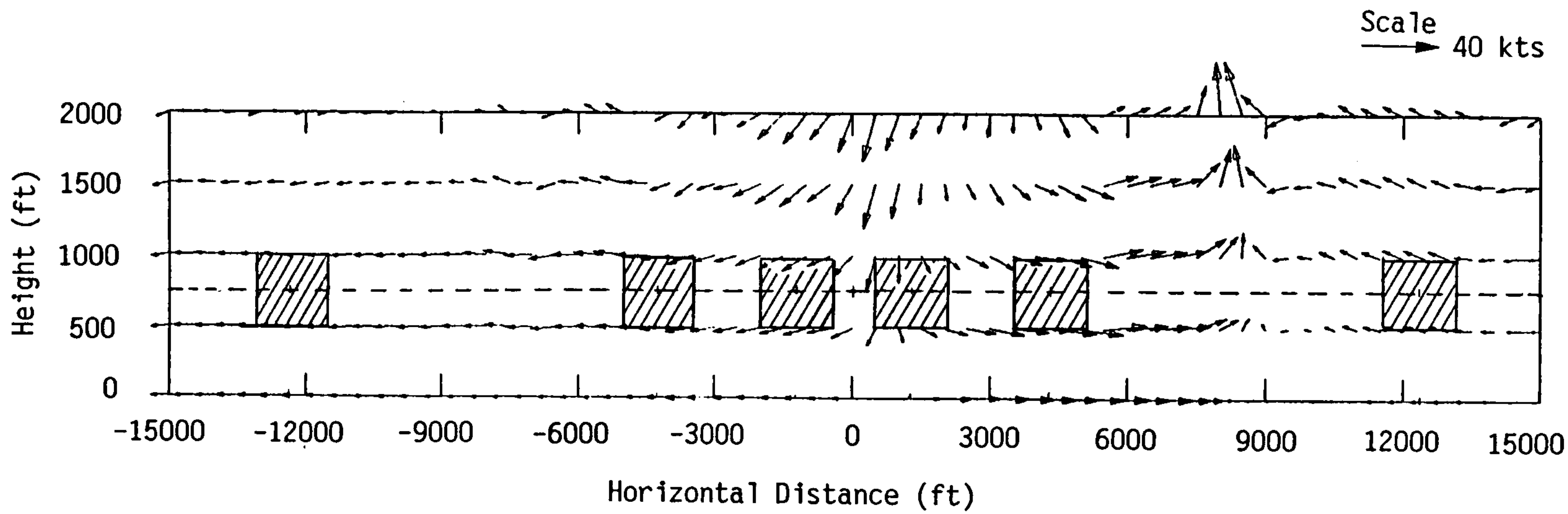
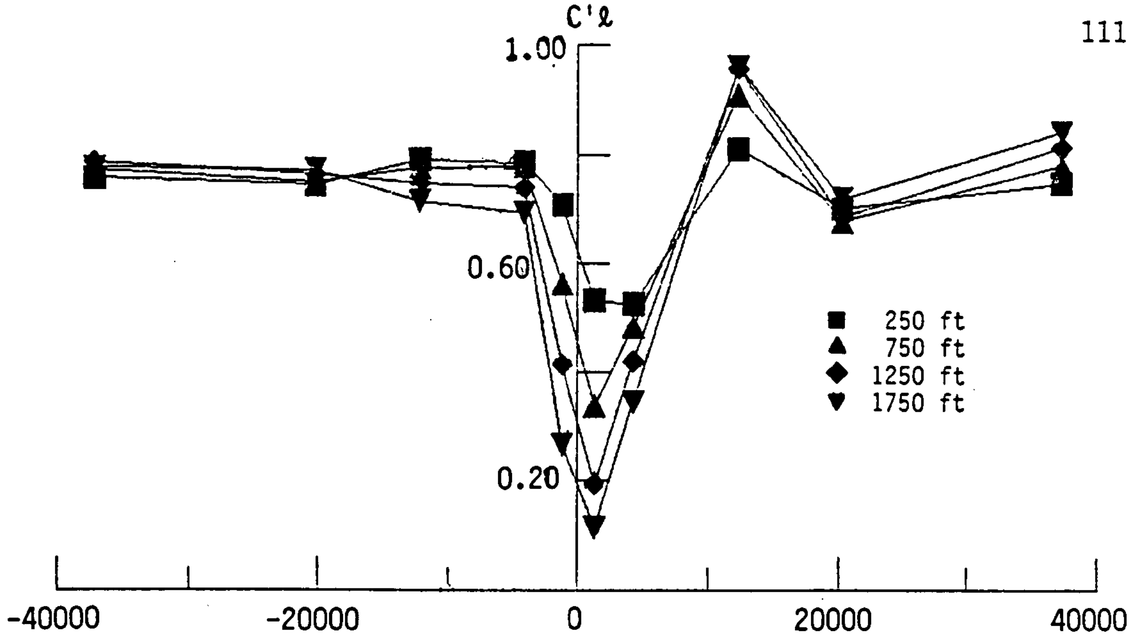
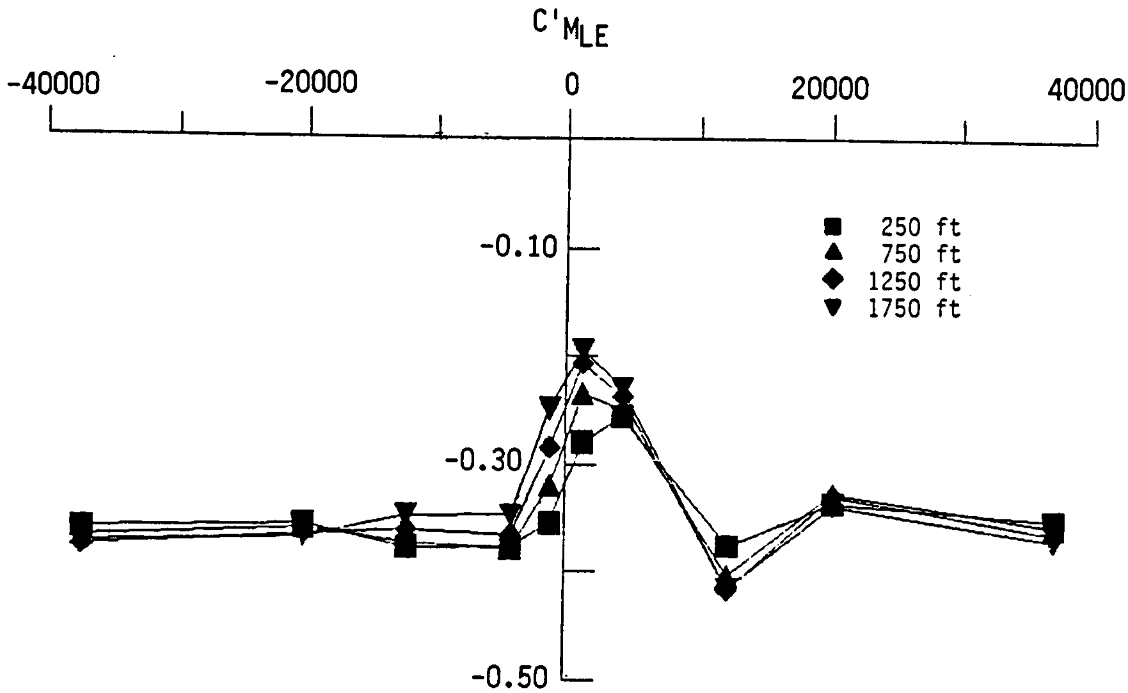


Figure 5.15. Illustration of a simulated flight of an airfoil flying through JAWS wind shear data, 5AU1847AB.

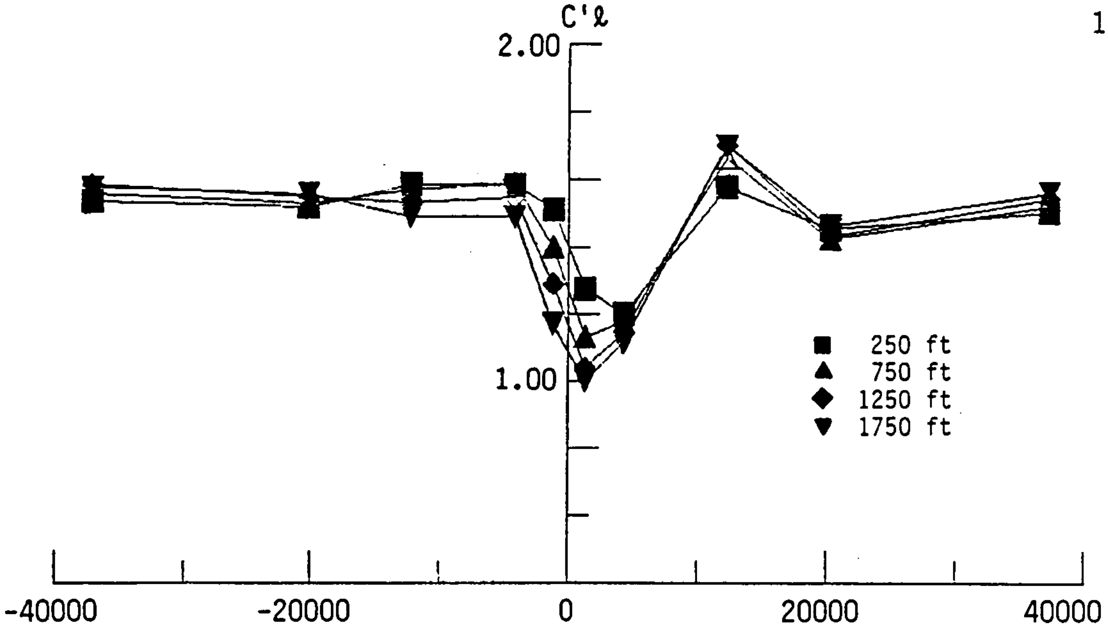


(a) Lift coefficient.

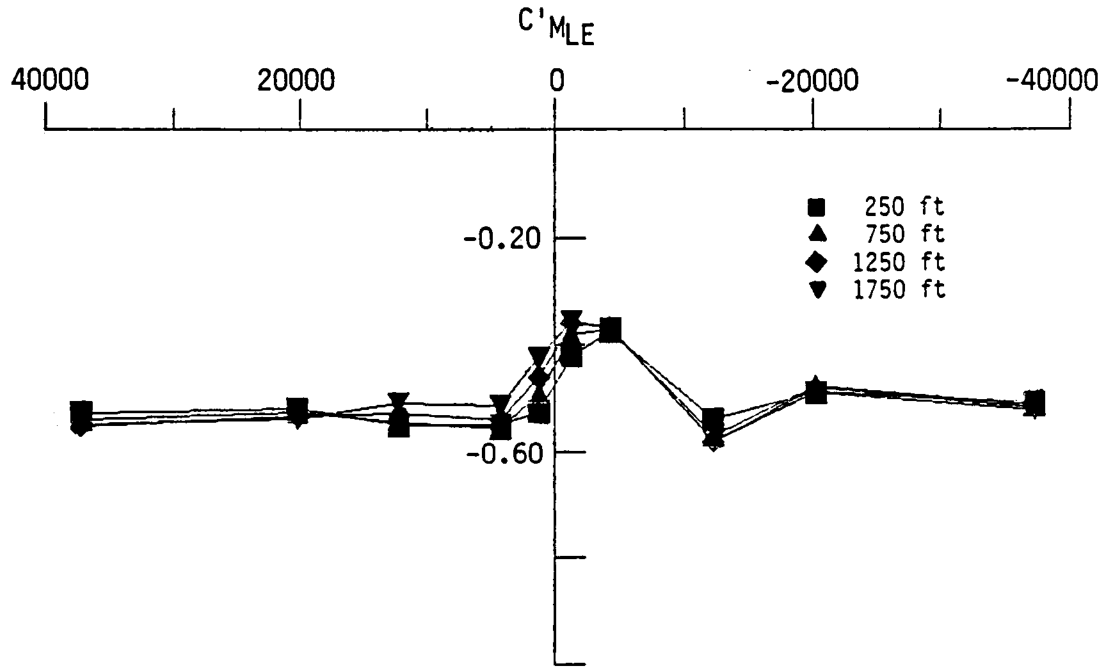


(b) Moment coefficient.

Figure 5.16. Lift and moment variations of an airfoil in JAWS wind shear for four different altitudes ($\alpha = 0^o$, $V_{air} = 100m/s$ $= 194.2kts.$, 5AU1847AB.COR).



(a) Lift coefficient.



(b) Moment coefficient.

Figure 5.17. Lift and moment variations of an airfoil in JAWS wind shear for four different altitudes ($\alpha = 5^o$, $V_{air} = 100m/s = 194.2kts.$, 5AU1847AB.COR).

onset flow relative to the airfoil. Because the local angle of attack is different from point to point in the nonuniform flow field. It is observed that C_l' drops around 20 % - 60 % for different altitudes and that C_m' has a strong nose-up tendency, when passing the center of the downburst. This loss of lift, and nose-up pitch are a direct consequence of the downdraft and the increase in tailwind that occur as the airfoil emerges from the downburst. For an aircraft in its landing configuration (high incidence and large flap deflection) such a rapid change of force and moment is difficult to control (partly due to the slow response of the aircraft to increase engine power in this configuration), placing the aircraft in a hazardous condition. However, the above observations can also be found from early investigations based on the point mass and uniform flow assumptions. That means the wind shear effect is insignificant in the present calculation by using the JAWS wind field data. Only the strong downdraft effect, or reducing the local angle of attack, is shown. In order to further investigate the wind shear effects, a comparison of the C_l variations between a nonuniform flow model, a uniform flow model and an empirical model is shown in Figure 5.18. The wind field velocity at the center of the computational domain (see Figure 4.2 page 80) is calculated by interpolating the measured JAWS wind field data at four corners and then is input into the computer program for the uniform flow model. Additionally results generated from empirical design equations for uniform flow are also included in the Figure 5.18. These equations are based on a computational and empirical model from a commercial airline company computer code for lift and drag. From Figure 5.18 the two-dimensional results are basically similar to the three-dimensional empirical results. The differences between the two-dimensional nonuniform (i.e. with wind shear) and uniform (i.e. without wind shear) flows are the effects of wind shear

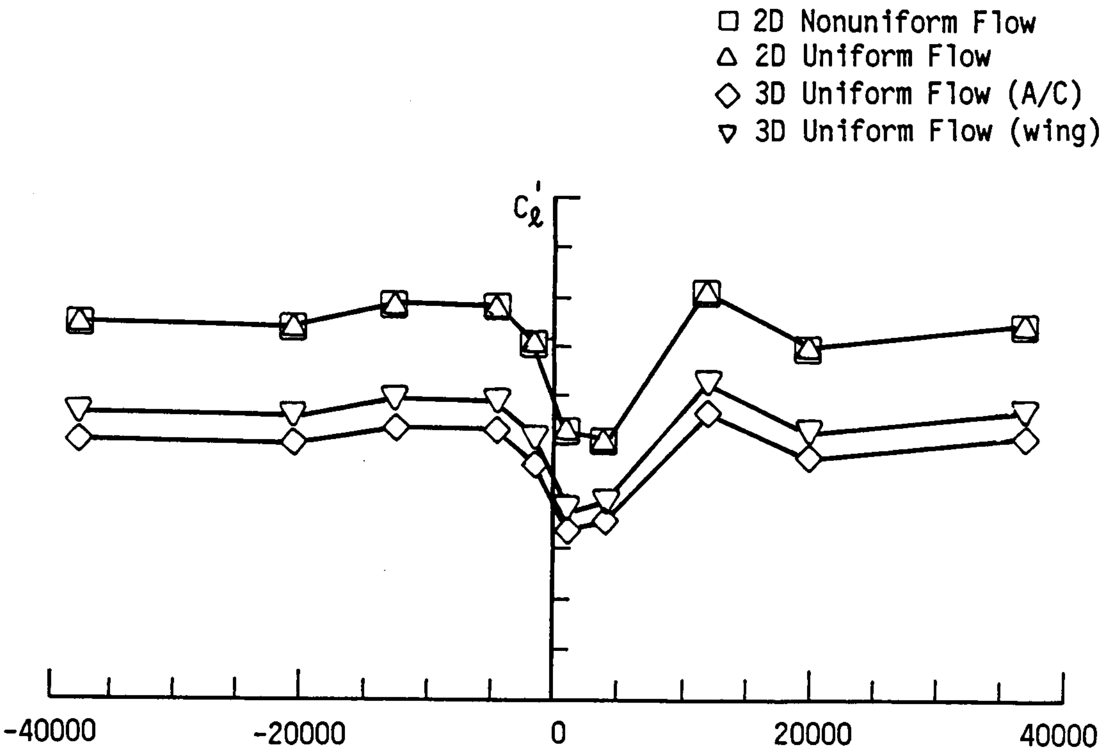


Figure 5.18. Comparison of present model (w. & w/o wind shear) with empirical model (250 ft. $\alpha = 0^o$).

over the airfoil (i.e. the area integration in the flow field domain). Figure 5.18 shows that the contribution of the area integral due to the JAWS wind shear data is insignificant.

A number of reasons for the wind shear effects not being significant are readily apparent. The maximum onset vorticity of the JAWS wind field data is of the order of 10^{-3} which is a very small number. Besides the JAWS wind field data (no larger than 20 kts) is small compared to the air speed (almost 200 kts). Based on the above reasons, it is clear why the difference between the two inviscid models (with or without wind shear) is small. Although the present method has all the advantages as described before, it is still an inviscid model which can not predict the viscous effects such as flow separation and viscous drag. It is believed that for such a strong downdrafts viscous effects should be a vital factor and the flow separation phenomenon may be crucial in the present problem. However, the present method is still very helpful for the preliminary design and study of the wind shear effects. The calculated C_l and C_m coefficients can be used to analyze an airfoil in a quasi-steady simulated flight through JAWS microburst.

CHAPTER VI

CONCLUSION

A recent innovation in aerodynamic calculation techniques is a formulation for the velocity field based upon scalar and vector potentials (Stokes-Helmholtz decomposition technique). This technique is presented and implemented into the panel method such that it is able to solve not only irrotational flow but also rotational flow. The application of interest is to two-dimensional airfoils moving into a nonuniform approach flow.

The present method also has general applicability to many other areas such as airfoils in various uniform or nonuniform shear flows, and the comparisons with theoretical and small perturbation theories [10-13], or finite-difference [18] and finite-element [20] methods are also good. More specifically, the preceding calculation results reveal that the present formulation has the following advantages: (1) It can be used for different kinds of airfoil shape, including symmetric or cambered, Joukowski or NACA-4 digit, with or without cusped trailing edge. (2) It can work with various onset velocity profiles without restriction to small disturbances. (3) It can simulate the infinite space, in good agreement with the solutions of finite-difference [18] or finite-element [20] methods.

An innovation to the panel method consisted of splitting the onset velocity into two components (see Figure 2.9 page 32) one along the zero-lift line and one perpendicular to the zero-lift line. This approach provided a stable and efficient way to solve airfoils with cusped trailing edge using the lower-order panel methods. The inappropriateness of using higher-order panel methods and of imposing an artificial vorticity distributor for airfoils with cusped trailing edge, as suggested by Hess [46], is also explained. The coefficient matrix (i.e. the left-hand-side) of the system of equations is normally thought to be the most sensitive part of the analysis when solving the system of equations. This is true for either direct matrix inversion or for any other elimination procedures. It is believed that this idea led most researchers in the 1970s to raise the order of the panel method in order to adjust the coefficient matrix to attain a numerically superior form. However it is shown (see Chapter II, on the basis of eigensystem analyses, Equation 2.23) that the influence of the right-hand-side of the system of equations is also crucial. If the eigenvalues λ_i are widely spaced, any small change of the right-hand-side such as to α_i and β_i, will be magnified by either λ_i or $1/\lambda_i$, which can lead to large errors.

The present study demonstrates the same hazard to flight in microbursts as reported elsewhere, but it is concluded that the shear distribution over the airfoil dose not significantly effect the aerodynamic coefficients. The present method indicates that only the decrease of the effective angle of attack is an important factor. However, in a wind field with intense downdrafts, viscous effects and flow separation could also be crucial.

The present inviscid model is shown to be an effective tool for preliminary studies of the wind shear effects on aerodynamic coefficients. It cannot however predict viscous effects and flow separation phenomena. Moreover the model only investigates two-dimensional flow effects. It is expected that the shear over a three-dimensional airfoil in the spanwise direction will have a much more profound influence on lift and particularly rolling moment. The formulation of the three-dimensional scalar/vector potential solution is developed. It is recommended that the computer algorithm be programmed and analysis of wind shear be extended to include spanwise variations

BIBLIOGRAPHY

BIBLIOGRAPHY

1. Fujita, T. T., "Downbursts and Microbursts - An Aviation Hazard," Proceedings of the 19th Conference on Radar Meteorology, American Meteorological Society, Boston, MA, 1980, pp. 94-101.

2. Fujita, T. T., "Microbursts as an Aviation Wind Shear Hazard," AIAA paper No. 81-0386, 1981.

3. Frost, W., Camp, D. W., and Wang, S. T., "Wind Shear Modeling for Aircraft Hazard Definition," FAA-RD-78-3, February 1978.

4. Ivan, M., "A Ring-Vortex Downburst Model for Flight Simulations," Journal of Aircraft, Vol. 23, March 1986, pp. 232-236.

5. McCarthy, J., "Joint Airport Weather Studies (JAWS) Project," Proceedings: Sixth Annual Workshop on Meteorological and Environmental Inputs to Aviation Systems, October 26-28, 1982, NASA CP-2274, DOT/FAA/RD-83/67.

6. Frost, W. and Bowles, R., "Wind Shear Terms in the Equations of Aircraft Motion," Journal of Aircraft, Vol. 21, November 1984, pp. 866-872.

7. Frost, W., Chang, H. P., Elmore, K. L., and McCarthy, J., "Simulated Flight Through JAWS Wind Shear," Journal of Aircraft, Vol. 21, October 1984, pp. 797-802.

8. Psiaki, M. L. and Stengel, R. F., "Optimal Flight Paths Through Microburst Wind Profiles," Journal of Aircraft, Vol. 23, August 1986, pp. 629-635.

9. von Sanden, H., "Über den Auftrieb im natürlichen Winde," Zeitschrift für Mathematik & Physik, Vol. 61, 1912, p. 225.

10. Tsien, H. S., "Symmetric Joukowsky Airfoils in Shear Flow," Quarterly of Applied Mathematics, Vol. I, 1943, pp. 130-148.

11. Sowyrda, A., "Theory of Cambered Joukowsky Airfoils in Shear Flow," Cornell Aero. Lab. Rep. No. AI-1190-A2, September 1958.

12. Jones, E. E., "The Forces on a Thin aerofoil in Slightly Parabolic Shear Flow," Zeitschrift für Angewandte Mathematik & Mechanik, Vol. 37, 1957, pp. 362-370.

13. Jones, E. E., " The Elliptical Cylinder in a Shear Flow with Hyperbolic Velocity Profile," Quarterly Journal of Mechanics and Applied Mathematics, Vol. 12, 1959, pp. 191-210.

14. Dowell, E. H., "Generalized Aerodynamic Forces on a Flexible Plate Undergoing Transient Motion in a Shear Flow with an Application to Panel Flutter," AIAA Journal, Vol. 9, May 1971, pp. 834-841.

15. Yates, J. E., "Linearized Integral Theory of Three-Dimensional Unsteady Flow in a Shear Layer," AIAA Journal, Vol. 12, May 1974, pp. 596-601.

16. Ventres, C. S., "Shear Flow Aerodynamics: Lifting Surface Theory," AIAA Journal, Vol. 13, September 1975, pp. 1183-1189.

17. Williams, M. H., Chi, M. R., Ventres, C. S., and Dowell, E. H., "Aerodynamic Effects of Inviscid Parallel Shear Flows," AIAA Journal, Vol. 15, August 1977, pp. 1159-1166.

18. Chow, F., Krause, E., Liu, C. H., and Mao, J., "Numerical Investigations of an Airfoil in Nonuniform Stream," Journal of Aircraft, Vol. 7, June 1970, pp. 531-537.

19. Ludwig, G. R. and Erickson, J. C., "Airfoils in Two-Dimensional Nonuniformly Sheared Slipstreams," Journal of Aircraft, Vol. 8, November 1971, pp. 874-880.

20. van der Vooren, J. and Labrujere, T. E., "Finite Element Solution of the Incompressible Flow over an Airfoil in a Non-uniform Stream," International Conference on Numerical Methods in Fluid Dynamics, University of Southampton, England, September 26-28, 1973, pp. 23-41.

21. Hanin, M. and Barsony-Nagy, A., "Slender Wing Theory for Nonuniform Stream," AIAA Journal, Vol. 18, April 1980, pp. 381-384.

22. Hanin, M. and Barsony-Nagy, A., "Induced Drag of a Slender Wing in a Nonuniform Stream," AIAA Journal, Vol. 21, May 1983, pp. 781-783.

23. Ecer, A., Akay, H. U., and Sener, B., "Solution of Three-Dimensional Inviscid Rotational Flows in a Curved Duct," AIAA paper No. 84-0032.

24. Orszag, S. A. and Israeli, M., "Numerical Simulation of Viscous Incompressible Flows," Annual Review of Fluid Mechanics ,1974, 6 : 281-318.

25. MacCormack, R. W. and Lomax, H., "Numerical Solution of Compressible Viscous Flows," Annual Review of Fluid Mechanics, 1979, 11 : 289-316.

26. Shen, Shan-fu, "Finite-Element Methods in Fluid Mechanics," Annual Review of Fluid Mechanics, 1977, 9 : 421-445.

27. Hirasaki, G. J. and Hellums, J. D., "A General Formulation of the Boundary Conditions on the Vector Potential in Three Dimensional Hydrodynamics," Quarterly of Applied Mathematics, Vol. 26, No. 3, April 1968, pp. 331-342.

28. Hirasaki, G. J., "Boundary Conditions on the Vector and Scalar Potentials in Viscous Three-Dimensional Hydrodynamics," Quarterly of Applied Mathematics, Vol. 28, No. 2, July 1970, pp. 293-296.

29. Richardson, S. M. and Cornish, A. R. H., "Solution of Three Dimensional Incompressible Flow Problems," Journal of Fluid Mechanics, Vol. 82, 1977, pp. 309-319.

30. Gustafson, K. and Halasi, K., "On the Divergence-Free (i.e. Mass Conservation, Solenoidal) Condition in Computational Fluid Dynamics: How important is it ?" Proceedings: Third International Conference on Numerical Methods in Laminar and Turbulent Flow, Seattle, Washington, 1983, pp. 617-626.

31. Gustafson, K. and Halasi, K., "A new Method for Computing Solenoidal Vector Fields on Arbitrary Regions," International Journal for Numerical Methods in Fluids, Vol. 5, 1985, pp. 763-783.

32. Morino, L., " Scalar/Vector Potential Formulation for Compressible Viscous Unsteady Flows," NASA-CR-3921, 1985.

33. Morino, L., "Helmholtz Decomposition Revisted: Vorticity Generation and Trailing Edge Condition," Computational Mechanics, Springer-Verlag, 1986, pp. 65-90.

34. Del Marco, S. P., "The Mathematical Foundations of the Scalar-Vector Potential Approach to Analyzing Viscous Flows," Ph.D. Thesis, Boston University, Department of Mathematics, 1986.

35. Tutty, O. R., "On Vector Potential-Vorticity Methods for Incompressible Flow Problems," Journal of Computational Physics, Vol. 64, 1986, pp. 368-379.

36. Lemos, N. A., "Symbolic Proof of the Helmholtz Theorem," American Journal of Physics, 55 (1), January 1987, pp. 57-59.

37. Morino, L., "Nonpotential Aerodynamics for Windmills in Shear Winds," ERDA/NSF 2975, Wind Energy Conversion Systems Workshop , Washington D. C., June 9-11, 1975

38. Morino, L. et al., "Nonpotential Aerodynamics for Buildings in Shear Winds," Boston University TR-78-01, 1977.

39. Lacor, C. and Hirsch, C., "Rotational Flow Calculations in Three-Dimensional Blade Passages," ASME paper No. 82-GT-316.

40. Chaviaropoulos, P., Giannakoglou, K., and Papailiou, K. D., "Numerical Computation of Three-Dimensional Rotational Inviscid Subsonic Flows using the Decomposition of the Flow Field into a Potential and a Rotational Part," ASME-GT-169, 1986.

41. Chaderjian, N. M., Steger, J. L., "The Numerical Simulation of Steady Transonic Rotational Flow Using a Dual Potential Formulation," AIAA paper No. 85-0368.

42. Rao, K. V., Steger, J. L., and Pletcher, R. H., "A Three-Dimensional Dual Potential Procedure for Inlets and Indraft Wind Tunnels," AIAA paper No. 87-0598.

43. Hess, J. L. and Smith, A. M. O., "Calculation of Potential Flow about Arbitrary Bodies," Progress in Aeronautical Sciences, Vol. 8, pp. 1-138, Pergamon Press.

44. Hunt, B., "The Panel Method for Subsonic Aerodynamic Flows: A Survey of Mathematical Formulations and Numerical Models and an Outline of the New British Aerospace Scheme," Computational Fluid Dynamics, Chapter 2, VKI Book, Wolfgang Kollmann ed., 1980.

45. Hess, J. L., "Calculation of Potential Flow about Arbitrary Three-Dimensional Lifting Bodies," MDC-J5679-01, 1972

46. Hess, J. L., "The Use of Higher-Order Surface Singularity Distributions to Obtain Improved Potential Flow Solutions for Two-Dimensional Lifting Airfoils," Computer Methods in Applied Mechanics and Engineering, No. 5, 1975, pp. 11-35.

47. Maskew, B., "Prediction of Subsonic Aerodynamic Characteristics: A Case for Lower-Order Panel Methods," Journal of Aircraft, Vol. 19, No. 2, 1982, pp. 157-163.

48. Ardonceau, P. L., "Computation of the Potential Flow over Airfoils with Cusped or Thin Trailing Edges," AIAA Journal, Vol. 24, No. 8, 1986, pp. 1375-1377.

49. Groh, G. Gy., "Comment on "Prediction of Subsonic Aerodynamic Characteristics: A Case for Lower-Order Panel Methods"," Journal of Aircraft, Vol. 22, No. 1, 1985, pp. 92-93.

50. Wang, J. C. A. and Frost, W., "On the Order of Panel Methods," AIAA/ASME/SIAM/APS 1st National Fluid Dynamics Congress, July 25-28, 1988, Cincinnati, Ohio, Part 1, 88-3798-CP, pp. 644-651

51. Karamcheti, K., Principles of Ideal-Fluid Aerodynamics, Chapter 17, John Wiley & Sons, Inc., 1966.

52. Mangler, K. W.,"Improper Integrals in Theoretical Aerodynamics", Rept. Aero. 2424, C. P. 94, 1951

53. Abbott, I. A. and von Doenhoff, A. E., Theory of Wing Sections, Chapter 6, Dover Publications, Inc., 1959

54. Isaacson, E. and Keller, H. B., Analysis of Numerical Methods, New York, Wiley, 1966

55. Vidal, R. J., "The Influence of Two-Dimensional Stream Shear on Airfoil Maximum Lift" Journal of the Aerospace Sciences, Vol. 29, August, 1962.

APPENDIXES

APPENDIX A

THE RELATIONSHIP BETWEEN DOUBLET AND VORTEX SHEETS

This appendix develops the relationship between a doublet and vortex sheets and demonstrates that for a closed surface a distribution of doublets can be replaced with a vortex sheet. The derivations basically follow Reference [45]. Consider a surface S in space bounded by a closed curve C (see Figure A.1). If S is a closed surface, C vanishes. At any point (x_o, y_o, z_o) on S the unit normal vector is $\vec{\mathbf{n}}$, and at any point on C the unit tangent vector is $\vec{\mathbf{t}}$. The vector between the point (x_o, y_o, z_o) and a general point (x, y, z) in space is denoted $\vec{\mathbf{r}}$, where

$$\vec{\mathbf{r}} = (x - x_o)\ \hat{\mathbf{i}} + (y - y_o)\ \hat{\mathbf{j}} + (z - z_o)\ \hat{\mathbf{k}} \tag{A.1}$$

$$|\ \vec{\mathbf{r}}\ | = \sqrt{(x - x_o)^2 + (y - y_o)^2 + (z - z_o)^2} \tag{A.2}$$

The gradient operator ∇_x is used to denote that derivatives with respect to (x, y, z) and ∇_{x_o} is with respect to (x_o, y_o, z_o).

According to Biot-Savart law the velocity at (x, y, z) due to a vortex filament of variable strength Γ lying along any curve C is given by:

$$\vec{\mathbf{V}} = \int_C \frac{\vec{\mathbf{t}} \times \vec{\mathbf{r}}}{r^3}\ \Gamma\ d\ s \tag{A.3}$$

where s denotes the arc length along C. If the velocity due to the vortex filament with the strength Γ equivalent to the doublet strength μ on a closed curve C, then Equation (A.3) can be expressed as

$$\vec{\mathbf{V}}_\Gamma = \int_C \frac{\vec{\mathbf{t}} \times \vec{\mathbf{r}}}{r^3}\ \mu\ d\ s \tag{A.4}$$

The expression for the velocity due to a vortex sheet can be obtained from Equation (A.4) by writing the vector vortex strength $\vec{\gamma} = \Gamma\vec{\mathbf{t}}$ so that Equation (A.3) becomes

$$\vec{\mathbf{V}} = \int_C \frac{\vec{\gamma} \times \vec{\mathbf{r}}}{r^3}\ d\ s \tag{A.5}$$

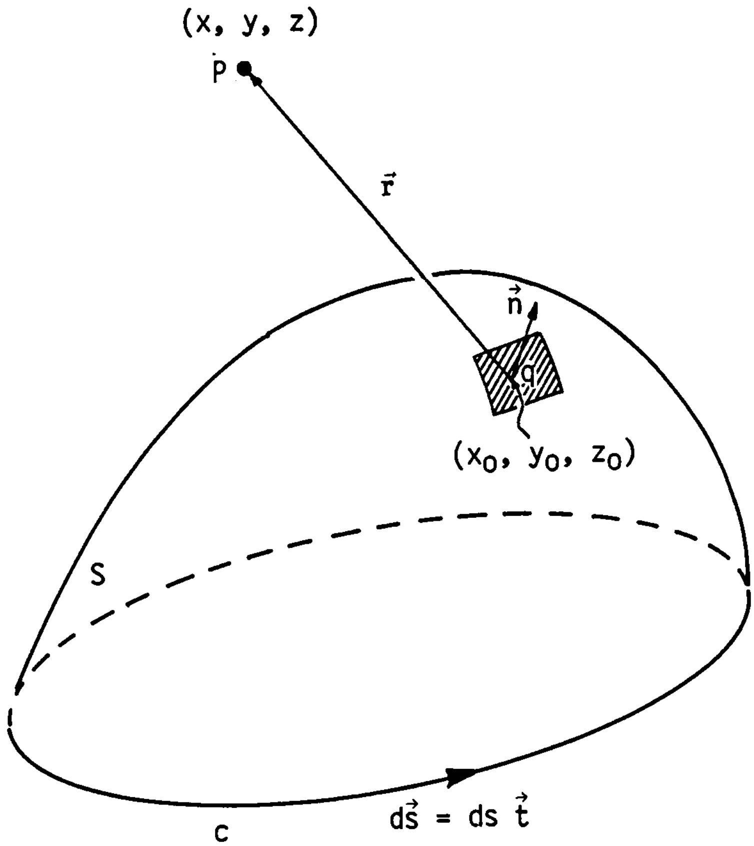


Figure A.1. Notation for a general surface.

Now simply redefine $\vec{\gamma}$ as a surface density instead of a linear density and rewrite Equation (A.5) to a surface integral over S. This gives the velocity at any point (x, y, z) due to a vortex distribution of strength $\vec{\gamma}$ on S as

$$\vec{\mathbf{V}}_\gamma = \int_S \int \frac{\vec{\gamma} \times \vec{\mathbf{r}}}{r^3} \, d\, S \tag{A.6}$$

where dS is an elemental surface area on S. From the theorem of Hess [45] the particular vortex strength can be expressed as $\vec{\gamma} = -\vec{\mathbf{n}} \times \nabla_{x_o} \mu$ then Equation (A.6) can be rewritten as

$$\vec{\mathbf{V}}_\gamma = -\int_S \int \frac{(\vec{\mathbf{n}} \times \nabla_{x_o} \mu) \times \vec{\mathbf{r}}}{r^3} \, d\, S \tag{A.7}$$

or

$$\vec{\mathbf{V}}_\gamma = -\int_S \int \frac{(\vec{\mathbf{n}} \cdot \vec{\mathbf{r}}) \nabla_{x_o} \mu - (\vec{\mathbf{r}} \cdot \nabla_{x_o} \mu) \vec{\mathbf{n}}}{r^3} \, d\, S \tag{A.8}$$

The velocity potential of a doublet sheet is

$$\begin{aligned} \phi_d &= \vec{\mathbf{n}} \cdot \nabla_{x_o} (\frac{-1}{r}) \\ &= -\frac{\vec{\mathbf{n}} \cdot \vec{\mathbf{r}}}{r^3} \end{aligned} \tag{A.9}$$

where $\vec{\mathbf{n}}$ is the unit vector along the axis of the doublet, and in this application the axis is along the normal vector to S. Therefore the velocity due to the doublet is

$$\begin{aligned} \vec{\mathbf{V}}_d &= \nabla\, \phi_d = \nabla(-\frac{\vec{\mathbf{n}} \cdot \vec{\mathbf{r}}}{r^3}) \\ &= \frac{-1}{r^3} \nabla(\vec{\mathbf{n}} \cdot \vec{\mathbf{r}}) - (\vec{\mathbf{n}} \cdot \vec{\mathbf{r}}) \nabla(\frac{1}{r^3}) \end{aligned} \tag{A.10}$$

Equation (A.10) can be further expressed as

$$\begin{aligned} \vec{\mathbf{V}}_d &= \frac{-1}{r^3} (\vec{\mathbf{n}} \cdot \nabla) \vec{\mathbf{r}} - (\vec{\mathbf{n}} \cdot \vec{\mathbf{r}}) \nabla(\frac{1}{r^3}) \\ &= -\frac{\vec{\mathbf{n}}}{r^3} + 3\, \vec{\mathbf{n}} \cdot \vec{\mathbf{r}} \frac{\vec{\mathbf{r}}}{r^5} \end{aligned} \tag{A.11}$$

Thus, the velocity at any point p (x, y, z) due to a normal doublet distribution of strength μ on S is

$$\vec{\mathbf{V}}_d = \int_S\int\left[-\frac{\vec{\mathbf{n}}}{r^3} + 3(\vec{\mathbf{n}}\cdot\vec{\mathbf{r}})\frac{\vec{\mathbf{r}}}{r^5}\right]\mu\ d\ S \tag{A.12}$$

The proof of the theorem stated in Chapter II (see page 20) consists of showing that $\vec{\mathbf{V}}_d$ from Equation (A.12) equals the sum of $\vec{\mathbf{V}}_\Gamma$ from Equation (A.4) and $-\vec{\mathbf{V}}_\gamma$ from Equation (A.8). This can be done as follows: (1) Write out the line integral Equation (A.4) in terms of components. (2) Apply the Stokes' theorem to each component separately to obtain surface integrals over S. (3) Manipulate the result to obtain the expected equality.

The details of the proof are shown in the following paragraphs. For a point q on the curve C (x_o, y_o, z_o) are functions of the arc length s along a curve. The unit tangent vector to C is

$$\vec{\mathbf{t}} = \frac{dx_o}{ds}\,\hat{\mathbf{i}} + \frac{dy_o}{ds}\,\hat{\mathbf{j}} + \frac{dz_o}{ds}\,\hat{\mathbf{k}} \tag{A.13}$$

Take the cross product with $\vec{\mathbf{r}}$ from Equation (A.1) and substitute the result into Equation (A.4) such that

$$\begin{aligned}\vec{\mathbf{V}}_\Gamma = \hat{\mathbf{i}}\int_C&\left[+\mu\frac{z-z_o}{r^3}\ d\ y_o - \mu\frac{y-y_o}{r^3}\ d\ z_o\right]\\ +\hat{\mathbf{j}}\int_C&\left[-\mu\frac{z-z_o}{r^3}\ d\ x_o + \mu\frac{x-x_o}{r^3}\ d\ z_o\right]\\ +\hat{\mathbf{k}}\int_C&\left[\mu\frac{y-y_o}{r^3}\ d\ x_o - \mu\frac{x-x_o}{r^3}\ d\ y_o\right]\end{aligned} \tag{A.14}$$

The Stokes theorem in component form is

$$\int_C [P\ dx_o + Q\ dy_o + R\ dz_o] =$$

$$\int_S\int \left[(\frac{\partial R}{\partial y_o} - \frac{\partial Q}{\partial z_o})\ n_1 + (\frac{\partial P}{\partial z_o} - \frac{\partial R}{\partial x_o})\ n_2 + (\frac{\partial Q}{\partial x_o} - \frac{\partial P}{\partial y_o})\ n_3\right] d\ S \qquad (A.15)$$

where

$$\vec{\mathbf{n}} = n_1\ \hat{\mathbf{i}} + n_2\ \hat{\mathbf{j}} + n_3\ \hat{\mathbf{k}}$$

Apply the Stokes' theorem to each component of Equation (A.14) so that

$$\begin{aligned}
\vec{\mathbf{V}}_\Gamma = \hat{\mathbf{i}} \int_S\int \{ & -[(y-y_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial y_o} + 3\mu\frac{y-y_o}{r^5}) - \frac{\mu}{r^3} \\
& + (z-z_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial z_o} + 3\mu\frac{z-z_o}{r^5}) - \frac{\mu}{r^3}]\ n_1 \\
& + (y-y_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial x_o} + 3\mu\frac{x-x_o}{r^5})\ n_2 \\
& + (z-z_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial x_o} + 3\mu\frac{x-x_o}{r^5})\ n_3\}\ d\ S \\
+ \hat{\mathbf{j}} \int_S\int \{ & (x-x_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial y_o} + 3\mu\frac{y-y_o}{r^5})\ n_1 \\
& - [(z-z_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial z_o} + 3\mu\frac{z-z_o}{r^5}) - \frac{\mu}{r^3} \\
& + (x-x_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial x_o} + 3\mu\frac{x-x_o}{r^5}) - \frac{\mu}{r^3}]\ n_2 \\
& + (z-z_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial y_o} + 3\mu\frac{y-y_o}{r^5})\ n_3\}\ d\ S \\
+ \hat{\mathbf{k}} \int_S\int \{ & (x-x_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial x_o} + 3\mu\frac{z-z_o}{r^5})\ n_1 \\
& + (y-y_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial z_o} + 3\mu\frac{z-z_o}{r^5})\ n_2 \\
& - [(x-x_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial x_o} + 3\mu\frac{x-x_o}{r^5}) - \frac{\mu}{r^3} \\
& + (y-y_o)\,(\frac{1}{r^3}\frac{\partial \mu}{\partial y_o} + 3\mu\frac{y-y_o}{r^5}) - \frac{\mu}{r^3}]n_3\ \}\ d\ S
\end{aligned}$$

$$(A.16)$$

After lengthy algebraic manipulations, Equation (A.16) can be expressed as

$$\vec{\mathbf{V}}_\Gamma = -\int_S\!\!\int \left[\frac{\mu}{r^3}\ \vec{\mathbf{n}} + 3\frac{(\vec{\mathbf{n}}\cdot\vec{\mathbf{r}})}{r^5}\vec{\mathbf{r}}\ \mu\right] d\ S$$

$$-\int_S\!\!\int \frac{1}{r^3}\ [(\vec{\mathbf{r}}\cdot\nabla_{x_o}\mu)\vec{\mathbf{n}} - (\vec{\mathbf{n}}\cdot\vec{\mathbf{r}})\nabla_{x_o}\mu]\ d\ S \qquad (A.17)$$

Finally, from Equations (A.8) and (A.12), the following equality is proven,

$$\vec{\mathbf{V}}_\Gamma = \vec{\mathbf{V}}_d - \vec{\mathbf{V}}_\gamma \qquad (A.18).$$

If S is a closed surface, such as an airfoil, then $\Gamma = 0$ and $\vec{\mathbf{V}}_\Gamma = 0$ hence the velocity due to a vortex sheet γ is equal to that of a doublet distribution, μ.

APPENDIX B

THE SINGULARITY TREATMENT AND LOCATION OF CONTROL POINT

In this Appendix the treatment of the singularity for a two-dimensional source and vortex is described along with the location of the panel control point for an accurate solution.

(1) *The Singularity*

The singularity problem arises when $\mid \vec{\mathbf{r}}(p,q) \mid \rightarrow 0$ resulting in $\ln(r(p,q))$ or Θ, or alternately stated the value of either $\mathbf{I}^n_{ij}$ or $\mathbf{D}^n_{ij}$, becoming indefinite for $i = j$. The proper value of these coefficients can be determined by a careful integration as follows.

Consider $\mathbf{I}^n_{ij}$ for $i = j$, i.e. the matrix coefficient for the, u, perturbed velocity normal to the ith panel due to the source distribution σ_i on the panel itself. Let

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (z_i - z_j)^2} \qquad (B.1)$$

and

$$\begin{aligned} \vec{\mathbf{n}}_i &= cos(\beta_i)\ \vec{\mathbf{i}} + sin(\beta_i)\ \vec{\mathbf{k}} \\ &= -sin(\theta_i)\ \vec{\mathbf{i}} + cos(\theta_i)\ \vec{\mathbf{k}} \end{aligned} \qquad (B.2)$$

where β and θ are defined in Figure 3.7 page 60. Since (x_i, z_i) and (x_j, z_j) are both on panel i, they can be expressed as (see Figure B.1 (a))

$$\begin{pmatrix} \bar{x}_i \\ \bar{z}_i \end{pmatrix} = \begin{pmatrix} s_i \\ 0 \end{pmatrix} \qquad (B.3)$$

and

$$\begin{pmatrix} \bar{x}_j \\ \bar{z}_j \end{pmatrix} = \begin{pmatrix} s_j \\ 0 \end{pmatrix} \qquad (B.4)$$

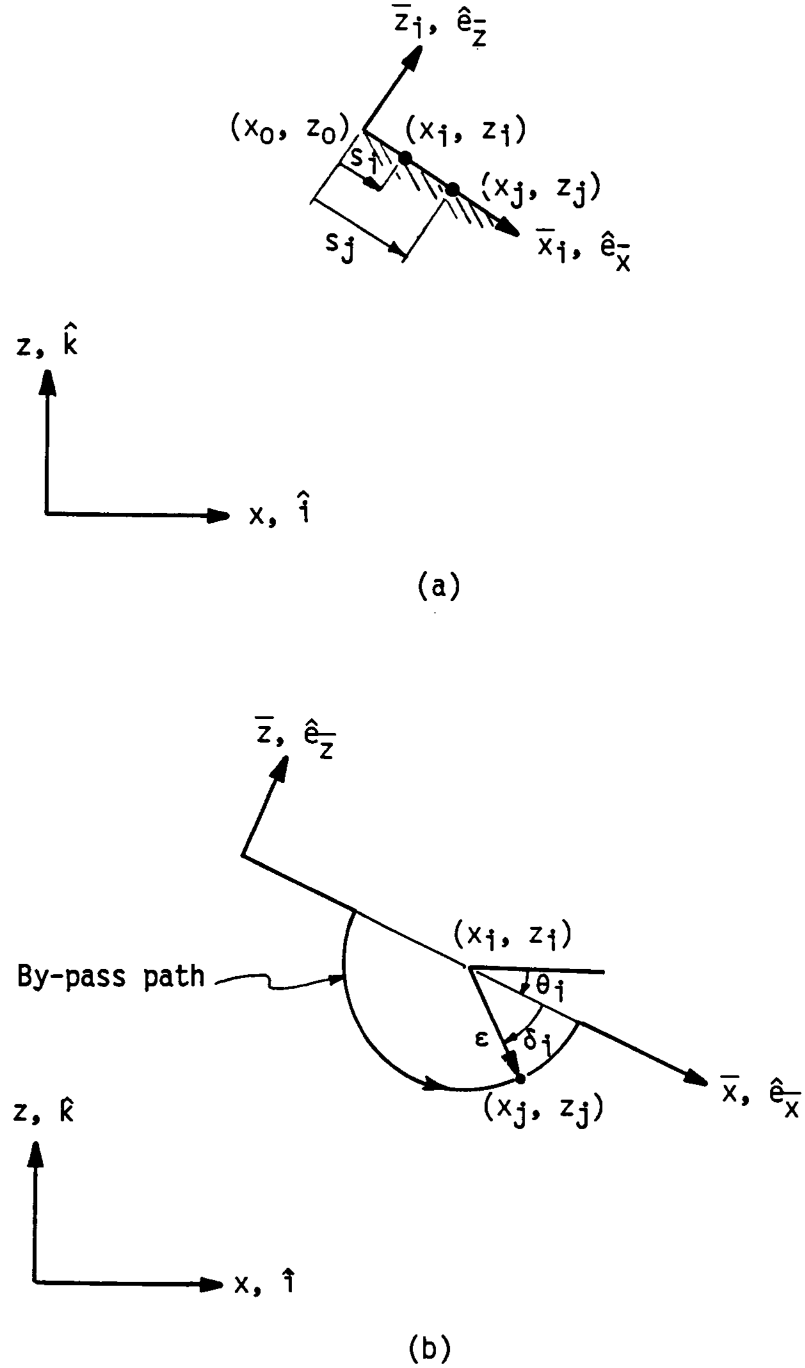


Figure B.1. Illustration of the path along the panel and the by-pass integration curve.

where $(\bar{x}_i, \bar{z}_i)$ and $(\bar{x}_j, \bar{z}_j)$ are transformed coordinates with respect to the local coordinate on the ith panel and s_j is the dummy variable of integration (for example, Equation (3.38)). The transformed unit vectors for $(\hat{\mathbf{i}}, \hat{\mathbf{k}})$ and the local normal $\vec{\mathbf{n}}_i$ are as follows:

$$\hat{\mathbf{i}} = cos(\theta_i)\ \hat{\mathbf{e}}_{\bar{x}} - sin(\theta_i)\ \hat{\mathbf{e}}_{\bar{z}} \tag{B.5a}$$

$$\hat{\mathbf{k}} = sin(\theta_i)\ \hat{\mathbf{e}}_{\bar{x}} + cos(\theta_i)\ \hat{\mathbf{e}}_{\bar{z}} \tag{B.5b}$$

$$\vec{\mathbf{n}}_i = \hat{\mathbf{e}}_{\bar{z}} \tag{B.5c}$$

The integrand of the source distribution can be transformed as follows:

$$\begin{aligned}
\frac{\partial}{\partial n_i}(\ \ln(r_{ij})\ ) &= [\ \hat{\mathbf{i}}\frac{x_i - x_j}{(x_i - x_j)^2 + (z_i - z_j)^2} + \hat{\mathbf{k}}\frac{z_i - z_j}{(x_i - x_j)^2 + (z_i - z_j)^2}\ ] \\
&\quad \cdot (\ -sin(\theta_i)\ \hat{\mathbf{i}} + cos(\theta_i)\hat{\mathbf{k}}\ ) \\
&= [\ (cos(\theta_i)\hat{e}_{\bar{x}} - sin(\theta_i)\hat{e}_{\bar{z}})\ \frac{(s_i - s_j)cos(\theta_i)}{(s_i - s_j)^2} + \\
&\quad (sin(\theta_i)\ \hat{e}_{\bar{x}} + cos(\theta_i)\hat{e}_{\bar{z}})\ \frac{(s_i - s_j)sin(\theta_i)}{(s_i - s_j)^2}\ ] \cdot \hat{\mathbf{e}}_{\bar{z}} \\
&= (\ \hat{\mathbf{e}}_{\bar{x}}\ \frac{1}{s_i - s_j} + \hat{\mathbf{e}}_{\bar{z}}\ 0\ ) \cdot \hat{\mathbf{e}}_{\bar{z}} \\
&= 0
\end{aligned} \tag{B.6}$$

for any point on panel i except $s_j = s_i$. This means that the source distribution for $s_i \neq s_j$ on the same panel i has no contribution on the perturbed velocity normal to the panel i nor the singularity problem. The integration for the point $s_i = s_j$ can be treated as follows. The singularity is excluded by integrating along a infinitely small by-pass as shown in Figure B.1 (b) such that

$$\begin{pmatrix} x_i \\ z_i \end{pmatrix} = \begin{pmatrix} x_o \\ z_o \end{pmatrix} + s_i \begin{pmatrix} cos(\theta_i) \\ sin(\theta_i) \end{pmatrix} \tag{B.7}$$

$$\begin{pmatrix} x_j \\ z_j \end{pmatrix} = \begin{pmatrix} x_i \\ z_i \end{pmatrix} + \epsilon \begin{pmatrix} cos(\delta_i + \theta_i) \\ sin(\delta_i + \theta_i) \end{pmatrix} \tag{B.8}$$

Therefore from Equation B.3

$$\begin{pmatrix} x_i - x_j \\ z_i - z_j \end{pmatrix} = -\epsilon \begin{pmatrix} cos(\delta_i + \theta_i) \\ sin(\delta_i + \theta_i) \end{pmatrix} \tag{B.9}$$

Substitute Equation (B.9) into the relationship

$$\begin{aligned}
\frac{\partial}{\partial n_i}(\ \ln(r_{ij})\ ) &= \frac{\partial}{\partial r}(\ \ln(r_{ij})\ ) \\
&= [\ \hat{\mathbf{i}}\frac{x_i - x_j}{(x_i - x_j)^2 + (z_i - z_j)^2} + \hat{\mathbf{k}}\frac{z_i - z_j}{(x_i - x_j)^2 + (z_i - z_j)^2}\ ] \cdot\ \hat{\mathbf{e}}_r \\
&= [\ (cos(\theta_i)\hat{\mathbf{e}}_{\bar{x}} - sin(\theta_i)\hat{\mathbf{e}}_{\bar{z}})\ \frac{-\epsilon cos(\delta_i + \theta_i)}{(\epsilon cos(\delta_i + \theta_i))^2 + (\epsilon sin(\delta_i + \theta_i))^2} + \\
&\quad (sin(\theta_i)\hat{\mathbf{e}}_{\bar{x}} + cos(\theta_i)\hat{\mathbf{e}}_{\bar{z}})\ \frac{-\epsilon sin(\delta_i + \theta_i)}{(\epsilon cos(\delta_i + \theta_i))^2 + (\epsilon sin(\delta_i + \theta_i))^2}] \cdot \hat{\mathbf{e}}_r \\
&= (\ \hat{\mathbf{e}}_{\bar{x}}\ \frac{-cos(\delta_i)}{\epsilon} + \hat{\mathbf{e}}_{\bar{z}}\ \frac{-sin(\delta_i)}{\epsilon}) \cdot -(\ cos(\delta_i)\ \hat{\mathbf{e}}_{\bar{x}} + sin(\delta_i)\ \hat{\mathbf{e}}_{\bar{z}} \\
&= \frac{1}{\epsilon}
\end{aligned} \tag{B.10}$$

Therefore

$$\begin{aligned}
\mathbf{I}^n_{ii} &= \int_{-\pi}^{0} \frac{\partial}{\partial r_i}(\ \ln(r_{ii})\ d\ s_i \\
&= \int_{-\pi}^{0} (\frac{1}{\epsilon})\ \epsilon\ d\ \delta_i \\
&= \pi
\end{aligned} \tag{B.11}$$

The singularity treatment for the vortex distribution is similar but easier. The integrand of $\mathbf{D}^n_{ii}$ can be transformed as follows:

$$\begin{aligned}
\frac{\partial}{\partial n_i}[tan^{-1}(\ \frac{z_i - z_j}{x_i - x_j}\ )] &= [\hat{\mathbf{i}}\frac{-(z_i - z_j)}{(x_i - x_j)^2 + (z_i - z_j)^2} + \hat{\mathbf{k}}\frac{x_i - x_j}{(x_i - x_j)^2 + (z_i - z_j)^2}] \cdot \vec{\mathbf{n}}_i \\
&= [\ (cos(\theta_i)\hat{\mathbf{e}}_{\bar{x}} - sin(\theta_i)\hat{\mathbf{e}}_{\bar{z}})\ \frac{-(s_i - s_j)sin(\theta_i)}{(s_i - s_j)^2} + \\
&\quad (sin(\theta_i)\ \hat{\mathbf{e}}_{\bar{x}} + cos(\theta_i)\hat{\mathbf{e}}_{\bar{z}})\ \frac{(s_i - s_j)cos(\theta_i)}{(s_i - s_j)^2}] \cdot \hat{\mathbf{e}}_{\bar{z}} \\
&= \frac{1}{(s_i - s_j)}
\end{aligned} \tag{B.12}$$

By Cauchy principal value theorem $\mathbf{D}_{ii}^n$ can be expressed as

$$\mathbf{D}_{ii}^n = \int_0^{l_i} \frac{ds_i}{s_i - s_j}$$
$$= -\ln\left(\left| \frac{s_i}{s_i - l_i} \right| \right) \qquad (B.13)$$

where l_i is the panel length for ith panel. It may be noted that for $s_i = l_i/2$, $\mathbf{D}_{ii}^n$ equals zero.

(2) *Control Point Location*

From the discussion in the preceding, it is obvious that the location of the control point should not be chosen at the center of each panel. The standard analysis for the location of the control point for a single vortex located on a single panel [51] (as in the vortex lattice method) is extendible to a continuous distribution of vorticity along that panel. The Joukowski theorem will give

$$C_L = \frac{2}{V_\infty l} \int_0^l \gamma(s) ds \equiv 2\pi\alpha \qquad (B.14)$$

the second equality following from slender body theory. Hence

$$\pi V_\infty l \alpha = \int_0^l \gamma(s) ds$$
$$\equiv \gamma l \qquad (B.15)$$

where γ is constant over each panel.

To satisfy the boundary condition of no normal velocity at point s on the panel, small disturbance theory gives:*

$$V_\infty \alpha = \frac{1}{2\pi} \int_0^l \frac{\gamma(s')ds'}{(s - s')}$$
$$\equiv \gamma/\pi \qquad (B.16)$$

* If $\gamma(s)$ is not a constant then

$$\int_{s-l}^{s} \left(\frac{2}{l} - \frac{1}{s'}\right)\gamma(s')ds' = 0$$

is the equivalent condition.

from Equation B.15. Hence by integration :

$$\ln\left|\frac{s}{s-l}\right| = 2 \tag{B.17}$$

but as $0 \leq s \leq l$:

$$\ln\left(\frac{s}{l-s}\right) = 2$$

or

$$\frac{s}{l} = \frac{\exp(2)}{1+\exp(2)} \tag{B.18}$$

Hence the requirement $s/l \approx 0.88$. It may be noted from Equation (B.17) that for $s/l < 0.5$ there is *no* solution since $\ln|s/(s-l)| < 0$, that is, the required tangency boundary condition cannot be satisfied. There is a fundamental restriction $0.5l < RCP \leq 1.0l$, where RCP is the multiplicative factor (used in the code described in Chapter IV - see also Appendix D) on the panel length l to specify the location of the control point on the panel. Figure B.2 depicts the influence of different choices for the location of the control point. The pressure distribution for $RCP = 0.50l$ is a bad solution because Equation (B.17) cannot be satisfied. The pressure distribution for $RCP = 0.60l$ is almost identical to that for $RCP = 0.75l$. Figure B.3 compares calculations for $RCP = 0.6l$ and for the estimate $RCP = 0.88l$. Again, there is little difference except near the leading-edge (and this may be due to an insufficient number of panels). It should be noted that this estimate, Equation (B.18), is for a single panel. Loss of sensitivity of the precise location of the control point is evident for the large number of panels used for the calculations herein. $RCP = 0.6l$ is chosen for the present study.

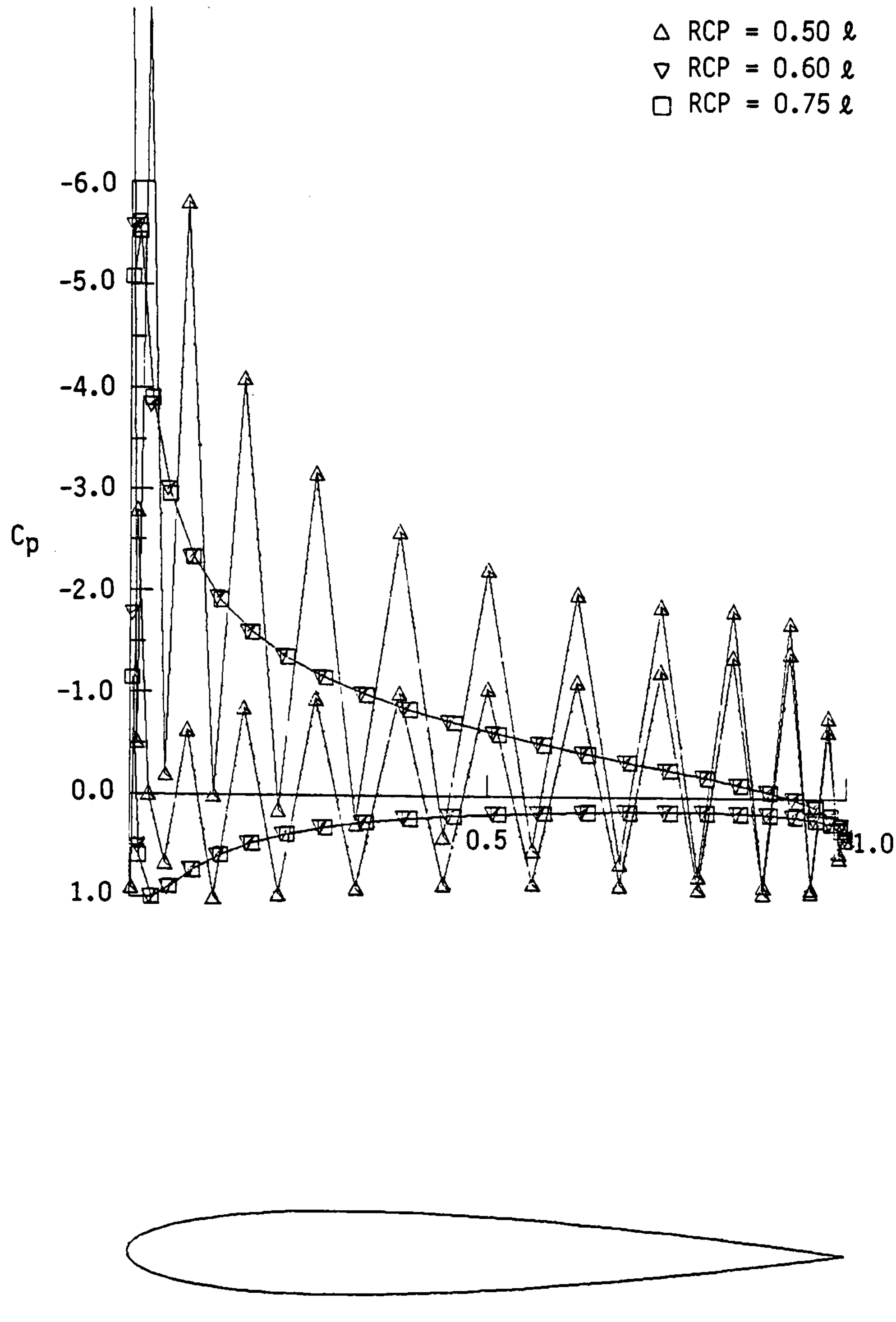


Figure B.2. Influence of different choices of the location of the control point on NACA 0012 at 10 deg. angle of attack.

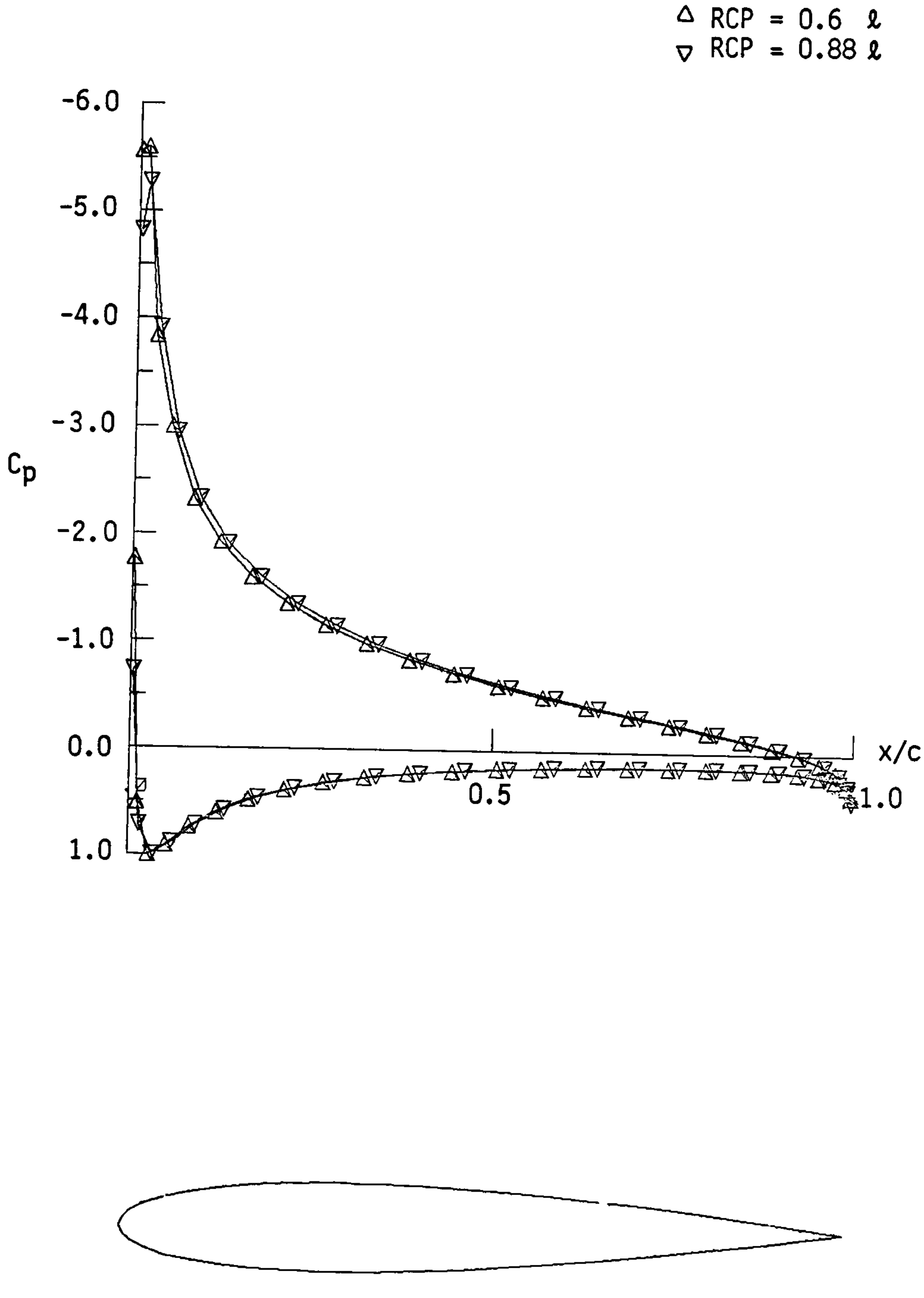


Figure B.3. Comparison of the C_p distribution for $RCP = 0.6l$ and $RCP = 0.88l$ on NACA 0012 at 10 deg. angle of attack.

APPENDIX C

THE RELATIONSHIP BETWEEN MATRIX COEFFICIENTS OF THE SOURCE AND VORTEX STRENGTHS

The elements of $\mathbf{I}^n_{ij}$ and $\mathbf{D}^n_{ij}$ in Equations (3.38) and (3.41) are the perturbed velocities normal to each panel i due to the source σ_j and the vortex γ_j respectively. Now let the elements of $\mathbf{I}^t_{ij}$ and $\mathbf{D}^t_{ij}$ be the perturbed velocities tangential to each panel i and expressed as follows:

$$\begin{aligned}
\mathbf{I}^t_{ij} &= \int_j \nabla_i(\log r_{ij}) \cdot \mathbf{t}_i \, ds_j \\
&= \int_j \frac{\partial}{\partial t_i}(\log r_{ij}) \, ds_j \\
&= \int_j \frac{\bar{x}_i cos(\theta_j - \theta_i) - \bar{z}_i sin(\theta_j - \theta_i) - s_j cos(\theta_j - \theta_i)}{(\bar{x}_i^2 + \bar{z}_i^2) - 2\bar{x}_i s_j + s_j^2} \, ds_j \\
&= \int_j \frac{d + es_j}{a + bs_j + cs_j^2} \, ds_j \qquad (i \neq j) \qquad\qquad (C.1)
\end{aligned}$$

In Equation (C.1), $\bar{x}_i$ and $\bar{z}_i$ are transformed in terms of local coordinates (see Equation (3.27)), and θ_i and θ_j are defined as in Figure 3.9 page 64. The definitions of the quantities a through e are

$$\begin{aligned}
a &= \bar{x}_i^2 + \bar{z}_i^2 \\
b &= -2\bar{x}_i \\
c &= 1 \\
d &= \bar{x}_i cos(\theta_j - \theta_i) - \bar{z}_i sin(\theta_j - \theta_i) \\
e &= -cos(\theta_j - \theta_i) \qquad\qquad (C.2)
\end{aligned}$$

Also

$$\begin{aligned}
R &= a + bs_j + cs_j^2 \\
\Delta &= 4ac - b^2
\end{aligned}$$

For $\Delta > 0$,

$$\begin{aligned}
\mathbf{I}^t_{ij} &= \frac{e}{2c}[\log(R)]_0^{l_j} + (d - \frac{be}{2c})\frac{2}{\sqrt{\Delta}}tan^{-1}[\frac{b+2cs_j}{\sqrt{\Delta}}]_0^{l_j} \\
&= \frac{-1}{2}cos(\theta_j - \theta_i)\log\left(1 + \frac{l_j^2 - 2\bar{x}_i l_j}{\bar{x}_i^2 + \bar{z}_i^2}\right) \\
&- sin(\theta_j - \theta_i)\left[tan^{-1}(\frac{l_j - \bar{x}_i}{\bar{z}_i}) - tan^{-1}(\frac{-\bar{x}_i}{\bar{z}_i})\right] \quad (i \neq j) \qquad (C.3a)
\end{aligned}$$

and for $\Delta = 0$,

$$\begin{aligned}
\mathbf{I}^t_{ij} &= \frac{e}{2c}[\log(R)]_0^{l_j} + (d - \frac{be}{2c})[\frac{-2}{b+2cs_j}]_0^{l_j} \\
&= \frac{-1}{2}cos(\theta_j - \theta_i)\log\left(1 + \frac{l_j^2 - 2\bar{x}_i l_j}{\bar{x}_i^2 + \bar{z}_i^2}\right) \quad (i \neq j) \qquad (C.3b)
\end{aligned}$$

where l_j is the panel length for the jth panel. The singularity treatment for $i = j$ can done as Appendix B.

Similarly, for the influence matrix $\mathbf{D}^t_{ij}$,

$$\begin{aligned}
\mathbf{D}^t_{ij} &= -\int_j \nabla_i(\Theta_{ij}) \cdot \mathbf{t}_i \, ds_j \\
&= -\int_j \nabla_i \left[tan^{-1}(\frac{z_i - z_j}{x_i - x_j})\right] \cdot \mathbf{t}_i \, ds_j \\
&= -\int_j \frac{\partial}{\partial t_i}\left[tan^{-1}(\frac{z_i - z_j}{x_i - x_j})\right] ds_j \\
&= \int_j \frac{\bar{x}_i sin(\theta_j - \theta_i) + \bar{z}_i cos(\theta_j - \theta_i) - s_j sin(\theta_j - \theta_i)}{(\bar{x}_i^2 + \bar{z}_i^2) - 2\bar{x}_i s_j + s_j^2} \, ds_j \\
&= -\int_j \frac{d + es_j}{a + bs_j + cs_j^2} \, ds_j \qquad (i \neq j) \qquad (C.4)
\end{aligned}$$

where

$$\begin{aligned}
a &= \bar{x}_i^2 + \bar{z}_i^2 \\
b &= -2\bar{x}_i \\
c &= 1 \\
d &= -\bar{x}_i sin(\theta_j - \theta_i) - \bar{z}_i cos(\theta_j - \theta_i) \\
e &= sin(\theta_j - \theta_i) \qquad (C.5)
\end{aligned}$$

For $\Delta > 0$,

$$\begin{aligned}\mathbf{D}^t_{ij} &= -\frac{e}{2c}[\log(R)]_0^{l_j} - (d - \frac{be}{2c})\frac{2}{\sqrt{\Delta}}tan^{-1}[\frac{b+2cs_j}{\sqrt{\Delta}}]_0^{l_j} \\ &= \frac{-1}{2}sin(\theta_j - \theta_i)\log\left(1 + \frac{l_j^2 - 2\bar{x}_i l_j}{\bar{x}_i^2 + \bar{z}_i^2}\right) \\ &+ cos(\theta_j - \theta_i)\left[tan^{-1}(\frac{l_j - \bar{x}_i}{\bar{z}_i}) - tan^{-1}(\frac{-\bar{x}_i}{\bar{z}_i})\right] \quad (i \neq j) \qquad (C.6a)\end{aligned}$$

and for $\Delta = 0$,

$$\begin{aligned}\mathbf{D}^t_{ij} &= -\frac{e}{2c}[\log(R)]_0^{l_j} - (d - \frac{be}{2c})[\frac{-2}{b+2cs_j}]_0^{l_j} \\ &= \frac{-1}{2}sin(\theta_j - \theta_i)\log\left(1 + \frac{l_j^2 - 2\bar{x}_i l_j}{\bar{x}_i^2 + \bar{z}_i^2}\right) \quad (i \neq j) \qquad (C.6b)\end{aligned}$$

By comparing Equation (3.40) with Equation (C.6) and Equation (3.43) with Equation (C.3) the following equalities are satisfied,

$$\begin{aligned}\mathbf{I}^t_{ij} &= -\mathbf{D}^n_{ij} \\ \mathbf{D}^t_{ij} &= \mathbf{I}^n_{ij} \qquad (C.7)\end{aligned}$$

which are equivalent to Equation (3.48).

APPENDIX D

COMPUTER PROGRAM FOR THE TWO-DIMENSIONAL FLOW

```
**********          PROGRAM IDENTIFICATION                 **********
*                                                                   *
*            PROGRAM NAME : AIRFOIL                                 *
*                                                                   *
*********************************************************************
*                                                                   *
*              WRITTEN BY : JYH-CHYANG ALEX WANG                    *
*                                                                   *
*                    DATE : APRIL 10, 1988                          *
*                                                                   *
*                    NOTE : SOLUTION OF THE PROBLEM OF UNIFORM OR   *
*                           SHEAR FLOW W. OR WO. ANGLE OF ATTACK    *
*                           PAST AN AIRFOIL BY " NEW SCHEME "       *
*                           SOURCE & VORTEX PANELS + ITERATIONS     *
*                           VARIABLE GRIDS AND PANEL # ( EVEN )     *
*                           VORTICITY TRANSPORT IMPLEMENTED         *
*                                                                   *
*                   USAGE : (1) BREAK COMMON.FIL INTO SUB-FILES     *
*                                                                   *
*                           (2) FOR/CH FILENAME                     *
*                                                                   *
*                           (3) LINK FILENAME,LIB_IMSL_S/LIB        *
*                                                                   *
*                           (4) RUN FILENAME                        *
*                                                                   *
*                           (5) READ INPUT FILE, AIRFOIL.DAT        *
*                                                                   *
*                           (6) WRITE OUTPUT FILE, AIRFOIL.OUT      *
*                                                                   *
*                           (7) DATA1.OUT, DATA FOR AIRFOIL PLOTTING *
*                                                                   *
*                           (8) DATA3.OUT, DATA FOR CP PLOTTING     *
*                                                                   *
********************     NOMENCLATURE     ***************************
*                                                                   *
*     IDBL   = 0,     W/0. CROSS FLOW                               *
*              1,     W. UNIFORM CROSS FLOW                         *
*              2,     W. LINEAR SHEAR CROSS FLOW                    *
*              3,     W. EXPONENTIAL SHEAR CROSS FLOW               *
*                                                                   *
*     IDIR   = 0,     UPDRAFT CROSS FLOW                            *
*              1,     DOWNDRAFT CROSS FLOW                          *
*                                                                   *
*     IOUT   = 0,     NO OUTPUT                                     *
*              1,     COMPACT OUTPUT                                *
*              2      FULL OUTPUT                                   *
*                                                                   *
*     ALPHA           ANGLE OF ATTACK                               *
*                                                                   *
*     ZETA0           ONSET VORTICITY                               *
*                                                                   *
*     UY0             REFERENCE VELOCITY                            *
*                                                                   *
*********************************************************************
*                                                                   *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' AUXI.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' DBLE.FIL '
      INCLUDE ' DFWS.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' JOUK.FIL '
```

```
      INCLUDE ' MTRX.FIL '
      INCLUDE ' NAC4.FIL '
      INCLUDE ' STFN.FIL '
      INCLUDE ' VLTY.FIL '

      REAL AA( NPAN )

      NAMELIST / INPUT1 / IFLOW, IDBL, IAIR, IOUT, ALPHA, ZETA0,
     1                    UY0, S, IMAX, ERROR, RCP, IDIR
      NAMELIST / INPUT2 / AM, AP, AT
      NAMELIST / INPUT3 / AREF, CREF
      NAMELIST / INPUT4 / AREF1, CREF1, ZETA1, VX0
      NAMELIST / INPUT5 / YREF
      NAMELIST / INPUT6 / XTOT, YTOT, UCN, VCN, VREF
      NAMELIST / INPUT7 / UCN1, VCN1, VREF1

      OPEN( UNIT =  10, NAME = '        AIRFOIL.DAT ', TYPE = 'OLD' )
      OPEN( UNIT =  11, NAME = '        AIRFOIL.OUT ', TYPE = 'NEW' )
      OPEN( UNIT = 111, NAME = '          DATA1.OUT ', TYPE = 'NEW' )
      OPEN( UNIT = 113, NAME = '          DATA3.OUT ', TYPE = 'NEW' )

*23456789-123456789-123456789-123456789-123456789-123456789-123456789-12
*                                                                      *
********************    INITIALIZATION    ******************************
*                                                                      *
*                    / CONTROL PARAMETER  /                            *
*                                                                      *
*                  IFLOW = 1,    UNIFORM FLOW                          *
*                          2,    LINEAR SHEAR FLOW                     *
*                          3,    EXPONENTIAL SHEAR FLOW( NONLINEAR )   *
*                          4,    INPUT JAWS DATA( NONLINEAR )          *
*                          5     INPUT DFW DATA ( TASS AXISY. WIND )   *
*                                                                      *
*                   IDBL = 0,    W/O. CROSS FLOW                       *
*                          1,    W. UNIFORM CROSS FLOW                 *
*                          2,    W. LINEAR SHEAR CROSS FLOW            *
*                          3,    W. EXPONENTIAL SHEAR CROSS FLOW       *
*                          4,    W. COSINE SHEAR CROSS FLOW            *
*                          5,    W. SINE SHEAR CROSS FLOW              *
*                                                                      *
*                   IDIR = 0,    UPDRAFT CROSS FLOW                    *
*                          1,    DOWNDRAFT CROSS FLOW                  *
*                                                                      *
*                   IAIR = 1,    NACA 0012                             *
*                          2,    NACA 2412                             *
*                          3,    NACA 4 DIGIT( INPUT AM, AP, T )       *
*                          4,    SYM. JOUKOWSKI(-9.2376D-2,0,1.092376)*
*                          5,    CAMBERED JOUKOWSKI( -0.05,0.05,0.5 ) *
*                                                                      *
*                   IOUT = 0,    NO OUTPUT                             *
*                          1,    COMPACT OUTPUT                        *
*                          2     FULL OUTPUT                           *
*                                                                      *
      PI    = ACOS( -1.0 )
      READ ( 10, INPUT1 )
      WRITE( 11, INPUT1 )

      IF( IAIR .EQ. 3 )THEN
      READ ( 10, INPUT2 )
      WRITE( 11, INPUT2 )
      END IF

      IF( IFLOW .EQ. 3 )THEN
```

```
      READ ( 10, INPUT3 )
      WRITE( 11, INPUT3 )

      ELSEIF( IFLOW .EQ. 4 )THEN
      READ ( 10, INPUT6 )
      WRITE( 11, INPUT6 )
      CALL SORT( UCN, VCN, VREF )

      ELSEIF( IFLOW .EQ. 5 )THEN
      READ ( 10, INPUT7 )
      WRITE( 11, INPUT7 )
      CALL SORT( UCN1, VCN1, VREF1 )
      END IF

      IF( IDBL .NE. 0 )THEN
      READ ( 10, INPUT4 )
      WRITE( 11, INPUT4 )
      END IF

      WRITE( 11, 50 ) NPAN, NGRID
50    FORMAT( 1X, 'NPAN = ', I5, /, 1X, 'NGRID = ', I5 )

**********************************************************************
*     CONSTRUCT INITIAL CONDITIONS                                   *

      ALPHA  = ALPHA * PI / 180.0
      NPANH  = NPAN / 2
      NPANH2 = NPANH + 2

      CALL SHAPE
      CALL BDYCORD
      CALL PANEL
      CALL CNTCORD
      CALL GRDCORD
      CALL ONSET

      CLOSE( UNIT = 10 )

      CALL TRACORD
      CALL COEF

      CALL INVERSE( A1STORE, AINVER )
      CALL INVERSE( D1STORE, DINVER )

      CALL RHS1
      CALL LEQINV( AINVER, RHSC1, ASIGAMA )

10    ICONT = ICONT + 1

      DO I = 1, NPAN
      AA( I ) = AGAMA( I )
      END DO

      CALL RHS2( ICONT )
      CALL LEQINV( DINVER, RHSC2, AGAMA )

      CALL VELFLD( ICONT )


      IF( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) )THEN
      CALL STREAM( ICONT )
      CALL VORFLD
```

```
      END IF

      IF( ( IFLOW .LE. 2 ) .AND. ( IDBL .LE. 2 ) ) THEN
      GOTO 100

      ELSEIF( ICONT .GE. IMAX )THEN
      TYPE*,'ITERATIONS OVER IMAX', ICONT
      GOTO 100

      ELSEIF( ICONT .EQ. 1 )THEN
      GOTO 10

      ELSE
      CALL ERRCHK( AA, AGAMA, ERMAX )
      TYPE*,'ICONT,ERMAX=',ICONT,ERMAX

      END IF

      IF( ERMAX .GT. ERROR )THEN
      GOTO 10
      END IF

      TYPE*,'ITERATION CONVERGES AT',ICONT

100   CALL PRESUR
      CALL AEROFR
      CALL OUTPUT

      STOP
      END

      SUBROUTINE VELFLD( ICONT )
************************************************************************
*                                                                      *
*     CALCULATE THE PRESENT VELOCITY FIELD : ON THE SURFACE OF THE     *
*     AIRFOIL AND IN THE COMPUTATIONAL DOMAIN                          *
*                                                                      *
********************      NOMENCLATURE      ****************************
*                                                                      *
*     VS              TANGENTIAL SPEED ON THE CONTROL POINT            *
*                                                                      *
*     U, V            VELOCITY FIELD ON THE CONTROL POINTS             *
*                                                                      *
*     UG, VG          VELOCITY FIELD IN THE INNER DOMAIN               *
*                                                                      *
*     UOUT, VOUT      VELOCITY FIELD IN THE OUTER DOMAIN               *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' MTRX.FIL '
      INCLUDE ' VLTY.FIL '

      DO I = 1, NPAN
      SUM1 = 0.0
      SUM2 = 0.0

        DO J = 1, NPAN
        SUM1 = SUM1 + AINTGL2( I, J ) * ASIGAMA( J )
        SUM2 = SUM2 + DINTGL2( I, J ) * AGAMA( J )
        END DO
```

```
          IF( ( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) .AND.
     1            ( ICONT .GT. 1 ) )THEN
          VS( I ) = U0( I ) * COSINE( I ) + V0( I ) * SINE( I ) +
     1                SUM1 + SUM2 + RESLT( I )
          ELSE
          VS( I ) = U0( I ) * COSINE( I ) + V0( I ) * SINE( I ) +
     1                SUM1 + SUM2
          END IF

          U( I ) = VS( I ) * COSINE( I )
          V( I ) = VS( I ) * SINE( I )

        END DO

        IF( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) )THEN

        DO I = 1, NGRID
        DO J = 1, NGRID
        SUMU = 0.0
        SUMV = 0.0

          DO K = 1, NPAN
          SUMU = SUMU + AGINTGL2( I, J, K ) * ASIGAMA( K ) +
     1           DGINTGL2( I, J, K ) * AGAMA( K )
          SUMV = SUMV + AGINTGL1( I, J, K ) * ASIGAMA( K ) +
     1           DGINTGL1( I, J, K ) * AGAMA( K )
          END DO

        IF( ( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) .AND.
     1        ( ICONT .GT. 1 ) )THEN
        UG( I, J ) = UG0( I, J ) + SUMU + RESLU( XG(I,J), YG(I,J) )
        VG( I, J ) = VG0( I, J ) + SUMV + RESLV( XG(I,J), YG(I,J) )
        ELSE
        UG( I, J ) = UG0( I, J ) + SUMU
        VG( I, J ) = VG0( I, J ) + SUMV
        END IF

        END DO
        END DO

        END IF

        RETURN
        END


        SUBROUTINE VORFLD
************************************************************************
*                                                                      *
*       CALCULATE THE PRESENT VORTICITY FIELD IN THE DOMAIN AND AT     *
*       FAR UPSTREAM: OMEGAG AND OMEG01                                *
*                                                                      *
********************     NOMENCLATURE     *****************************
*                                                                      *
*       OMEGAG          VORTICITY FIELD IN THE DOMAIN                  *
*                                                                      *
*       OMEG01          VORTICITY FIELD AT FAR UPSTREAM                *
*                                                                      *
************************************************************************
*                                                                      *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' STFN.FIL '
```

```
      INCLUDE ' STRM.FIL '
      INCLUDE ' VOTY.FIL '
      REAL C( NG100, 3 )

      DO I = 1, NGRID
      DO J = 1, NGRID

      DO K = 1, NG101-1
        K1 = K + 1

      IF( PSI1( K ) .EQ. PSIG( I, J ) )THEN
      OMEGAG( I, J ) = OMEG01( K )
      GOTO 100

      ELSEIF( ( PSI1( K  ) .LE. PSIG( I,J ) ) .AND.
     1        ( PSI1( K1 ) .GT. PSIG( I,J ) ) )THEN
         N = NG101
        IC = N - 1
         D = PSIG( I, J ) - PSI1( K )
        CALL ICSCCU( PSI1, OMEG01, N, C, IC, IER )
        OMEGAG(I,J) = ( ( C(K,3) * D + C(K,2) ) * D + C(K,1) ) * D +
     1                 OMEG01( K )
      GOTO 100

      END IF

      END DO

      DO K = 1, NG101-1
        K1 = K + 1

      IF( PSI2( K ) .EQ. PSIG( I, J ) )THEN
      OMEGAG( I, J ) = OMEG02( K )
      GOTO 100

      ELSEIF( ( PSI2( K  ) .LE. PSIG( I,J ) ) .AND.
     1        ( PSI2( K1 ) .GT. PSIG( I,J ) ) )THEN
         N = NG101
        IC = N - 1
         D = PSIG( I, J ) - PSI2( K )
        CALL ICSCCU( PSI2, OMEG02, N, C, IC, IER )
        OMEGAG(I,J) = ( ( C(K,3) * D + C(K,2) ) * D + C(K,1) ) * D +
     1                 OMEG02( K )
      GOTO 100

      END IF

      END DO

100   CONTINUE

      END DO
      END DO

      RETURN
      END

      SUBROUTINE ONESD( FX, L, DH, OMEG )
*********************************************************************
*                                                                   *
*     1st DERIVATIVE BY 4th ORDER ONE-SIDED DIFFERENCING (DOWNSTREAM)*
*                                                                   *
*******************     NOMENCLATURE     ****************************
*                                                                   *
*     FX              FUNCTION ARRAY                                *
*                                                                   *
```

```
*       L                GRID INDEX                                         *
*                                                                           *
*       DH               GRID SIZE                                          *
*                                                                           *
*       OMEG             FIRST DERIVATIVE                                   *
*                                                                           *
*****************************************************************************
*                                                                           *
        PARAMETER ( NGRID = 41 )
        REAL FX( NGRID )

        OMEG = ( -25.0 * FX( L   ) + 48.0 * FX( L+1 ) -
     1            36.0 * FX( L+2 ) + 16.0 * FX( L+3 ) -
     1             3.0 * FX( L+4 ) ) / 12.0 / DH

        RETURN
        END

        SUBROUTINE ONESU( FX, L, DH, OMEG )
*****************************************************************************
*                                                                           *
*       1st DERIVATIVE BY 4th ORDER ONE-SIDED DIFFERENCING (UPSTREAM)       *
*                                                                           *
********************       NOMENCLATURE      ********************************
*                                                                           *
*       FX               FUNCTION ARRAY                                     *
*                                                                           *
*       L                GRID INDEX                                         *
*                                                                           *
*       DH               GRID SIZE                                          *
*                                                                           *
*       OMEG             FIRST DERIVATIVE                                   *
*                                                                           *
*****************************************************************************
*                                                                           *
        PARAMETER ( NGRID = 41 )
        REAL FX( NGRID )

        OMEG = - ( -25.0 * FX( L   ) + 48.0 * FX( L-1 ) -
     1              36.0 * FX( L-2 ) + 16.0 * FX( L-3 ) -
     1               3.0 * FX( L-4 ) ) / 12.0 / DH

        RETURN
        END

        SUBROUTINE CNTLD( FX, L, DH, OMEG )
*****************************************************************************
*                                                                           *
*       1st DERIVATIVE BY 4th ORDER CENTRAL DIFFERENCING                    *
*                                                                           *
********************       NOMENCLATURE      ********************************
*                                                                           *
*       FX               FUNCTION ARRAY                                     *
*                                                                           *
*       L                GRID INDEX                                         *
*                                                                           *
*       DH               GRID SIZE                                          *
*                                                                           *
*       OMEG             FIRST DERIVATIVE                                   *
*                                                                           *
*****************************************************************************
*                                                                           *
        PARAMETER ( NGRID = 41 )
        REAL FX( NGRID )
```

```
      OMEG = ( FX( L-2 ) - 8.0 * FX( L-1 ) + 8.0 *
     1         FX( L+1 ) - FX( L+2 ) ) / 12.0 / DH

      RETURN
      END

      SUBROUTINE OUTPUT
***********************************************************************
*                                                                     *
*                          OUTPUT FILE                                *
*                                                                     *
***********************************************************************
*                                                                     *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' AUXI.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' DBLE.FIL '
      INCLUDE ' JOUK.FIL '
      INCLUDE ' MTRX.FIL '
      INCLUDE ' NAC4.FIL '
      INCLUDE ' VLTY.FIL '
      INCLUDE ' VOTY.FIL '

      WRITE(113, *) ( X( I ), CP( I ), I = NPAN, 1, -1 )

      IF( ( IOUT .EQ. 1 ) .OR. ( IOUT .EQ. 2 ) )THEN
        SUM1 = 0.0
        SUM2 = 0.0
      DO 1 I = 1, NPAN
        SUM1 = SUM1 + ASIGAMA( I ) * PL( I )
        SUM2 = SUM2 + AGAMA( I ) * PL( I )
 1    CONTINUE
      WRITE(11, 50)
      WRITE(11, 52)  ( I, X(I), Y(I), XB(I), YB(I), PL(I), THETA(I),
     1                 ASIGAMA( I ), AGAMA( I ), CP( I ), I = 1, NPAN )

      IF( IAIR .GT. 3 )THEN
      WRITE(11, 57) XZ, YZ, AZ
      ELSE
      IAM = 100.0 * AM
      IAP =  10.0 * AP
      IAT = 100.0 * AT
      WRITE(11, 53) IAM, IAP, IAT
      END IF

         AD = DALPHA * 180.0 / PI
      ALPHA = 180.0 * ALPHA / PI
      WRITE(11, 54) SUM1, SUM2
      WRITE(11, 55) ALPHA, ZETA0, RCP
      WRITE(11, 56) CL, CD, AD
      WRITE(11, 58) YINTY, UINTY
      WRITE(11, 59) UY0, VX0

      WRITE( 11, 100 )
      WRITE( 11, 101 )( I, U0(I), V0(I), U(I), V(I), I = 1, NPAN )

      END IF


      IF( IOUT .EQ. 2 )THEN
      WRITE( 11, 102 )
      WRITE( 11, 103) ( ( AINTGL1( I, J ), J=1,NPAN,2 ), I=1,NPAN,2 )
```

```
      WRITE( 11, 104 )
      WRITE( 11, 103) ( ( DINTGL1( I, J ), J=1,NPAN,2 ), I=1,NPAN,2 )
      WRITE( 11, 105 )
      WRITE( 11, 103) ( ( AINTGL2( I, J ), J=1,NPAN,2 ), I=1,NPAN,2 )
      WRITE( 11, 106 )
      WRITE( 11, 103) ( ( DINTGL2( I, J ), J=1,NPAN,2 ), I=1,NPAN,2 )

      WRITE(11,201)
      WRITE(11,200) ( ( XG( I, J ), I = 1, NG21 ), J = NG21,1,-1 )
      WRITE(11,202)
      WRITE(11,200) ( ( YG( I, J ), I = 1, NG21 ), J = NG21,1,-1 )
      WRITE(11,203)
      WRITE(11,200) ( ( UG( I, J ), I = 1, NG21 ), J = NG21,1,-1 )
      WRITE(11,204)
      WRITE(11,200) ( ( VG( I, J ), I = 1, NG21 ), J = NG21,1,-1 )
      WRITE(11,205)
      WRITE(11,200) ( ( OMEGAG0( I,J ), I = 1, NG21 ), J = NG21,1,-1 )
      WRITE(11,206)
      WRITE(11,200) ( ( OMEGAG( I, J ), I = 1, NG21 ), J = NG21,1,-1 )

      WRITE(11,301)
      WRITE(11,103)( ( XE( I, J ),J = 1, NPAN, 2 ),I = 1, NPAN, 2 )
      WRITE(11,302)
      WRITE(11,103)( ( YE( I, J ),J = 1, NPAN, 2 ),I = 1, NPAN, 2 )

      END IF

*********************  OUTPUT FORMAT  ********************************
*                                                                    *

 50   FORMAT( '1',// 35X, 'PANEL CONFIGURATION & STRENGTH :' /// 3X,
     1        'I', 7X, 'X(I)', 5X, 'Y(I)', 7X, 'XB(I)', 4X,
     1        'YB(I)', 6X, 'PL(I)', 4X, 'THETA(I)', 6X, 'SIGAMA',
     1        6X, 'AGAMA', 6X, 'CP',
     1        // 2X,'---',4X, '-------',2X, '-------', 5X, '-------',
     1        2X, '-------', 5X, '-----', 4X, '---------', 4X,
     1        '---------', 4X, '-------', 6X,'---' / )
 52   FORMAT( I4, 3X, 2F9.4, 3X, 2F9.4, F10.4, 1X, F11.4, F12.4, 2X,
     1        F12.4, F9.4 )
 53   FORMAT( ///, 10X,' NACA - ', 2( I1, 1X ), I2 )
 54   FORMAT( 10X,' SUM OF SOURCE DISTRIBUTION = ', F10.4, /,
     1        10X,' SUM OF VORTEX DISTRIBUTION = ', F10.4 )
 55   FORMAT( 21X, ' ANGLE OF ATTACK = ', F10.4, 'DEG.', /, 21X,
     1        ' ONSET VORTICITY = ', F10.4, /, 20X,
     1        ' RATIO OF CNTL PT = ', F10.4 )
 56   FORMAT( 32X,'   CL = ', F10.4, /, 32X,'   CD = ', F10.4,
     1        /, 22X,' ZERO LIFT ANGL = ', F10.4 )
 57   FORMAT( ///, 10X,' JOUKOWSKI AIRFOIL ', 3E20.6 )
 58   FORMAT( 31X,' YINTY = ', F10.4,/,31X,' UINTY = ', F10.4)
 59   FORMAT( 31X,'   UY0 = ', F10.4,/,31X,'   VX0 = ', F10.4)

 100   FORMAT( '1', /// 35X, 'VELOCITY AND VORTICITY ON AIRFOIL :' ///
     1        3X, 'I', 7X, 'U0(I)', 5X, 'V0(I)', 5X, 'U(I)', 5X,
     1        'V(I)', //, 2X, '---', 4X, '-------', 2X,
     1        '-------', 2X, '-------', 2X, '-------', // )
 101   FORMAT( I4, 3X,4F9.4 )
 102   FORMAT( '1', /// 35X, 'INFLUENCE MATRIX AINTGL1 :', // )
 103   FORMAT( 1X, 25F5.2 )
 104   FORMAT( '1', /// 35X, 'INFLUENCE MATRIX DINTGL1 :', // )
 105   FORMAT( '1', /// 35X, 'INFLUENCE MATRIX AINTGL2 :', // )
 106   FORMAT( '1', /// 35X, 'INFLUENCE MATRIX DINTGL2 :', // )

 200   FORMAT( 1X, 21F6.2 //)
 201   FORMAT( '1', 40X, 'XG( I, J )', ///)
 202   FORMAT( '1', 40X, 'YG( I, J )', ///)
 203   FORMAT( '1', 40X, 'UG( I, J )', ///)
```

```
204     FORMAT( '1', 40X, 'VG( I, J )', ///)
205     FORMAT( '1', 40X, 'OMEGAG0( I, J )', ///)
206     FORMAT( '1', 40X, 'OMEGAG( I, J )', ///)

301     FORMAT( '1', 40X, 'XE( I, J )', ///)
302     FORMAT( '1', 40X, 'YE( I, J )', ///)

        RETURN
        END

        SUBROUTINE PRESUR
************************************************************************
*                                                                      *
*       CALCULATE PRESSURE COEFFICIENT                                 *
*                                                                      *
********************      NOMENCLATURE      ****************************
*                                                                      *
*       CP                PRESSURE COEFFICIENT                         *
*                                                                      *
************************************************************************
*                                                                      *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' AUXI.FIL '
        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' DBLE.FIL '
        INCLUDE ' JOUK.FIL '
        INCLUDE ' STRM.FIL '
        INCLUDE ' VLTY.FIL '

        IF( ( IFLOW .EQ. 1 ) .AND. ( IDBL .EQ. 0 ) )THEN
        USSQ = UY0** 2

        ELSEIF( ( IFLOW .EQ. 1 ) .AND. ( IDBL .LE. 2 ) )THEN
        USSQ = UY0** 2 + VX0** 2

        ELSE
        CALL INTY( PSIST, USSQ )

        END IF

        TYPE*,' USSQ = ', USSQ

        DO K = 1, NPAN

        IF( ( IFLOW .EQ. 1 ) .AND. ( IDBL .LE. 1 ) )THEN
        UY0SQ = UY0** 2
        CP( K ) = ( USSQ - VS( K )** 2 ) / UY0SQ

        ELSEIF( ( IFLOW .EQ. 1 ) .OR. ( IFLOW .EQ. 2 ) )THEN
        UY0SQ = UY0** 2
        CP( K ) = ( USSQ - VS( K )** 2 ) / UY0SQ

        ELSEIF( ( IFLOW .EQ. 1 ) .AND. ( IDBL .EQ. 4 ) )THEN
        UY0SQ = UY0** 2
        CP( K ) = ( USSQ - VS( K )** 2 ) / UY0SQ

        ELSEIF( ( IFLOW .EQ. 1 ) .AND. ( IDBL .EQ. 5 ) )THEN
        UY0SQ = UY0** 2
        CP( K ) = ( USSQ - VS( K )** 2 ) / UY0SQ
```

```
      ELSEIF( IFLOW .EQ. 3 )THEN
      UY0SQ = UG01( 1 )** 2 + VG01( 1 )** 2
      CP( K ) = ( USSQ - VS( K )** 2 ) / UY0SQ

      ELSEIF( IFLOW .EQ. 4 )THEN
      CP( K ) = ( USSQ - VS( K )** 2 ) / 1.0

      ELSEIF( IFLOW .EQ. 5 )THEN
      CP( K ) = ( USSQ - VS( K )** 2 ) / 1.0
      END IF

      END DO


      RETURN
      END

      SUBROUTINE STREAM( ICONT )
***********************************************************************
*                                                                     *
*     CALCULATE STAGNATION STREAM FUNCTION                            *
*                                                                     *
*******************     NOMENCLATURE     ******************************
*                                                                     *
*     XSTAG, YSTAG    COORDINATES FOR STAG. STREAMLINE                *
*                                                                     *
*     XINTG, YINTG    COORDINATES FOR CALCULATING                     *
*                     STAG. STREAM FUNCTION                           *
*                                                                     *
*     PSIST           STAGNATION STREAM FUNCTION                      *
*                                                                     *
***********************************************************************
*                                                                     *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' MTRX.FIL '
      INCLUDE ' STRM.FIL '
      INCLUDE ' VLTY.FIL '
      REAL XX( NG101 ), YY( NG101 ), ASTAG( 300 )

      DO I = 1, NPAN-1
        I1 = I + 1
        IF( ( U( I ) .GE. 0.0 ) .AND. ( U( I1 ) .LE. 0.0 ) )THEN
          ASTAG( ICONT ) = FLOAT( I )
          XSTAG = ( X( I ) + X( I1 ) ) / 2.0
          YSTAG = ( Y( I ) + Y( I1 ) ) / 2.0
          USTAG = ( U( I ) + U( I1 ) ) / 2.0
          VSTAG = ( V( I ) + V( I1 ) ) / 2.0
          GOTO 1
        END IF
          ASTAG( ICONT ) = FLOAT( NPANH )
          XSTAG = 0.0
          YSTAG = 0.0
```

```
          USTAG = 0.0
          VSTAG = 0.0
      END DO

1     IF( ICONT .EQ. 1 )THEN
      X0 = XG1( 1 )
      Y0 = YG1( 1 )

      ELSEIF( ASTAG( ICONT ) .EQ. ASTAG( ICONT-1 ) )THEN
      GOTO 100

      END IF

      DX = ABS( XSTAG - X0 ) / FLOAT( NG100 - 1 )
      DY = ABS( YSTAG - Y0 ) / FLOAT( NG100 - 1 )

      DO K = 1, NG100
        XINTG( K, 1 ) = XSTAG
        YINTG( K, 1 ) = Y0 + FLOAT( K - 1 ) * DY
        XINTG( K, 2 ) = X0 + FLOAT( K - 1 ) * DX
        YINTG( K, 2 ) = Y0
      END DO

      CALL TRANXY
      CALL COEF1
      CALL ONSET1

      DO I = 1, NG100
      SUMU = 0.0
      SUMV = 0.0

        DO J = 1, NPAN
        SUMU = SUMU + A2( I, J, 1 ) * ASIGAMA( J ) + D2( I, J, 1 ) *
     1         AGAMA( J )
        SUMV = SUMV + A1( I, J, 2 ) * ASIGAMA( J ) + D1( I, J, 2 ) *
     1         AGAMA( J )
        END DO

      IF( ( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) .AND.
     1      ( ICONT .GT. 1 ) )THEN
      UINTG( I ) = SUMU + UGINT( I ) + RESLU( XINTG(I,1),YINTG(I,1) )
      VINTG( I ) = SUMV + VGINT( I ) + RESLV( XINTG(I,2),YINTG(I,2) )
      ELSE
      UINTG( I ) = SUMU + UGINT( I )
      VINTG( I ) = SUMV + VGINT( I )
      END IF

      END DO

      UINTG( NG100 ) =  USTAG

      DO I = 1, NG100
      XX( I ) = YINTG( I, 1 )
      YY( I ) = UINTG( I )
      END DO

      A = XX( 1 )
      B = XX( NG100 )
      IY = NG100 - 1
      CALL INTEG1( XX, YY, A, B, IY, NG100, Q )
      PSIY = Q

      DO I = 1, NG100
      XX( I ) = XINTG( I, 2 )
      YY( I ) = VINTG( I )
      END DO
```

```
      A = XX( 1 )
      B = XX( NG100 )
      IX = NG100 - 1
      CALL INTEG1( XX, YY, A, B, IX, NG100, Q )
      PSIX = Q

      PSIST = PSIY - PSIX

100   CALL STFN( PSIST )

      RETURN
      END


      SUBROUTINE INTY( P, USSQ )
***********************************************************************
*                                                                     *
*       CALCULATE THE STREAM FUNCTION OF FAR UPSTREAM AND             *
*       INTERPOLATE THE STAGNATION STREAM FUNCTION                    *
*                                                                     *
*******************     NOMENCLATURE     *****************************
*                                                                     *
*       PSI1            STREAM FUNCTION ARRAY AT FAR UPSTREAM         *
*                                                                     *
*       P               STAGNATION STREAM FUNCTION                    *
*                                                                     *
***********************************************************************
*                                                                     *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' DBLE.FIL '
      INCLUDE ' VLTY.FIL '
      INCLUDE ' STRM.FIL '
      REAL C( NG100, 3 )

      DO J = 1, NG101-1
        J1 = J + 1

      IF( PSI1( J ) .EQ. P )THEN
        UU   = UG01( J )
        VV   = VG01( J )
      USSQ  = UU** 2 + VV** 2
      YINTY = YG1( J )
      UINTY = SQRT( USSQ )
      GOTO 100

      ELSEIF( ( PSI1( J ) .LE. P ) .AND. ( PSI1( J1 ) .GT. P ) )THEN
         N = NG101
        IC = N - 1
         D = P - PSI1( J )
        CALL ICSCCU( PSI1, UG01, N, C, IC, IER )
        UU = ( ( C(J,3) * D + C(J,2) ) * D + C(J,1) ) * D + UG01( J )
        CALL ICSCCU( PSI1, VG01, N, C, IC, IER )
        VV = ( ( C(J,3) * D + C(J,2) ) * D + C(J,1) ) * D + VG01( J )
        USSQ = UU** 2 + VV** 2
        CALL ICSCCU( PSI1, YG1, N, C, IC, IER )
        YINTY = ( ( C(J,3) * D + C(J,2) ) * D + C(J,1) ) * D + YG1(J)
        UINTY = SQRT( USSQ )
        GOTO 100

      END IF
```

```
      END DO

      IF( ABS( P ) .LT. 0.1 )THEN
      P = 0.0
      END IF

100   TYPE*,' USSQ = ', USSQ
      TYPE*,' YINTY = ', YINTY
      TYPE*,' UINTY = ', UINTY

      RETURN
      END


      SUBROUTINE STFN( PSI0 )
************************************************************************
*                                                                      *
*     CALCULATE THE STREAM FUNCTION IN THE DOMAIN                      *
*                                                                      *
********************    NOMENCLATURE     ******************************
*                                                                      *
*     PSIG            STREAM FUNCTION IN THE DOMAIN                    *
*                                                                      *
*     PSI0            STAGNATION STREAM FUNCTION                       *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' STFN.FIL '
      INCLUDE ' VLTY.FIL '
      REAL XX( NGRID ), YY( NGRID )

      DO I = 1, ICRD
        NX = I
        IX = NX - 1

      DO J = 1, NGRID
        NY = J
        IY = NY - 1

      IF( ( I .EQ. 1 ) .AND. ( IY .EQ. 0 ) )THEN
      PSIG( I, J ) = 0.0

      ELSEIF( ( I .EQ. 1 ) .AND. ( IY .GT. 0 ) )THEN
      DO K = 1, NY
      XX( K ) = YG( I, K )
      YY( K ) = UG( I, K )
      END DO
      A = XX( 1 )
      B = XX( NY )
      CALL INTEG2( XX, YY, A, B, IY, NY, Q )
      PSIG( I, J ) = Q

      ELSEIF( ( I .GT. 1 ) .AND. ( IY .EQ. 0 ) )THEN
      DO K = 1, NX
      XX( K ) = XG( K, J )
      YY( K ) = VG( K, J )
      END DO
      A = XX( 1 )
      B = XX( NX )
      CALL INTEG2( XX, YY, A, B, IX, NX, Q )
      PSIG( I, J ) = PSIG( 1, 1 ) - Q
```

```
      ELSEIF( ( I .GT. 1 ) .AND. ( IY .GT. 0 ) )THEN
      DO K = 1, NY
      XX( K ) = YG( I, K )
      YY( K ) = UG( I, K )
      END DO
      A = XX( 1 )
      B = XX( NY )
      CALL INTEG2( XX, YY, A, B, IY, NY, Q )
      PSIG( I, J ) = PSIG( I, 1 ) + Q
      END IF

        DO L = 1, NGRID
        XX( L ) = 0.0
        YY( L ) = 0.0
        END DO

      END DO
      END DO

      DO I = ICRD+1, ICRD1-1
        NX = I
        IX = NX - 1

      DO J = 1, NGRID
        NY = J
        IY = NY - 1

      IF( ( J .LE. JCRDM1 ) .AND. ( IY .EQ. 0 ) )THEN
      DO K = 1, NX
      XX( K ) = XG( K, J )
      YY( K ) = VG( K, J )
      END DO
      A = XX( 1 )
      B = XX( NX )
      CALL INTEG2( XX, YY, A, B, IX, NX, Q )
      PSIG( I, J ) = PSIG( 1, 1 ) - Q

      ELSEIF( ( J .LE. JCRDM1 ) .AND. ( IY .GT. 0 ) )THEN
      DO K = 1, NY
      XX( K ) = YG( I, K )
      YY( K ) = UG( I, K )
      END DO
      A = XX( 1 )
      B = XX( NY )
      CALL INTEG2( XX, YY, A, B, IY, NY, Q )
      PSIG( I, J ) = PSIG( I, 1 ) + Q
      END IF

      IF( J .EQ. JCRDP1 )THEN
      DO N = NPANH1, NPAN
        N1 = N + 1
        IF( ( XG(I,J) .GE. X(N) ) .AND. ( XG(I,J) .LE. X(N1) ) )THEN
        UR = U(N) + ( U(N1)-U(N) ) * ( XG(I,J)-X(N) ) / ( X(N1)-X(N) )
        YR = Y(N) + ( Y(N1)-Y(N) ) * ( XG(I,J)-X(N) ) / ( X(N1)-X(N) )
        GOTO 100
        END IF
      END DO
100   XX( 1 ) = YR
      YY( 1 ) = UR
      XX( 2 ) = YG( I, JCRDP1 )
      YY( 2 ) = UG( I, JCRDP1 )
      A = XX( 1 )
      B = XX( 2 )
      CALL INTEG2( XX, YY, A, B, 1, 2, Q )
      PSIG( I, J ) = PSI0 + Q
```

```
ELSEIF( J .GT. JCRDP1 )THEN
DO K = JCRDP1, NY
KK   = K-JCRDP1+1
XX( KK ) = YG( I, K )
YY( KK ) = UG( I, K )
END DO
A = XX( 1 )
B = XX( KK )
KK1 = KK - 1
CALL INTEG2( XX, YY, A, B, KK1, KK, Q )
PSIG( I, J ) = PSIG( I, JCRDP1 ) + Q
END IF

  DO L = 1, NGRID
  XX( L ) = 0.0
  YY( L ) = 0.0
  END DO

END DO
END DO


DO I = ICRD1, NGRID
  NX = I
  IX = NX - 1

DO J = 1, NGRID
  NY = J
  IY = NY - 1

IF( IY .EQ. 0 )THEN
DO K = 1, NX
XX( K ) = XG( K, J )
YY( K ) = VG( K, J )
END DO
A = XX( 1 )
B = XX( NX )
CALL INTEG2( XX, YY, A, B, IX, NX, Q )
PSIG( I, J ) = PSIG( 1, 1 ) - Q

ELSEIF( IY .GT. 0 )THEN
DO K = 1, NY
XX( K ) = YG( I, K )
YY( K ) = UG( I, K )
END DO
A = XX( 1 )
B = XX( NY )
CALL INTEG2( XX, YY, A, B, IY, NY, Q )
PSIG( I, J ) = PSIG( I, 1 ) + Q
END IF
  DO L = 1, NGRID
  XX( L ) = 0.0
  YY( L ) = 0.0
  END DO
END DO
END DO

DO K = 1, NGRID
IF( ABS( PSIG( K, 1 ) ) .LT. 0.1 )PSIG( K, 1 ) = 0.0
END DO

DO I = ICRD, ICRD1
PSIG( I, JCRD ) = PSI0
END DO
```

```
      RETURN
      END

      SUBROUTINE AEROFR
************************************************************************
*                                                                      *
*     CALCULATE AERODYNAMIC LIFT & DRAG BY CALLING IMSL:               *
*     ICSCCU, DCSQDU ( CUBIC SPLINE )                                  *
*     AND CALCULATE THE COEFFICIENT OF MOMENT REFERENCED TO LE,        *
*     POSITIVE WHEN NOSE-UP                                            *
*                                                                      *
********************      NOMENCLATURE     ****************************
*                                                                      *
*     CL                LIFT COEFFICIENT                               *
*                                                                      *
*     CD                DRAG COEFFICIENT                               *
*                                                                      *
*     CM                MOMENT COEFFICIENT                             *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' VLTY.FIL '
      REAL XX1( NPAN1 ), XX2( NPAN1 ), YY( NPAN1 ), ZZ( NPAN1 )

      SUMX = 0.0
      SUMY = 0.0
        CM = 0.0
      DO K = 1, 8

        GOTO( 1, 2, 3, 4, 5, 6, 7, 8 ), K
 1      I1 = NPANH
        I2 = IMAXL
        I3 = -1
        GOTO 100

 2      I1 = IMAXL
        I2 = 1
        I3 = -1
        GOTO 100

 3      I1 = NPANH1
        I2 = IMAXU
        I3 = 1
        GOTO 100

 4      I1 = IMAXU
        I2 = NPAN
        I3 = 1
        GOTO 100

 5      I1 = IMAXL
        I2 = NPANH
        I3 = 1
        GOTO 200

 6      I1 = IMAXL
        I2 = 1
        I3 = -1
        GOTO 200
```

```
7         I1 = NPANH1
          I2 = IMAXU
          I3 = 1
          GOTO 200

8         I1 = NPAN
          I2 = IMAXU
          I3 = -1
          GOTO 200

100       II = 0
        DO I = I1, I2, I3
          II = II + 1
          XX1( II ) =  X( I )
          YY( II )  = CP( I )
          ZZ( II )  = XX1( II ) * YY( II )
        END DO
                 NN = II
                 LL = NN - 1
                  A = XX1( 1 )
                  B = XX1( NN )
        CALL INTEG( XX1, YY, A, B, LL, NN, CX )
        CALL INTEG( XX1, ZZ, A, B, LL, NN, SUM )

        IF( I3 .GT. 0.0 )THEN
              SUMX = SUMX + CX
                CM = CM + SUM
        ELSE
              SUMX = SUMX - CX
                CM = CM - SUM
        END IF
        GOTO 111


200       II = 0
        DO I = I1, I2, I3
          II = II + 1
          XX2( II ) = Y( I )
          YY( II )  = CP( I )
        END DO
                 NN = II
                 LL = NN - 1
                 A  = XX2( 1 )
                 B  = XX2( NN )

        IF( ( IAIR .EQ. 5 ) .AND. ( IMAXLL .NE. 0 ) .AND.
     1      ( K .EQ. 6 ) )THEN
        IEXE = IMAXL - IMAXLL + 1
        IEND = II
        CALL AEROFR1( XX2, YY, IEXE, IEND, CY )
        ELSE
        CALL INTEG( XX2, YY, A, B, LL, NN, CY )
        END IF

        IF( I3 .GT. 0.0 )THEN
              SUMY = SUMY + CY
        ELSE
              SUMY = SUMY - CY
        END IF

111     DO I = 1, NPAN+1
        XX1( I ) = 0.0
        XX2( I ) = 0.0
        YY( I )  = 0.0
        ZZ( I )  = 0.0
```

```
      END DO
      END DO

      DO I = 1, 8
        GOTO( 12, 13, 14, 15, 16, 17, 18, 19 ), I
12      XX1( 1 ) =  XB( NPANH1 )
        XX1( 2 ) =  X( NPANH )
         YY( 1 ) = 0.5 * ( CP( NPANH ) + CP( NPANH1 ) )
         YY( 2 ) = CP( NPANH )
        GOTO 300

13      XX1( 1 ) =  XB( NPANH1 )
        XX1( 2 ) =  X( NPANH1 )
         YY( 1 ) =  0.5 * ( CP( NPANH ) + CP( NPANH1 ) )
         YY( 2 ) =  CP( NPANH1 )
        GOTO 300

14      XX1( 1 ) =  X( NPAN )
        XX1( 2 ) =  XB( 1 )
         YY( 1 ) = CP( NPAN )
         YY( 2 ) = 0.5 * ( CP( NPAN ) + CP( 1 ) )
        GOTO 300

15      XX1( 1 ) =  X( 1 )
        XX1( 2 ) =  XB( 1 )
         YY( 1 ) =  CP( 1 )
         YY( 2 ) =  0.5 * ( CP( NPAN ) + CP( 1 ) )
        GOTO 300

16      XX2( 1 ) =  Y( NPANH )
        XX2( 2 ) =  YB( NPANH1 )
         YY( 1 ) =  CP( NPANH )
         YY( 2 ) =  0.5 * ( CP( NPANH ) + CP( NPANH1 ) )
        GOTO 400

17      XX2( 1 ) =  YB( NPANH1 )
        XX2( 2 ) =  Y( NPANH1 )
         YY( 1 ) =  0.5 * ( CP( NPANH ) + CP( NPANH1 ) )
         YY( 2 ) =  CP( NPANH1 )
        GOTO 400

18      XX2( 1 ) =  YB( 1 )
        XX2( 2 ) =  Y( NPAN )
         YY( 1 ) =  0.5 * ( CP( NPAN ) + CP( 1 ) )
         YY( 2 ) =  CP( NPAN )
        GOTO 400

19      IF( ( IAIR .EQ. 5 ) .AND. ( IMAXLL .NE. 0 ) )THEN
        XX2( 1 ) = YB( 1 )
        XX2( 2 ) = Y( 1 )
         YY( 1 ) = 0.5 * ( CP( NPAN ) + CP( 1 ) )
         YY( 2 ) = CP( 1 )
           ISEED = 1

        ELSE
        XX2( 1 ) =  Y( 1 )
        XX2( 2 ) =  YB( 1 )
         YY( 1 ) =  CP( 1 )
         YY( 2 ) =  0.5 * ( CP( NPAN ) + CP( 1 ) )
        END IF
        GOTO 400

300           A = XX1( 1 )
```

```
              B = XX1( 2 )
        CALL INTEG( XX1, YY, A, B, 1, 2, CX )
          ZZ( 1 ) = XX1( 1 ) * YY( 1 )
          ZZ( 2 ) = XX1( 2 ) * YY( 2 )
        CALL INTEG( XX1, ZZ, A, B, 1, 2, SUM )

        IF( ( I .EQ. 2 ) .OR. ( I .EQ. 3 ) )THEN
              SUMX = SUMX + CX
                CM = CM + SUM
        ELSEIF( ( I .EQ. 1 ) .OR. ( I .EQ. 4 ) )THEN
              SUMX = SUMX - CX
                CM = CM - SUM
        END IF
        GOTO 20

400           A = XX2( 1 )
              B = XX2( 2 )
        CALL INTEG( XX2, YY, A, B, 1, 2, CY )
        IF( ISEED .EQ. 1 ) CY = - CY
        IF( ( I .EQ. 5 ) .OR. ( I .EQ. 6 ) )THEN
              SUMY = SUMY + CY
        ELSEIF( ( I .EQ. 7 ) .OR. ( I .EQ. 8 ) )THEN
              SUMY = SUMY - CY
        END IF

20      CONTINUE
        END DO

        CL = - COS( ALPHA ) * SUMX - SIN( ALPHA ) * SUMY
        CD = - SIN( ALPHA ) * SUMX + COS( ALPHA ) * SUMY
        TYPE*,' CL = ', CL
        TYPE*,' CD = ', CD
        TYPE*,' CM = ', CM

        RETURN
        END


        SUBROUTINE AEROFR1( XX, YY, IEXE, IEND, Q )
*********************************************************************
*                                                                   *
*       SUB-INTEGRATION OF JOUKOWSKI AIRFOIL W. 2ND LOCAL LIMIT     *
*                                                                   *
*********************************************************************
*                                                                   *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        REAL XX( NPAN1 ), YY( NPAN1 )
        REAL X1( NPAN1 ), Y1( NPAN1 ), X2( NPAN1 ), Y2( NPAN1 )

        DO I = 1, IEXE
        X1( I ) = XX( I )
        Y1( I ) = YY( I )
        END DO

        NN = IEXE
        LL = NN - 1
        A  = X1( 1 )
        B  = X1( NN )
        CALL INTEG( X1, Y1, A, B, LL, NN, Q1 )

          II = 0
        DO I = IEND, IEXE, -1
```

```
          II = II + 1
          X2( II ) = X( I )
          Y2( II ) = Y( I )
        END DO

        NN = II
        LL = NN - 1
        A  = X2( 1 )
        B  = X2( NN )
        CALL INTEG( X2, Y2, A, B, LL, NN, Q2 )

        Q = Q1 - Q2

        RETURN
        END

        SUBROUTINE INTEG( X, Y, A, B, IC, NX, Q )
***********************************************************************
*                                                                     *
*          INTEGRATION BY CALLING  IMSL:DCSQDU, ICSCCU                *
*          CUBIC SPLINE QUADRATURE ( INTEGRATION )                    *
*          FOR ARRAY ( NPAN )                                         *
*                                                                     *
********************     NOMENCLATURE     *****************************
*                                                                     *
*       X                 INDEPENDENT VARIABLE                        *
*                                                                     *
*       Y                 DEPEDENT VARIABLE                           *
*                                                                     *
*       A, B              END CONDITIONS                              *
*                                                                     *
*       IC                ÀRRAY OF INTEGRATION                        *
*                                                                     *
*       NX                ORDER OF X ARRAY                            *
*                                                                     *
*       Q                 INTEGRATION VALUE                           *
*                                                                     *
***********************************************************************
*                                                                     *
        PARAMETER ( NPAN = 100, NPAN1 = 101 )
        REAL X( NPAN1 ), Y( NPAN1 ), C( NPAN, 3 )
        INTEGER NX, IC, IER1, IER2

        CALL ICSCCU( X, Y, NX, C, IC, IER1 )

        CALL DCSQDU( X, Y, NX, C, IC, A, B, Q, IER2 )


        IF( ( IER1 .NE. 0.0 ) .OR. ( IER2 .NE. 0.0 ) )THEN
        TYPE*, ' IER1 = ', IER1
        TYPE*, ' IER2 = ', IER2
        TYPE*,'X=',X
        TYPE*,'Y=',Y
        CALL OUTPUT
        STOP 555
        END IF

        RETURN
        END

        SUBROUTINE INTEG1( X, Y, A, B, IC, NX, Q )
***********************************************************************
*                                                                     *
*          INTEGRATION BY CALLING  IMSL:DCSQDU, ICSCCU                *
*          CUBIC SPLINE QUADRATURE ( INTEGRATION )                    *
```

```
*          FOR ARRAY ( NG101 )                                            *
*                                                                         *
*******************     NOMENCLATURE     *****************************
*                                                                         *
*       X              INDEPENDENT VARIABLE                               *
*                                                                         *
*       Y              DEPEDENT VARIABLE                                  *
*                                                                         *
*       A, B           END CONDITIONS                                     *
*                                                                         *
*       IC             ARRAY OF INTEGRATION                               *
*                                                                         *
*       NX             ORDER OF X ARRAY                                   *
*                                                                         *
*       Q              INTEGRATION VALUE                                  *
*                                                                         *
***************************************************************************
*                                                                         *
*                                                                         *
      PARAMETER ( NG101 = 101, NG100 = 100 )
      REAL X( NG101 ), Y( NG101 ), C( NG100, 3 )
      INTEGER NX, IC, IER1, IER2

      CALL ICSCCU( X, Y, NX, C, IC, IER1 )

      CALL DCSQDU( X, Y, NX, C, IC, A, B, Q, IER2 )


      IF( ( IER1 .NE. 0.0 ) .OR. ( IER2 .NE. 0.0 ) )THEN
      TYPE*, ' IER1 = ', IER1
      TYPE*, ' IER2 = ', IER2
      TYPE*,'X=',X
      TYPE*,'Y=',Y
      CALL OUTPUT
      STOP 555
      END IF

      RETURN
      END


      SUBROUTINE INTEG2( X, Y, A, B, IC, NX, Q )
***************************************************************************
*                                                                         *
*          INTEGRATION BY CALLING  IMSL:DCSQDU, ICSCCU                    *
*          CUBIC SPLINE QUADRATURE ( INTEGRATION )                        *
*          FOR ARRAY ( NGRID )                                            *
*                                                                         *
*******************     NOMENCLATURE     *****************************
*                                                                         *
*       X              INDEPENDENT VARIABLE                               *
*                                                                         *
*       Y              DEPEDENT VARIABLE                                  *
*                                                                         *
*       A, B           END CONDITIONS                                     *
*                                                                         *
*       IC             ARRAY OF INTEGRATION                               *
*                                                                         *
*       NX             ORDER OF X ARRAY                                   *
*                                                                         *
*       Q              INTEGRATION VALUE                                  *
*                                                                         *
***************************************************************************
*                                                                         *
*                                                                         *
      PARAMETER ( NGRID = 41 )
```

```
        REAL X( NGRID ), Y( NGRID ), C( NGRID-1, 3 )
        INTEGER NX, IC, IER1, IER2

        CALL ICSCCU( X, Y, NX, C, IC, IER1 )

        CALL DCSQDU( X, Y, NX, C, IC, A, B, Q, IER2 )


        IF( ( IER1 .NE. 0.0 ) .OR. ( IER2 .NE. 0.0 ) )THEN
        TYPE*, ' IER1 = ', IER1
        TYPE*, ' IER2 = ', IER2
        TYPE*,'X=',X
        TYPE*,'Y=',Y
        CALL OUTPUT
        STOP 555
        END IF

        RETURN
        END

        SUBROUTINE SHAPE
************************************************************************
*                                                                      *
*       DETERMINE THE GEOMETRY OF THE AIRFOIL:                         *
*                                                                      *
********************       NOMENCLATURE      ****************************
*                                                                      *
*       IAIR  = 1,     NACA 0012                                       *
*               2,     NACA 2412                                       *
*               3,     NACA 4 DIGIT( INPUT AM, AP, T )                 *
*               4,     SYMMETRIC JOUKOWSKI                             *
*               5,     CAMBERED JOUKOWSKI                              *
*                                                                      *
*       AM, AP, AT     PARAMETERS FOR NACA-4 DIGIT AIRFOILS            *
*                                                                      *
*       XZ, YZ, AZ     PARAMETERS FOR JOUKOWSKI AIRFOILS               *
*                                                                      *
************************************************************************
*                                                                      *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' BDPT.FIL '
        INCLUDE ' CNST.FIL '
        INCLUDE ' JOUK.FIL '
        INCLUDE ' NAC4.FIL '

        GOTO( 100, 200, 300, 400, 500 ), IAIR

100     TYPE*,'NACA 0012'
        AM = 0.0
        AP = 0.0
        AT = 0.12
        CALL NACA4
        GOTO 600

200     TYPE*,'NACA 2412'
        AM = 0.02
        AP = 0.4
        AT = 0.12
        CALL NACA4
        GOTO 600

300     TYPE*,'NACA 4 DIGIT AIRFOIL, AM, AP, AT =', AM, AP, AT
        CALL NACA4
        GOTO 600
```

```
400     TYPE*,'SYMMETRIC JOUKOWSKI AIRFOIL'
        XZ = -9.2376E-2
        YZ =  0.0
        AZ =  1.092376
        CALL JOUKAR
        GOTO 600

500     TYPE*,'CAMBERED JOUKOWSKI AIRFOIL'
        XZ = -0.05
        YZ =  0.05
        AZ =  0.50
        CALL JOUKAR

600     DO I = 1, NPAN2
        WRITE( 111, * ) XB1( I ), YB1( I )
        END DO

        CLOSE( UNIT = 111 )

        RETURN
        END


        SUBROUTINE NACA4
************************************************************************
*                                                                      *
*                 CALCULATE NACA 4 DIGIT AIRFOIL                       *
*                                                                      *
********************       NOMENCLATURE      ******************************
*                                                                      *
*       XX; YY        THE COORDINATES FOR THE AIRFOIL GEOMETRY         *
*                                                                      *
*       C1 - C5       PARAMETERS FOR GENERATING THE AIRFOIL (ABBOT)    *
*                                                                      *
*       YT            THICKNESS DISTRIBUTION                           *
*                                                                      *
*       YC            CAMBER DISTRIBUTION                              *
*                                                                      *
************************************************************************
*                                                                      *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' BDPT.FIL '
        INCLUDE ' CNST.FIL '
        INCLUDE ' NAC4.FIL '
        REAL X( NPAN2 ), YT( NPAN2 ), YC( NPAN2 ), THETA( NPAN2 ),
     1       XU( NPANH1 ), YU( NPANH1 ), XL( NPANH1 ), YL( NPANH1 )

        DATA  C1, C2, C3, C4, C5 / 0.29640, -0.12640,
     1                             -0.35200, 0.28390, -0.10190 /

          DA = PI / FLOAT( NPANH )
        DO I = 1, NPAN2

          IF( I .LE. NPANH1 ) THEN
          ANGL = PI - ( FLOAT( I ) - 1.0 ) * DA
          ELSE
          ANGL = - PI + ( FLOAT( I ) - FLOAT( NPANH2 ) ) * DA
          END IF

          X( I ) = 0.5 * ( 1.0 + COS( ANGL ) )

          IF( X(I) .EQ. 0.0 ) THEN
          YT( I ) = 0.0
```

```
          ELSE
          YT( I ) = AT * ( C1 * SQRT( X(I) ) + C2 *
     1              X(I) + C3 * X(I)** 2 + C4 * X(I)** 3 +
     1              C5 * X(I)** 4 ) / 0.20
          END IF

          IF( AM .EQ. 0.0 )THEN
          YC( I ) = 0.0
          THETA( I ) = 0.0

          ELSEIF( ( X(I) .LE. AP ) .AND. ( AM .NE. 0.0 ) )THEN
          YC( I ) = AM * ( 2.0 * AP * X(I) - X(I)** 2 ) / AP / AP
          THETA( I ) = ATAN( 2.0 * AM * ( AP - X(I) ) / AP / AP )

          ELSEIF( ( X(I) .GT. AP ) .AND. ( AM .NE. 0.0 ) )THEN
          YC( I ) = AM * ( 1.0 - 2.0 * AP + 2.0 * AP * X(I) -
     1              X(I)** 2 ) / ( 1.0 - AP )** 2
          THETA( I ) = ATAN( 2.0 * AM * ( AP - X(I) ) /
     1                 ( 1.0 - AP )** 2 )
          END IF

          IF( I .LE. NPANH1 )THEN
          XU( I ) = X( I ) - YT( I ) * SIN( THETA(I) )
          YU( I ) = YC( I ) + YT( I ) * COS( THETA(I) )

          ELSE
          XL( I - NPANH1 ) = X( I ) + YT( I ) * SIN( THETA(I) )
          YL( I - NPANH1 ) = YC( I ) - YT( I ) * COS( THETA(I) )
          END IF

        END DO

        XU( NPANH1 ) = 1.0
        YU( NPANH1 ) = 0.0
        XL( NPANH1 ) = 1.0
        YL( NPANH1 ) = 0.0

        DO I = 1, NPAN2

        IF( I .LE. NPANH1 ) THEN
        XB1( I ) = XU( I )
        YB1( I ) = YU( I )

        ELSE
        XB1( I ) = XL( I - NPANH1 )
        YB1( I ) = YL( I - NPANH1 )
        END IF

        END DO

        CALL ZLANGL

        RETURN
        END


        SUBROUTINE ZLANGL
***********************************************************************
*                                                                     *
*     CALCULATE ZERO LIFT ANGL FOR NACA 4 DIGIT AIRFOIL               *
*                                                                     *
*******************     NOMENCLATURE      *****************************
*                                                                     *
*      DALPHA           ZERO LIFT ANGLE                               *
*                                                                     *
***********************************************************************
```

```
*                                                                      *
      INCLUDE ' CNST.FIL '
      INCLUDE ' NAC4.FIL '

      IF( AM .NE. 0.0 )THEN
          A0 = ACOS( 1.0 - 2.0 * AP )
          A1 = ( ( -2.0*AM/AP + 3.0*AM/2.0/AP/AP + 2.0*AM*AP/
     1          (1.0-AP)** 2 - 3.0*AM/2.0/(1.0-AP)** 2 ) * A0 +
     2          ( 2.0*AM/AP - 2.0*AM/AP/AP - 2.0*AM*AP/
     3          (1.0-AP)**2 + 2.0 * AM / ( 1.0 - AP )** 2 ) *
     4          SIN( A0 ) + ( AM/4.0/AP/AP - AM/4.0/
     5          (1.0-AP)** 2 )*SIN( 2.0*A0 ) + ( -2.0*AM*AP/
     6          ( 1.0-AP )** 2 + 3.0 * AM / 2.0 /
     7          ( 1.0 - AP )** 2 ) * PI ) / PI
      END IF
      DALPHA = ABS( A1 )
          A1 = -ABS( A1 * 180.0 / PI )
          TYPE*, ' ZERO LIFT ANGL (DEG) ', A1

      RETURN
      END


      SUBROUTINE JOUKAR
************************************************************************
*                                                                      *
*                   CALCULATE JOUKOWSKI AIRFOILS                       *
*                                                                      *
********************          NOMENCLATURE       ***********************
*                                                                      *
*       XS, YS       THE COORDINATES FOR THE AIRFOIL GEOMETRY          *
*                                                                      *
*       X            COORDINATES ALONG THE AIRFOIL CHORD               *
*                                                                      *
*       AL           AIRFOIL CHORD LENGTH                              *
*                                                                      *
*       DALPHA       ZERO LIFT LINE                                    *
*                                                                      *
*       BB           CAMBER                                            *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' BDPT.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' JOUK.FIL '

         ASQ = AZ** 2
           C = SQRT( AZ** 2 - YZ** 2 ) - ABS( XZ )
         CSQ = C** 2
          BB = ATAN( YZ / ( XZ - C ) )
          AM = SQRT( XZ** 2 + YZ** 2 )
        AMSQ = XZ** 2 + YZ** 2
          DD = PI + ATAN( YZ / XZ )

      DALPHA = ABS( BB )
         BBD = -ABS( BB * 180.0 / PI )

      TYPE*,'XZ,YZ,AZ =',XZ,YZ,AZ
      TYPE*,' ZERO LIFT ANGLE ( DEG ) = ', BBD

        DA = PI / FLOAT( NPANH )
      XMAX = 0.0
      XMIN = 0.0
```

```
      DO I = 1, NPAN2

        IF( I .LE. NPANH1 ) THEN
        ANGL = PI - ( FLOAT( I ) - 1.0 ) * DA
        ELSE
        ANGL = - PI + ( FLOAT( I ) - FLOAT( NPANH2 ) ) * DA
        END IF


           SINA = SIN( ANGL )
           COSA = COS( ANGL )
        XB1( I ) = XZ + AZ * COSA + CSQ * ( XZ + AZ * COSA ) /
     1              ( AMSQ + 2.0 * AZ * ( XZ * COSA + YZ * SINA ) +
     1                ASQ )

        YB1( I ) = YZ + AZ * SINA - CSQ * ( YZ + AZ * SINA ) /
     1              ( AMSQ + 2.0 * AZ * ( XZ * COSA + YZ * SINA ) +
     1                ASQ )

        IF( XB1( I ) .LT. XMIN ) THEN
        XMIN = XB1( I )
        IMIN = I
        AMIN = ANGL
        END IF

        IF( XB1( I ) .GT. XMAX ) THEN
        XMAX = XB1( I )
        IMAX = I
        AMAX = ANGL
        END IF

      END DO

      IF( ( ( IMIN .EQ. 1 ) .OR. ( IMIN .EQ. NPANH2 ) ) .AND.
     1    ( ( IMAX .EQ. NPANH1 ) .OR. ( IMAX .EQ. NPAN2 ) ) )THEN
      GOTO 100
      END IF

      IF( IMIN .LT. NPANH1 ) THEN
      DALU =  AMIN
      DALL = - PI - ( PI - AMIN )
      ELSE
      DALU = PI + ( AMIN - ( - PI ) )
      DALL = AMIN
      END IF

      IF( IMAX .LT. NPANH1 ) THEN
      DATU = AMAX
      DATL = AMAX
      ELSE
      DATU = AMAX
      DATL = AMAX
      END IF

      DAU = ( DALU - DATU ) / FLOAT( NPANH )
      DAL = ( DALL - DATL ) / FLOAT( NPANH )

     XMAX = 0.0
     XMIN = 0.0

     DO I = 1, NPAN2

       IF( I .LE. NPANH1 ) THEN
       ANGL = DALU - ( FLOAT( I ) - 1.0 ) * DAU
       ELSE
       ANGL = DALL - ( FLOAT( I ) - NPANH2 ) * DAL
```

```
          END IF

             SINA = SIN( ANGL )
             COSA = COS( ANGL )
          XB1( I ) = XZ + AZ * COSA + CSQ * ( XZ + AZ * COSA ) / (AMSQ +
     1              2.0 * AZ * ( XZ * COSA + YZ * SINA ) + ASQ )

          YB1( I ) = YZ + AZ * SINA - CSQ * ( YZ + AZ * SINA ) / (AMSQ +
     1              2.0 * AZ * ( XZ * COSA + YZ * SINA ) + ASQ )

          IF( XB1( I ) .LT. XMIN ) THEN
          XMIN = XB1( I )
          IMIN = I
          AMIN = ANGL
          END IF

          IF( XB1( I ) .GT. XMAX ) THEN
          XMAX = XB1( I )
          IMAX = I
          AMAX = ANGL
          END IF

       END DO

       IF( ( ( IMIN .EQ. 1 ) .OR. ( IMIN .EQ. NPANH2 ) ) .AND.
     1      ( ( IMAX .EQ. NPANH1 ) .OR. ( IMAX .EQ. NPAN2 ) ) )THEN
       GOTO 100

       ELSE
       TYPE*,'IMIN, XMIN, AMIN', IMIN, XMIN, AMIN
       TYPE*,'IMAX, XMAX, AMAX', IMAX, XMAX, AMAX
       STOP 777
       END IF

 100   ALZ = XMAX - XMIN
       TYPE*,'CHORD LENGTH OF JOUKOWSKI AIRFOIL', ALZ

       DO I = 1, NPAN+2

       XB1( I ) = ( XB1( I ) - XMIN ) / ALZ
       YB1( I ) = YB1( I ) / ALZ

       IF( ABS( XB1(I) ) .LT. 1.0E-6 )THEN
       XB1( I ) = 0.0
       END IF

       IF( ABS( YB1(I) ) .LT. 1.0E-6 )THEN
       YB1( I ) = 0.0
       END IF

       END DO

       RETURN
       END

       SUBROUTINE BDYCORD
************************************************************************
*                                                                      *
*      COMPUTE COORDINATES OF BOUNDARY PTS.                            *
*                                                                      *
********************      NOMENCLATURE      ******************************
*                                                                      *
*      XB, YB          THE COORDINATES FOR THE PANELS                  *
*                                                                      *
************************************************************************
*                                                                      *
```

```
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' BDPT.FIL '

      DO I = 1, NPAN2
       IM1 = I - 1

      IF( I .LE. NPANH1 ) THEN
      XB( I ) = XB1( NPAN2 - IM1 )
      YB( I ) = YB1( NPAN2 - IM1 )

      ELSE
      XB( IM1 ) = XB1( I - NPANH1 )
      YB( IM1 ) = YB1( I - NPANH1 )
      END IF

      END DO

      RETURN
      END

      SUBROUTINE PANEL
***********************************************************************
*                                                                     *
*     SPECIFY THE PANEL ORIENTATIONS ( NPAN PANELS ) :                *
*                                                                     *
*******************     NOMENCLATURE     ******************************
*                                                                     *
*     THETA          PANEL ORIENTATION ANGLES                         *
*                                                                     *
*     BETA           PANEL NORMAL ANGLE                               *
*                                                                     *
*     PL             PANEL LENGTH                                     *
*                                                                     *
***********************************************************************
*                                                                     *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '

      DO  I = 1, NPAN
         I1 = I + 1
       YXIJ = ( YB( I1 ) - YB( I ) ) / ( XB( I1 ) - XB( I ) )
      PL(I) = SQRT( ( XB(I1)-XB(I) )** 2 + ( YB(I1)-YB(I) )** 2 )

      IF( ( I1 .GE. NPANH1 ) .AND. ( I .GE. NPANH1 ) )THEN

*****  UPPER SURFACE   **********
      THETA( I ) = ATAN( YXIJ )

      ELSE
*****  LOWER SURFACE   **********
      THETA( I ) = - ( PI - ATAN( YXIJ ) )
      END IF

         BETA( I ) = THETA( I ) + PI / 2.0
       COSINE( I ) = COS( THETA( I ) )
         SINE( I ) = SIN( THETA( I ) )

      END DO
```

```
******* CHECK WITH OTHER METHOD ********
      DO   I = 1, NPAN
           I1 = I + 1
          EXX = ( XB( I1 ) - XB( I ) ) / PL( I )
          EXY = ( YB( I1 ) - YB( I ) ) / PL( I )
      IF( ( ABS( EXX-COSINE(I) ) .GT. 1.E-6 ) .OR.
     1     ( ABS( EXY-SINE(I) ) .GT. 1.E-6 ) ) THEN
      TYPE*, ' TANGENTIAL IS INCORRECT '
      TYPE*, THETA

      ELSEIF( ( ABS( -EXY+SINE(I) ) .GT. 1.E-6 ) .OR.
     1     ( ABS( EXX-COSINE(I) ) .GT. 1.E-6 ) ) THEN
      TYPE*, ' NORMAL IS INCORRECT '
      TYPE*, THETA
      END IF

      END DO

      RETURN
      END

      SUBROUTINE CNTCORD
************************************************************************
*                                                                      *
*     SPECIFY THE COORDINATES OF CONTROL PTS.                          *
*                                                                      *
********************    NOMENCLATURE    ******************************
*                                                                      *
*     X, Y            COORDINATES FOR CONTROL POINTS                   *
*                                                                      *
*     RCP             PARAMETER DETERMINING THE LOCATION OF C.P.       *
*                                                                      *
*     IMAXL           PARAMETER DETERMINING THE LOCAL EXTREME ON       *
*                     THE AIRFOIL LOWER SURFACE                        *
*                                                                      *
*     IMAXLL          PARAMETER DETERMINING THE 2ND LOCAL EXTREME ON   *
*                     THE AIRFOIL LOWER SURFACE                        *
*                                                                      *
*     IMAXU           PARAMETER DETERMINING THE LOCAL EXTREME ON       *
*                     THE AIRFOIL UPPER SURFACE                        *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '

      DO I = 1, NPAN
        I1 = I + 1

      IF( ( I1 .GE. NPANH1 ) .AND. ( I .GE. NPANH1 ) )THEN
******* UPPER SURFACE **************************************************
      X( I ) =  XB( I ) + RCP * PL( I ) * COSINE( I )
      Y( I ) =  YB( I ) + RCP * PL( I ) * SINE( I )

      ELSE
******* LOWER SURFACE **************************************************
      X( I ) =  XB( I ) + ( 1.0 - RCP ) * PL( I ) * COSINE( I )
      Y( I ) =  YB( I ) + ( 1.0 - RCP ) * PL( I ) * SINE( I )
      END IF
```

```
      END DO

      AMAX = 0.0
      DO I = 1, NPANH
        IF( ABS( Y(I) ) .GT. AMAX )THEN
        AMAX = ABS( Y(I) )
        IMAXL = I
        END IF
      END DO

      AMAX = 0.0
      DO I = NPANH1, NPAN
        IF( Y( I ) .GT. AMAX )THEN
        AMAX = Y( I )
        IMAXU = I
        END IF
      END DO

      DO I = 2, IMAXL-1
       IP1 = I + 1
       IM1 = I - 1

      IF( ( Y(I) .GT. Y(IM1) ) .AND. ( Y(I) .GT. Y(IP1) ) )THEN
      IMAXLL = I
      END IF

      END DO

      RETURN
      END

      SUBROUTINE GRDCORD
************************************************************************
*                                                                      *
*     COMPUTE THE COORDINATES OF INNER GRID PTS. OF A SQUARE DOMAIN    *
*                                                                      *
********************     NOMENCLATURE     *****************************
*                                                                      *
*     XG, YG          COORDINATES FOR INNER GRID POINTS                *
*                                                                      *
*     XG1, YG1        COORDINATES FOR FAR UPSTREAM POINTS              *
*                                                                      *
*     DS              INNER MESH SIZE                                  *
*                                                                      *
*     NG21            MAX. NO. OF MESH PT. ( NG21 BY NG21 )            *
*                                                                      *
*     NG101           # OF GRID AT FAR UPSTREAM                        *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' DBLE.FIL '

      SEDGE =  S / 2.0
         DS = S / FLOAT( NGRID  - 1 )
      DO  J = 1, NGRID
      DO  I = 1, NGRID
        XG( I, J ) = - SEDGE + ( FLOAT( I ) - 1.0 ) * DS
        YG( I, J ) = - SEDGE + ( FLOAT( J ) - 1.0 ) * DS
      END DO
      END DO
```

```
      DYI   = S / FLOAT( NG101 - 1 )
      IF( IFLOW .EQ. 3 )THEN
      DO  K = 1, NG101
      XG1( K ) = - SEDGE
      YG1( K ) = - SEDGE + ( FLOAT( K ) - 1.0 ) * DYI
      END DO

      ELSEIF( ( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) .AND.
     1          ( IDIR  .EQ. 1 ) )THEN
      DO  K = 1, NG101
      XG1( K ) = - SEDGE
      YG1( K ) = - SEDGE + ( FLOAT( K ) - 1.0 ) * DYI
      XG2( K ) = - SEDGE + ( FLOAT( K ) - 1.0 ) * DYI
      YG2( K ) = SEDGE
      END DO

      ELSEIF( ( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) .AND.
     1       ( IDIR .EQ. 0 ) )THEN
      DO  K = 1, NG101
      XG1( K ) = - SEDGE
      YG1( K ) = - SEDGE + ( FLOAT( K ) - 1.0 ) * DYI
      XG2( K ) = - SEDGE + ( FLOAT( K ) - 1.0 ) * DYI
      YG2( K ) = - SEDGE
      END DO

      END IF

      JCRD = FLOAT( NGRID - 1 ) / 2.0 + 1
      ICRD = JCRD
      INCR = 1.0 / DS
      ICRD1= ICRD + INCR

      ICRDM1 = ICRD - 1
      ICRD11 = ICRD1 + 1
      NGRID1 = NGRID - 1
      JCRDM1 = JCRD - 1
      JCRDM2 = JCRD - 2
      JCRDP1 = JCRD + 1
      JCRDP2 = JCRD + 2

      RETURN
      END

      SUBROUTINE TRACORD
************************************************************************
*                                                                      *
*     TRANSFORM THE COORDINATES OF CONTROL PTS.  & GRID PTS.          *
*     RELATIVE TO LOCAL COORD. ON EACH PANEL                           *
*                                                                      *
*******************       NOMENCLATURE      ****************************
*                                                                      *
*     XE, YE          COORDINATES OF CONTROL POINT W.R.T. EACH PANEL *
*                                                                      *
*     XGE, YGE        COORDINATES OF INNER GRID POINT W.R.T.           *
*                     EACH PANEL                                       *
*                                                                      *
*     XOE, YOE        COORDINATES OF OUTTER GRID POINT W.R.T.          *
*                     EACH PANEL                                       *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
```

```
      INCLUDE ' CORD.FIL '

      DO I = 1, NPAN
      DO J = 1, NPAN

        XE( I, J ) = COSINE( J ) * ( X( I ) - XB( J ) ) +
     1                 SINE( J ) * ( Y( I ) - YB( J ) )
        YE( I, J ) = - SINE( J ) * ( X( I ) - XB( J ) ) +
     1               COSINE( J ) * ( Y( I ) - YB( J ) )

        IF( ABS( XE(I,J) ) .LE. 1.0E-6 ) THEN
        XE( I, J ) = 0.0
        END IF

        IF( ABS( YE(I,J) ) .LE. 1.0E-6 ) THEN
        YE( I, J ) = 0.0
        END IF

      END DO
      END DO

      IF( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) THEN

      DO I = 1, NGRID
      DO J = 1, NGRID
      DO K = 1, NPAN

        XGE( I, J, K ) = COSINE( K ) * ( XG( I, J ) - XB( K ) ) +
     1                     SINE( K ) * ( YG( I, J ) - YB( K ) )
        YGE( I, J, K ) = - SINE( K ) * ( XG( I, J ) - XB( K ) ) +
     1                   COSINE( K ) * ( YG( I, J ) - YB( K ) )

        IF( ABS( XGE( I, J, K ) ) .LE. 1.0E-6 ) THEN
        XGE( I, J, K ) = 0.0
        END IF

        IF( ABS( YGE( I, J, K ) ) .LE. 1.0E-6 ) THEN
        YGE( I, J, K ) = 0.0
        END IF

      END DO
      END DO
      END DO

      END IF

      RETURN
      END

      SUBROUTINE TRANXY
**********************************************************************
*                                                                    *
*     TRANSFORM THE COORDINATES OF STREAM PTS. W.R.T.                *
*     THE LOCAL COORDINATE ON EACH PANEL                             *
*                                                                    *
********************     NOMENCLATURE     ****************************
*                                                                    *
*     XSE, YSE        TRANSFORMED COORD. FOR STREAM PT.              *
*                                                                    *
**********************************************************************
*                                                                    *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
```

```
      INCLUDE ' CORD.FIL '
      INCLUDE ' STRM.FIL '

      DO K = 1, 2
      DO I = 1, NG100
      DO J = 1, NPAN
        XSE( I, J, K ) = COSINE( J ) * ( XINTG( I, K ) - XB( J ) ) +
     1                     SINE( J ) * ( YINTG( I, K ) - YB( J ) )
        YSE( I, J, K ) = - SINE( J ) * ( XINTG( I, K ) - XB( J ) ) +
     1                   COSINE( J ) * ( YINTG( I, K ) - YB( J ) )
      END DO
      END DO
      END DO

      RETURN
      END

      SUBROUTINE SORT( U, V, V0 )
************************************************************************
*                                                                      *
*     SORT THE VELOCITIES( UCN, VCN ) OF FOUR CORNERS AND              *
*     NORMALIZE THEM REFERENCED TO VREF                                *
*                                                                      *
********************     NOMENCLATURE     ******************************
*                                                                      *
*     U, V            JAWS VELOCITIES AT FOUR CORNERS                  *
*                                                                      *
************************************************************************
*                                                                      *

      INCLUDE ' CNST.FIL '
      REAL U( 4 ), V( 4 ), VN( 4 ), A( 4 )

      DO K = 1, 4

        GOTO( 1, 2, 3, 4 ), K
1       KCONT = 2
        GOTO 5

2       KCONT = 1
        GOTO 5

3       KCONT = 4
        GOTO 5

4       KCONT = 3

5             UU = -U( KCONT ) - ( -V0 )
              VV =  V( KCONT )
        VN( K ) = SQRT( UU** 2 + VV** 2 ) / V0

        IF( ( ( UU .GE. 0.0 ) .AND. ( VV .GE. 0.0 ) ) .OR.
     1      ( ( UU .GE. 0.0 ) .AND. ( VV .LE. 0.0 ) ) )THEN
        A( K ) = ATAN( VV / UU )

        ELSEIF( ( UU .LE. 0.0 ) .AND. ( VV .GE. 0.0 ) )THEN
        A( K ) = PI + ATAN( VV / UU )

        ELSEIF( ( UU .LE. 0.0 ) .AND. ( VV .LE. 0.0 ) )THEN
        A( K ) = -PI + ATAN( VV / UU )
        END IF

      END DO

      DO K = 1, 4
        U( K ) = VN( K ) * COS( A( K ) )
```

```
          V( K ) = VN( K ) * SIN( A( K ) )
        END DO

        RETURN
        END


        SUBROUTINE JAWSFL
************************************************************************
*       READ ONSET FLOW VELOCITY AT 4 CORNERS FROM JAWS DATA AND THEN  *
*       INTERPOLATE THE ONSET FLOW VELOCITY AT ANY POINT IN THE DOMAIN *
*       USING SUB-AREA AS WEIGHTING PARAMETER :                        *
*                                                                      *
*                     4 _____________________ 3                        *
*                       l    A2     l  A1   l                          *
*                       l___________x_______l                          *
*                       l          pl       l                          *
*                       l    A3     l  A4   l                          *
*                       l___________l_______l                          *
*                     1                       2                        *
*                                                                      *
*                       Wp = ( Wi * Ai ) / ( Aj )  ( i, j = 1, 4 )     *
*                                                                      *
********************     NOMENCLATURE     *****************************
*                                                                      *
*       U0, V0          ONSET VELOCITY ON THE CONTROL POINT            *
*                                                                      *
*       UG0, VG0        ONSET VELOCITY IN THE DOMAIN                   *
*                                                                      *
*       UG01, VG01      ONSET VELOCITY IN THE FAR UPSTREAM             *
*                                                                      *
************************************************************************
*                                                                      *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' JAWS.FIL '
        INCLUDE ' VLTY.FIL '
        REAL A( 4 )

        XEDGE = XTOT / 2.0
        YEDGE = YTOT / 2.0
        DO  K = 1, NG101

           PX = XG1( K )
           PY = YG1( K )
           CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
           UG01( K ) = WINTPL( UCN, A )
           VG01( K ) = WINTPL( VCN, A )

           IF( ALPHA .NE. 0.0 )THEN
           CALL JAWSAL( UG01( K ), VG01( K ), ALPHA )
           END IF

           PX = XG2( K )
           PY = YG2( K )
           CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
           UG02( K ) = WINTPL( UCN, A )
           VG02( K ) = WINTPL( VCN, A )

           IF( ALPHA .NE. 0.0 )THEN
           CALL JAWSAL( UG02( K ), VG02( K ), ALPHA )
           END IF
```

```
      END DO


      DO  I = 1, NGRID
      DO  J = 1, NGRID

         PX = XG( I, J )
         PY = YG( I, J )
         CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
         UG0( I, J ) = WINTPL( UCN, A )
         VG0( I, J ) = WINTPL( VCN, A )

         IF( ALPHA .NE. 0.0 )THEN
         CALL JAWSAL( UG0( I, J ), VG0( I, J ), ALPHA )
         END IF

      END DO
      END DO

      DO  K = 1, NPAN

         PX = X( K )
         PY = Y( K )
         CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
         U0( K ) = WINTPL( UCN, A )
         V0( K ) = WINTPL( VCN, A )

         IF( ALPHA .NE. 0.0 )THEN
         CALL JAWSAL( U0( K ), V0( K ), ALPHA )
         END IF

      END DO

      RETURN
      END

      SUBROUTINE JAWSFL1
************************************************************************
*                                                                      *
*     SPECIFY THE ONSET VELOCITY FOR STREAM FUNCTION CALCULATION       *
*                                                                      *
*******************     NOMENCLATURE     *******************************
*                                                                      *
*     UGINT, VGINT    VELOCITIES FOR STREAM FUNCTION CALCULATIONS      *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' STRM.FIL '
      REAL A( 4 )

      XEDGE = XTOT / 2.0
      YEDGE = YTOT / 2.0

      DO J  = 1, 2
      DO I  = 1, INTG
         PX = XINTG( I, J )
         PY = YINTG( I, J )

      CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
      UU = WINTPL( UCN, A )
      VV = WINTPL( VCN, A )
```

```
            IF( ALPHA .NE. 0.0 )THEN
            CALL JAWSAL( UU, VV, ALPHA )
            END IF

            IF( J .EQ. 1 )THEN
            UGINT( I ) = UU
            ELSE
            VGINT( I ) = VV
            END IF

      END DO
      END DO

      RETURN
      END


      SUBROUTINE SUBAREA( X, Y, PX, PY, A )
************************************************************************
*                                                                      *
*     DETERMINE THE SUB-AREA FOR INTERPOLATION                         *
*                                                                      *
********************    NOMENCLATURE    ******************************
*                                                                      *
*     A(1) - A(4)     FOUR SUB-DIVIDED AREAS                           *
*                                                                      *
************************************************************************
*                                                                      *
      REAL A( 4 )

      XL = ABS( -X - PX )
      XR = ABS(  X - PX )
      YU = ABS(  Y - PY )
      YL = ABS( -Y - PY )

      A( 1 ) = XR * YU
      A( 2 ) = XL * YU
      A( 3 ) = XL * YL
      A( 4 ) = XR * YL

      RETURN
      END

      FUNCTION WINTPL( W, A )
************************************************************************
*                                                                      *
*     DETERMINE THE VELOCITY VALUE BY AREA-WEIGHTED INTERPOLATION      *
*                                                                      *
********************    NOMENCLATURE    ******************************
*                                                                      *
*     W(1) - W(4)     JAWS DATA AT FOUR CORNERS                        *
*                                                                      *
*     WINTPL          INTERPOLATED VELOCITY                            *
*                                                                      *
************************************************************************
*                                                                      *
      REAL W( 4 ), A( 4 )

      SUM1 = 0.0
      SUM2 = 0.0
      DO K = 1, 4
      SUM1 = SUM1 + W( K ) * A( K )
      SUM2 = SUM2 + A( K )
      END DO
```

```
      WINTPL = SUM1 / SUM2

      RETURN
      END

      SUBROUTINE JAWSAL( U, V, A )
************************************************************************
*                                                                      *
*     ADD ANGLE OF ATTACK                                              *
*                                                                      *
************************************************************************
*                                                                      *

      IF( U .EQ. 0.0 )THEN
      B = ACOS( -1.0 ) / 2.0

      ELSEIF( ( U .EQ. 0.0 ) .AND. ( V .EQ. 0.0 ) )THEN
      TYPE*,'ATAN(V/U), U, V = 0'
      STOP 333

      ELSE
      B = ATAN( V / U )
      END IF

      IF( ABS( U ) .LT. 1.0E-6 )THEN
      AU = V

      ELSEIF( ABS( V ) .LT. 1.0E-6 )THEN
      AU = U

      ELSE
      AU = SQRT( U** 2 + V** 2 )
      END IF

      U = AU * COS( B + A )
      V = AU * SIN( B + A )

      RETURN
      END

      SUBROUTINE COEF
************************************************************************
*                                                                      *
*     CALCULATE THE INFLUENCE MATRICES :                               *
*                                                                      *
*******************     NOMENCLATURE     ******************************
*                                                                      *
*     AINTGL1,2        INFLUENCE MATRICES FOR SOURCE IN NORMAL AND     *
*                      TANGENTIAL DIRECTIONS ( C.P. )                  *
*                                                                      *
*     DINTGL1,2        INFLUENCE MATRICES FOR VORTEX IN NORMAL AND     *
*                      TANGENTIAL DIRECTIONS ( C.P. )                  *
*                                                                      *
*     AGINTGL1,2       INFLUENCE MATRICES FOR SOURCE IN NORMAL AND     *
*                      TANGENTIAL DIRECTIONS ( INNER GRIDS )           *
*                                                                      *
*     DGINTGL1,2       INFLUENCE MATRICES FOR VORTEX IN NORMAL AND     *
*                      TANGENTIAL DIRECTIONS ( INNER GRIDS )           *
*                                                                      *
*     AOINTGL1,2       INFLUENCE MATRICES FOR SOURCE IN NORMAL AND     *
*                      TANGENTIAL DIRECTIONS ( INNER GRIDS )           *
*                                                                      *
*     DOINTGL1,2       INFLUENCE MATRICES FOR VORTEX IN NORMAL AND     *
*                      TANGENTIAL DIRECTIONS ( INNER GRIDS )           *
*                                                                      *
************************************************************************
```

```
*                                                                               *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' MTRX.FIL '
      INCLUDE ' NAC4.FIL '

      DO I = 1, NPAN
      DO J = 1, NPAN

            THEIJ = THETA( I ) - THETA( J )
              ALN = 1.0 + ( PL(J)** 2 - 2.0 * XE(I,J) * PL(J) ) /
     1                  ( XE(I,J)** 2 + YE(I,J)** 2 )

************ CHECK CALCULATION   ******************
            IF( ALN .LE. 0.0 ) THEN
            TYPE*, ' ERROR ON LN '
            STOP 111
            END IF

            IF( J .EQ. I ) THEN
              AINTGL1( I, J ) = PI
              AINTGL2( I, J ) = 0.0
              DINTGL1( I, J ) = 0.0
              DINTGL2( I, J ) = PI

            ELSEIF( YE( I, J ) .EQ. 0.0 ) THEN
              AINTGL1( I, J ) = 0.5 * SIN( THEIJ ) * ALOG( ALN )
              AINTGL2( I, J ) = - 0.5 * COS( THEIJ ) * ALOG( ALN )
              DINTGL1( I, J ) = - AINTGL2( I, J )
              DINTGL2( I, J ) = AINTGL1( I, J )

            ELSE
              ATNU = ( PL( J ) - XE( I, J ) ) / YE( I, J )
              ATNL = - XE( I, J ) / YE( I, J )
              AINTGL1( I, J ) = 0.5 * SIN( THEIJ ) * ALOG( ALN ) +
     1               COS( THEIJ ) * ( ATAN( ATNU ) - ATAN( ATNL ) )
              AINTGL2( I, J ) =- 0.5 * COS( THEIJ ) * ALOG( ALN ) +
     1               SIN( THEIJ ) * ( ATAN( ATNU ) - ATAN( ATNL ) )
              DINTGL1( I, J ) = -AINTGL2( I, J )
              DINTGL2( I, J ) = AINTGL1( I, J )

            END IF

      END DO
      END DO

      IF( RCP .NE. 0.5 ) THEN

      DO K = 1, NPAN

      IF( K .GE. NPANH1 )THEN
******* UPPER SURFACE ***************************************************
        AKK = ALOG( RCP / ( 1.0 - RCP ) )

      ELSE
******* LOWER SURFACE ***************************************************
        AKK = ALOG( ( 1.0 - RCP ) / RCP )
      END IF

        AINTGL2( K, K ) = AKK
        DINTGL1( K, K ) = - AINTGL2( K, K )
```

```
      END DO

      END IF

      CALL LASTRW


      IF( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) THEN

      DO I = 1, NGRID
      DO J = 1, NGRID

      IF( ( J .EQ. JCRD ) .AND.
     1      ( ( I .GE. ICRD ) .AND. ( I .LE. ICRD1 ) ) )THEN
      GOTO 1
      END IF

        DO K = 1, NPAN

           THEIJ = 0.0 - THETA( K )
             ALN = 1.0 + ( PL(K)** 2 - 2.0 * XGE(I,J,K) * PL(K) )
     1                / ( XGE(I,J,K)** 2 + YGE(I,J,K)** 2 )

*************  CHECK CALCULATION   ******************
             IF( ALN .LE. 0.0 ) THEN
             TYPE*, ' ERROR ON LN '
             STOP 222
             END IF

              ATNU = ( PL( K ) - XGE( I, J, K ) ) / YGE( I, J, K )
              ATNL = - XGE( I, J, K ) / YGE( I, J, K )

          AGINTGL1( I, J, K ) = 0.5 * SIN( THEIJ ) * ALOG( ALN ) +
     1               COS( THEIJ ) * ( ATAN( ATNU ) - ATAN( ATNL ) )
          AGINTGL2( I, J, K ) = -0.5 * COS( THEIJ ) * ALOG( ALN ) +
     1               SIN( THEIJ ) * ( ATAN( ATNU ) - ATAN( ATNL ) )
          DGINTGL1( I, J, K ) = - AGINTGL2( I, J, K )
          DGINTGL2( I, J, K ) = AGINTGL1( I, J, K )

        END DO
 1      CONTINUE

      END DO
      END DO

      END IF


      RETURN
      END


      SUBROUTINE LASTRW
***********************************************************************
*                                                                     *
*     CONSTRUCT THE LAST ROW OF INFLUENCE MATRICES : I ij, D ij       *
*     FOR KUTTA CONDITION                                             *
*                                                                     *
********************     NOMENCLATURE     ******************************
*                                                                     *
*     A1STORE          LAST ROW FOR SOURCE                            *
*                                                                     *
*     D1STORE          LAST ROW FOR VORTEX                            *
*                                                                     *
***********************************************************************
*                                                                     *
```

```
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' MTRX.FIL '

      DO I = 1, NPAN
      DO J = 1, NPAN
        A1STORE( I, J ) = AINTGL1( I, J )
        D1STORE( I, J ) = DINTGL1( I, J )
      END DO
      END DO


      IF( ( DALPHA .EQ. 0.0 ) .AND. ( ALPHA .EQ. 0.0 ) .AND.
     1     ( IFLOW .EQ. 1 ) .AND. ( IDBL .EQ. 0 ) )THEN

       DO J = 1, NPAN
       A1STORE( NPAN, J ) = AINTGL2( 1, J ) + AINTGL2( NPAN, J )
       END DO

      ELSE

       DO J = 1, NPAN
       D1STORE( NPAN, J ) = DINTGL2( 1, J ) + DINTGL2( NPAN, J )
       END DO

      END IF

      RETURN
      END


      SUBROUTINE COEF1
************************************************************************
*                                                                      *
*     CALCULATE THE INFLUENCE MATRICES : A1 ij, D1 ij, A2 ij, D2 ij    *
*     FOR CALCULATING STREAM FUNCTION                                  *
*                                                                      *
********************     NOMENCLATURE     ******************************
*                                                                      *
*     A1,A2            INFLUENCE MATRICES FOR SOURCE IN NORMAL AND     *
*                      TANGENTIAL DIRECTIONS ( STREAM )                *
*                                                                      *
*     D1,D2            INFLUENCE MATRICES FOR VORTEX IN NORMAL AND     *
*                      TANGENTIAL DIRECTIONS ( STREAM )                *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' STRM.FIL '

      DO K = 1, 2

        IF( K .EQ. 1 )THEN
        IMAX = INTG - 1
        ELSE
        IMAX = INTG
        END IF

        DO J = 1, NPAN
```

```
          DO I = 1, IMAX
            THEIJ = 0.0 - THETA( J )
            ALN   = 1.0 + ( PL(J)** 2 - 2.0 * XSE(I,J,K) * PL(J)
     1                 ) / ( XSE(I,J,K)** 2 + YSE(I,J,K)** 2 )

*************  CHECK CALCULATION   ******************
            IF( ALN .LE. 0.0 ) THEN
            TYPE*, ' ERROR ON LN '
            STOP 444
            END IF

            ATNU = ( PL( J ) - XSE( I, J, K ) ) / YSE( I, J, K )
            ATNL = - XSE( I, J, K ) / YSE( I, J, K )

            A1( I, J, K ) = 0.5 * SIN( THEIJ ) * ALOG( ALN ) +
     1                 COS( THEIJ ) * ( ATAN( ATNU ) - ATAN( ATNL ) )
            A2( I, J, K ) = - 0.5 * COS( THEIJ ) * ALOG( ALN ) +
     1                 SIN( THEIJ ) * ( ATAN( ATNU ) - ATAN( ATNL ) )
            D1( I, J, K ) = -A2( I, J, K )
            D2( I, J, K ) =  A1( I, J, K )

          END DO
          END DO

        END DO

        RETURN
        END

        SUBROUTINE INVERSE( A, AINV )
************************************************************************
*                                                                      *
*       SOLVE SIMULTANEOUS EQUTIONS FOR DIMENSIONLESS SINGULARITY      *
*       STRENGTHS USING IMSL PACKAGE: LINV2F.                          *
*                                                                      *
*       LINK : $LINK FILE, LIB_IMSL_D/LIB                              *
*                                                                      *
********************       NOMENCLATURE      **************************
*                                                                      *
*       A       INPUT LHS COEF. MATRIX ( INPUT )                       *
*                                                                      *
*       N       ORDER OF COEF. MATRIX A                                *
*                                                                      *
*       IA      ROW DIMENSION OF A AND AINV EXACTLY AS                 *
*               SPECIFIED IN THE DIMENSION STATEMENT                   *
*                                                                      *
*       IDGT    NO. GREATER THAN ZERO                                  *
*                                                                      *
*       WKAREA  WORKING ARRAY ( N X N  + 3 X N )                       *
*                                                                      *
*       IER     OUTPUT WARNING                                         *
*                                                                      *
************************************************************************
*                                                                      *
        PARAMETER ( NPAN = 100 )

        INCLUDE ' CNST.FIL '
        REAL A( NPAN, NPAN ), AINV( NPAN, NPAN ),
     1       WKAREA( NPAN*NPAN+3*NPAN )
        INTEGER  N, IA, IDGT, IER

        N = NPAN
        IA = NPAN
        IDGT = 6

        CALL LINV2F( A, N, IA, AINV, IDGT, WKAREA, IER )
```

```
      IF( IER .NE. 0.0 )THEN
      TYPE*, ' IER = ', IER
      TYPE*, ' IDGT = ', IDGT
      END IF

      RETURN
      END


      SUBROUTINE LEQINV ( BB, RR, ALAMBDA )
***********************************************************************
*                                                                     *
*     MULTIPLY BB BY RR FOR DIMENSIONLESS SINGULARITY STRENGTH        *
*                                                                     *
***********************************************************************
*                                                                     *
      PARAMETER ( NPAN = 100 )

      INCLUDE ' CNST.FIL '
      REAL ALAMBDA( NPAN ), RR( NPAN ), BB( NPAN, NPAN )

      DO I = 1, NPAN
      SUM  = 0.0

        DO J = 1, NPAN
        SUM  = SUM + BB( I, J ) * RR( J )
        END DO

      ALAMBDA( I ) = SUM

      END DO

      RETURN
      END

      SUBROUTINE RHS1
***********************************************************************
*                                                                     *
*     CALCULATE THE RHS COLUMN VECTOR  OF SOURCE DISTRIBUTION         *
*                                                                     *
*******************      NOMENCLATURE      ****************************
*                                                                     *
*     RHSC1          RHS COLUMN VECTOR                                *
*                                                                     *
***********************************************************************
*                                                                     *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' MTRX.FIL '
      INCLUDE ' VLTY.FIL '

      DO I = 1, NPAN
      RHSC1( I ) = - ( U0(I)*COS(DALPHA) - V0(I)*SIN(DALPHA) ) *
     1             COS( BETA( I ) + DALPHA )
      END DO

      IF( ( DALPHA .EQ. 0.0 ) .AND. ( ALPHA .EQ. 0.0 ) .AND.
     1     ( IFLOW .EQ. 1 ) .AND. ( IDBL .EQ. 0 ) )THEN
      RHSC1( NPAN ) = - ( U0( NPAN ) * COSINE( NPAN ) + V0( NPAN ) *
     1          SINE( NPAN ) + U0(1) * COSINE( 1 ) + V0(1) * SINE( 1 ) )
      END IF
```

```
      RETURN
      END


      SUBROUTINE RHS2( ICONT )
************************************************************************
*                                                                      *
*     CALCULATE THE RHS COLUMN VECTOR OF VORTEX DISTRIBUTION           *
*                                                                      *
*******************    NOMENCLATURE     *******************************
*                                                                      *
*     RHSC2            RHS COLUMN VECTOR                               *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' MTRX.FIL '
      INCLUDE ' VLTY.FIL '

      IF( ( DALPHA .EQ. 0.0 ) .AND. ( ALPHA .EQ. 0.0 ) .AND.
     1    ( IFLOW .EQ. 1 ) .AND. ( IDBL .EQ. 0 ) )THEN
          RETURN

      ELSE
        DO I = 1, NPAN
          RHSC2( I ) = - ( U0(I)*SIN(DALPHA) + V0(I)*COS(DALPHA) )
     1                   * SIN( BETA( I ) + DALPHA )

          IF( ( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) .AND.
     1          ( ICONT .GT. 1 ) )THEN
          RHSC2( I ) = RHSC2( I ) + RESL( I )
          END IF

        END DO

        SUMN = 0.0
        SUM1 = 0.0
        DO J = 1, NPAN
        SUMN = SUMN + AINTGL2( NPAN, J ) * ASIGAMA( J )
        SUM1 = SUM1 + AINTGL2( 1, J ) * ASIGAMA( J )
        END DO

        IF( ( ( IFLOW .GT. 2 ) .OR. ( IDBL .GT. 2 ) ) .AND.
     1        ( ICONT .GT. 1 ) )THEN
        CCTN = RESLT(NPAN)
        CCT1 = RESLT(1)
        RHSC2(NPAN) = - ( U0(NPAN)*COSINE(NPAN) + V0(NPAN)*SINE(NPAN)
     1                + SUMN + CCTN + U0(1) * COSINE(1) +
     1                V0( 1 ) * SINE( 1 ) + SUM1 + CCT1 )
        ELSE
        RHSC2(NPAN) = - ( U0(NPAN)*COSINE(NPAN) + V0(NPAN)*SINE(NPAN)
     1                + SUMN  +  U0( 1 ) * COSINE( 1 ) +
     1                V0( 1 ) * SINE( 1 ) + SUM1 )
        END IF

      END IF

      RETURN
      END

      FUNCTION RESL( K )
************************************************************************
```

```
*                                                                      *
*       CALCULATE THE RHS RESIDUAL COLUMN VECTOR  OF                   *
*       VORTEX DISTRIBUTION DUE TO NONLINEAR SHEAR FLOW                *
*       IN NORMAL DIRECTION                                            *
*                                                                      *
********************      NOMENCLATURE     *****************************
*                                                                      *
*       RESL              RESIDULE DUE TO THE ONSET VORTICITY          *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' VOTY.FIL '
      REAL F( NGRID, NGRID ), XX( NGRID ), YY( NGRID ),
     1     WK( NGRID*NGRID+11*NGRID )
      INTEGER IFD, NX, NY, IER, K

      IFD = NGRID
       NX = NGRID
       NY = NGRID

      DO I  = 1, NX
      XX(I) = XG( I, 1 )
      YY(I) = YG( 1, I )
      END DO

      A = XX( 1  )
      B = XX( NX )
      C = YY( 1  )
      D = YY( NY )

      DO J = 1, NY
      DO I = 1, NX

        IF( ( J .EQ. JCRD ) .AND.
     1       ( ( I .GE. ICRD ) .AND. ( I .LE. ICRD1 ) ) )THEN
        F(I,J) = 0.0

        ELSE
        F(I,J) = ( OMEGAG( I, J ) - OMEGAG0( I, J ) ) * ( ( X( K ) -
     1             XG( I,J ) ) * COSINE( K ) + ( Y( K ) - YG( I,J ) ) *
     1             SINE( K ) ) / ( ( X( K ) - XG( I, J ) )** 2 +
     1              ( Y( K ) - YG( I, J ) )** 2 )
        END IF

      END DO
      END DO

      CALL DBCQDU( F, IFD, XX, NX, YY, NY, A, B, C, D, Q, WK, IER )
      RESL = - Q / 2.0 / PI

      IF( IER .NE. 0 )THEN
      TYPE*,' IER=', IER
      TYPE*,'RESL=', RESL
      STOP 888
      END IF

      RETURN
      END
```

```
      FUNCTION RESLT( K )
***********************************************************************
*                                                                     *
*     CALCULATE THE RHS RESIDUAL COLUMN VECTOR  OF                    *
*     VORTEX DISTRIBUTION DUE TO NONLINEAR SHEAR FLOW                 *
*     IN TANGENTIAL DIRECTION                                         *
*                                                                     *
*******************     NOMENCLATURE     *****************************
*                                                                     *
*     RESLT            RESIDULE DUE TO THE ONSET VORTICITY            *
*                                                                     *
***********************************************************************
*                                                                     *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' VOTY.FIL '
      REAL F( NGRID, NGRID ), XX( NGRID ), YY( NGRID ),
     1     WK( NGRID*NGRID+11*NGRID )
      INTEGER IFD, NX, NY, IER, K

      IFD = NGRID
       NX = NGRID
       NY = NGRID

      DO I  = 1, NX
      XX(I) = XG( I, 1 )
      YY(I) = YG( 1, I )
      END DO

      A = XX( 1  )
      B = XX( NX )
      C = YY( 1  )
      D = YY( NY )

      DO J = 1, NY
      DO I = 1, NX

        IF( ( J .EQ. JCRD ) .AND.
     1      ( ( I .GE. ICRD ) .AND. ( I .LE. ICRD1 ) ) )THEN
        F(I,J) = 0.0

        ELSE
        F(I,J) = ( OMEGAG( I, J ) - OMEGAG0( I, J ) ) * ( ( X( K ) -
     1            XG( I,J ) ) * SINE( K ) - ( Y( K ) - YG( I,J ) ) *
     1            COSINE( K ) ) / ( ( X( K ) - XG( I, J ) )** 2 +
     1            ( Y( K ) - YG( I, J ) )** 2 )
        END IF

      END DO
      END DO

      CALL DBCQDU( F, IFD, XX, NX, YY, NY, A, B, C, D, Q, WK, IER )
      RESLT = Q / 2.0 / PI

      IF( IER .NE. 0 )THEN
      TYPE*,' IER=', IER
      TYPE*,'RESLT=', RESLT
      STOP 889
      END IF

      RETURN
```

```
      END

      FUNCTION RESLU( X0, Y0 )
************************************************************************
*                                                                      *
*     CALCULATE THE RHS RESIDUAL COLUMN VECTOR  OF                     *
*     VORTEX DISTRIBUTION DUE TO NONLINEAR SHEAR FLOW                  *
*     IN X DIRECTION                                                   *
*                                                                      *
********************      NOMENCLATURE     ******************************
*                                                                      *
*     RESLU            RESIDULE DUE TO THE ONSET VORTICITY             *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' VOTY.FIL '
      REAL F( NGRID, NGRID ), XX( NGRID ), YY( NGRID ),
     1     WK( NGRID*NGRID+11*NGRID )
      INTEGER IFD, NX, NY, IER, K

      IFD = NGRID
       NX = NGRID
       NY = NGRID

      DO I  = 1, NX
      XX(I) = XG( I, 1 )
      YY(I) = YG( 1, I )
      END DO

      A = XX( 1  )
      B = XX( NX )
      C = YY( 1  )
      D = YY( NY )

      DO J = 1, NY
      DO I = 1, NX

        IF( ( J .EQ. JCRD ) .AND.
     1      ( ( I .GE. ICRD ) .AND. ( I .LE. ICRD1 ) ) )THEN
        F(I,J) = 0.0

        ELSEIF( ( X0 .EQ. XG(I,J) ) .AND. ( Y0 .EQ. YG(I,J) ) )THEN
        F(I,J) = 0.0

        ELSE
        F(I,J) = ( OMEGAG( I, J ) - OMEGAG0( I, J ) ) * ( Y0 -
     1           YG( I,J ) ) / ( ( X0 - XG( I, J ) )** 2 +
     1           ( Y0 - YG( I, J ) )** 2 )
        END IF

      END DO
      END DO

      CALL DBCQDU( F, IFD, XX, NX, YY, NY, A, B, C, D, Q, WK, IER )
      RESLU = - Q / 2.0 / PI

      IF( IER .NE. 0 )THEN
      TYPE*,' IER=', IER
      TYPE*,'RESLU=', RESLU
```

```
      STOP 888
      END IF

      RETURN
      END

      FUNCTION RESLV( X0, Y0 )
************************************************************************
*                                                                      *
*     CALCULATE THE RHS RESIDUAL COLUMN VECTOR  OF                     *
*     VORTEX DISTRIBUTION DUE TO NONLINEAR SHEAR FLOW                  *
*     IN Y DIRECTION                                                   *
*                                                                      *
*******************     NOMENCLATURE     ******************************
*                                                                      *
*     RESLV          RESIDULE DUE TO THE ONSET VORTICITY               *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' ANGL.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' VOTY.FIL '
      REAL F( NGRID, NGRID ), XX( NGRID ), YY( NGRID ),
     1     WK( NGRID*NGRID+11*NGRID )
      INTEGER IFD, NX, NY, IER, K

      IFD = NGRID
       NX = NGRID
       NY = NGRID

      DO I  = 1, NX
      XX(I) = XG( I, 1 )
      YY(I) = YG( 1, I )
      END DO

      A = XX( 1  )
      B = XX( NX )
      C = YY( 1  )
      D = YY( NY )

      DO J = 1, NY
      DO I = 1, NX

        IF( ( J .EQ. JCRD ) .AND.
     1      ( ( I .GE. ICRD ) .AND. ( I .LE. ICRD1 ) ) )THEN
        F(I,J) = 0.0

        ELSEIF( ( X0 .EQ. XG(I,J) ) .AND. ( Y0 .EQ. YG(I,J) ) )THEN
        F(I,J) = 0.0

        ELSE
        F(I,J) = ( OMEGAG( I, J ) - OMEGAG0( I, J ) ) * ( X0 -
     1           XG( I,J ) ) / ( ( X0 - XG( I, J ) )** 2 +
     1           ( Y0 - YG( I, J ) )** 2 )
        END IF

      END DO
      END DO

      CALL DBCQDU( F, IFD, XX, NX, YY, NY, A, B, C, D, Q, WK, IER )
      RESLV = Q / 2.0 / PI
```

```
      IF( IER .NE. 0 )THEN
      TYPE*,' IER=', IER
      TYPE*,'RESLV=', RESLV
      STOP 891
      END IF

      RETURN
      END


      SUBROUTINE ERRCHK( AA, BB, ERMAX )
************************************************************************
*                                                                      *
*     FIND THE MAXIMUM DIFFERENCE OF VORTEX STRENGTHS FOR              *
*     TWO DIFFERENT TIME STEPS                                         *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100 )

      INCLUDE ' CNST.FIL '
      REAL AA( NPAN ), BB( NPAN )

      ERMAX = 0.0
      DO I  = 1, NPAN
        DIF = ABS( AA( I ) - BB( I ) )
        IF( DIF .GT. ERMAX )THEN
        ERMAX = DIF
        END IF
      END DO

      RETURN
      END

      SUBROUTINE ONSET
************************************************************************
*                                                                      *
*     SPECIFY THE ONSET FLOW FIELD : ONSET VELOCITY & VORTICITY W.     *
*     IN THE INNER COMPUTATIONAL DOMAIN AND ON THE SURFACE OF          *
*     THE AIRFOIL                                                      *
*                                                                      *
********************     NOMENCLATURE     ******************************
*                                                                      *
*     IFLOW = 1,     UNIFORM FLOW                                      *
*             2,     LINEAR SHEAR FLOW                                 *
*             3,     EXPONENTIAL SHEAR FLOW( NONLINEAR )               *
*             4,     INPUT JAWS DATA( NONLINEAR )                      *
*             5      INPUT DFW DATA ( TASS AXISY. WIND )               *
*                                                                      *
*     U0, V0         ONSET VELOCITY ON THE CONTROL POINT               *
*                                                                      *
*     UG0, VG0       ONSET VELOCITY IN THE DOMAIN                      *
*                                                                      *
*     UG01, VG01     ONSET VELOCITY IN THE FAR UPSTREAM                *
*                                                                      *
*     OMEGAG0        ONSET VORTICITY IN THE DOMAIN                     *
*                                                                      *
* **********************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' AUXI.FIL '
      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
```

```
      INCLUDE ' DBLE.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' STRM.FIL '
      INCLUDE ' VLTY.FIL '
      INCLUDE ' VOTY.FIL '
      REAL AU( NGRID ), AV( NGRID )
      REAL XX( NG101 ), YY( NG101 )
      REAL AX( NGRID ), AY( NGRID ), AZ( NGRID ), C( NGRID-1, 3 )

      GOTO( 100, 100, 200, 300, 400 ), IFLOW


100   CALL UNISFL
      GOTO 500

200   CALL EXPSFL
      GOTO 500

300   CALL JAWSFL
      GOTO 600

400   CALL DFWSFL
      GOTO 600


500   IF( IDBL .EQ. 0 )THEN
      GOTO 600

      ELSEIF( ( IDBL .EQ. 1 ) .OR. ( IDBL .EQ. 2 ) )THEN
      CALL DBLFL

      ELSEIF( IDBL .EQ. 3 )THEN
      CALL DBLNS

      ELSEIF( IDBL .EQ. 4 )THEN
      CALL DBLCS

      ELSEIF( IDBL .EQ. 5 )THEN
      CALL DBLSN
      END IF

      DO J = 1, NGRID
      DO I = 1, NGRID
        IF( IDIR .EQ. 1 )THEN
        UG0( I, J ) = UG0( I, J ) - DUG( I, J )
        VG0( I, J ) = VG0( I, J ) - DVG( I, J )
        ELSE
        UG0( I, J ) = UG0( I, J ) + DUG( I, J )
        VG0( I, J ) = VG0( I, J ) + DVG( I, J )
        END IF
      END DO
      END DO

      DO K = 1, NG101
        IF( IDIR .EQ. 1 )THEN
        UG01( K ) = UG01( K ) - DUG1( K )
        VG01( K ) = VG01( K ) - DVG1( K )
        UG02( K ) = UG02( K ) - DUG2( K )
        VG02( K ) = VG02( K ) - DVG2( K )
        ELSE
        UG01( K ) = UG01( K ) + DUG1( K )
        VG01( K ) = VG01( K ) + DVG1( K )
        UG02( K ) = UG02( K ) + DUG2( K )
        VG02( K ) = VG02( K ) + DVG2( K )
        END IF
      END DO
```

```
      DO K = 1, NPAN
        IF( IDIR .EQ. 1 )THEN
        U0( K ) = U0( K ) - DU( K )
        V0( K ) = V0( K ) - DV( K )
        ELSE
        U0( K ) = U0( K ) + DU( K )
        V0( K ) = V0( K ) + DV( K )
        END IF
      END DO

600   IF( ( IFLOW .LE. 2 ) .AND. ( IDBL .LE. 2 ) )THEN
      RETURN
      END IF

      DO J = 1, NGRID
      DO I = 1, NGRID

      DO K = 1, NGRID
      AU( K ) = UG0( I, K )
      AV( K ) = VG0( K, J )
      END DO

      IF( ( I .EQ. 1 ) .OR. ( I .EQ. 2 ) ) THEN
      CALL ONESD( AV, I, DS, OMEG0 )

      ELSEIF( ( I .EQ. NGRID1 ) .OR. ( I .EQ. NGRID ) ) THEN
      CALL ONESU( AV, I, DS, OMEG0 )

      ELSE
      CALL CNTLD( AV, I, DS, OMEG0 )
      END IF

      OMEGAG0( I, J ) = OMEG0

      IF( ( J .EQ. 1 ) .OR. ( J .EQ. 2 ) ) THEN
      CALL ONESD( AU, J, DS, OMEG0 )

      ELSEIF( ( J .EQ. NGRID1 ) .OR. ( J .EQ. NGRID ) ) THEN
      CALL ONESU( AU, J, DS, OMEG0 )

      ELSE
      CALL CNTLD( AU, J, DS, OMEG0 )
      END IF

      OMEGAG0( I, J ) = OMEGAG0( I, J ) - OMEG0

      END DO
      END DO
************************************************************************
*     SPECIFY THE FARUPSTREAM STREAM FUNCTION                          *

      DO J = 1, NG101
        NY = J
        IY = NY - 1

      IF( IY .EQ. 0 )THEN
      PSI1( J ) = 0.0

      ELSE
      DO K = 1, NY
      XX( K ) = YG1(  K )
      YY( K ) = UG01( K )
      END DO
      A = XX( 1 )
```

```
B = XX( NY )
CALL INTEG1( XX, YY, A, B, IY, NY, Q )
PSI1( J ) = Q
END IF

END DO


DO K = 1, NG101
XX( K ) = 0.0
YY( K ) = 0.0
END DO


IF( ( IFLOW .NE. 3 ) .OR. ( IDBL .NE. 0 ) )THEN

DO I = 1, NG101
  NX = I
  IX = NX - 1

IF( ( IX .EQ. 0 ) .AND. ( IDIR .EQ. 1 ) )THEN
PSI2( I ) = PSI1( NG101 )

ELSEIF( ( IX .EQ. 0 ) .AND. ( IDIR .EQ. 0 ) )THEN
PSI2( I ) = PSI1( 1 )

ELSE
DO K = 1, NX
XX( K ) = XG2(  K )
YY( K ) = VG02( K )
END DO
A = XX( 1 )
B = XX( NX )
CALL INTEG1( XX, YY, A, B, IX, NX, Q )
PSI2( I ) = PSI2( 1 ) - Q
END IF

END DO

END IF

IF( ( IFLOW .EQ. 3 ) .AND. ( IDBL .EQ. 0 ) )THEN
  DO K = 1, NG101
  UREF = YG1( K )/ AREF
  OMEG01( K ) = 2.0 * UY0 * CREF * EXP( -UREF**2 ) * UREF / AREF
  END DO
  GOTO 999
ELSE
  DO J = 1, NGRID
  AY( J ) = YG( 1, J )
  AZ( J ) = OMEGAG0( 1, J )
  END DO

  DO K = 1, NG101
  DO J = 1, NGRID-1
    J1 = J + 1

  IF( AY( J ) .EQ. YG1( K ) )THEN
  OMEG01( K ) = OMEGAG0( 1, J )
  GOTO 700

  ELSEIF( ( AY( J  ) .LT. YG1( K ) ) .AND.
1         ( AY( J1 ) .GT. YG1( K ) ) )THEN
   N = NGRID
  IC = N - 1
   D = YG1( K ) - AY( J )
```

```
          CALL ICSCCU( AY, AZ, N, C, IC, IER )
          OMEG01( K ) = ( ( C(J,3) * D + C(J,2) ) * D + C(J,1) ) * D +
     1                   AZ( J )
          GOTO 700

          END IF

          END DO
 700      CONTINUE
          END DO

        END IF


        IF( ( IDBL .GT. 2 ) .AND. ( IDIR .EQ. 1 ) )THEN

          DO I = 1, NGRID
          AX( I ) = XG( I, NGRID )
          AZ( I ) = OMEGAG0( I, NGRID )
          END DO

          DO K = 1, NG101
          DO I = 1, NGRID-1
            I1 = I + 1
          IF( AX( I ) .EQ. XG2( K ) )THEN
          OMEG02( K ) = OMEGAG0( I, NGRID )
          GOTO 800

          ELSEIF( ( AX( I  ) .LT. XG2( K ) ) .AND.
     1            ( AX( I1 ) .GT. XG2( K ) ) )THEN
           N = NGRID
          IC = N - 1
           D = XG2( K ) - AX( I )
          CALL ICSCCU( AX, AZ, N, C, IC, IER )
          OMEG02( K ) = ( ( C(I,3) * D + C(I,2) ) * D + C(I,1) ) * D +
     1                   AZ( I )
          GOTO 800

          END IF

          END DO
 800      CONTINUE
          END DO

        ELSEIF( ( IDBL .GT. 2 ) .AND. ( IDIR .EQ. 0 ) )THEN
          DO I = 1, NGRID
          AX( I ) = XG( I, 1 )
          AZ( I ) = OMEGAG0( I, 1 )
          END DO

          DO K = 1, NG101
          DO I = 1, NGRID-1
            I1 = I + 1
          IF( AX( I ) .EQ. XG2( K ) )THEN
          OMEG02( K ) = OMEGAG0( I, 1 )
          GOTO 900

          ELSEIF( ( AX( I  ) .LT. XG2( K ) ) .AND.
     1            ( AX( I1 ) .GT. XG2( K ) ) )THEN
           N = NGRID
          IC = N - 1
           D = XG2( K ) - AX( I )
          CALL ICSCCU( AX, AZ, N, C, IC, IER )
          OMEG02( K ) = ( ( C(I,3) * D + C(I,2) ) * D + C(I,1) ) * D +
     1                   AZ( I )
          GOTO 900
```

```
          END IF

          END DO
900       CONTINUE
          END DO

        END IF

999     RETURN
        END

        SUBROUTINE UNISFL
***********************************************************************
*                                                                     *
*       SPECIFY ONSET UNIFORM FLOW AND LINEAR SHEAR FLOW              *
*                                                                     *
*******************       NOMENCLATURE       **************************
*                                                                     *
*      U0, V0         ONSET VELOCITY ON THE CONTROL POINT             *
*                                                                     *
*      UG0, VG0       ONSET VELOCITY IN THE DOMAIN                    *
*                                                                     *
*      UG01, VG01     ONSET VELOCITY IN THE FAR UPSTREAM              *
*                                                                     *
***********************************************************************
*                                                                     *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' ROTE.FIL '
        INCLUDE ' VLTY.FIL '

        IF ( ALPHA .EQ. 0.0 )THEN
          DO J = 1, NGRID
          DO I = 1, NGRID
          UG0( I, J ) = UY0 - ZETA0 * YG( I, J )
          VG0( I, J ) = 0.0
          END DO
          END DO

          DO K = 1, NG101
          UG01( K ) = UY0 - ZETA0 * YG1( K )
          VG01( K ) = 0.0
          UG02( K ) = UY0 - ZETA0 * YG2( K )
          VG02( K ) = 0.0
          END DO

          DO K = 1, NPAN
          U0( K ) = UY0 - ZETA0 * Y( K )
          V0( K ) = 0.0
          END DO

        ELSE
          CALL ROTATE( ALPHA )
          DO J = 1, NGRID
          DO I = 1, NGRID
          UROT = UY0 - ZETA0 * YIJ( I, J )
          UG0( I, J ) = UROT * COS( ALPHA )
          VG0( I, J ) = UROT * SIN( ALPHA )
          END DO
          END DO

          DO K = 1, NG101
```

```
          UROT = UY0 - ZETA0 * YIJ1( K )
          UG01( K ) = UROT * COS( ALPHA )
          VG01( K ) = UROT * SIN( ALPHA )
          UROT = UY0 - ZETA0 * YIJ2( K )
          UG02( K ) = UROT * COS( ALPHA )
          VG02( K ) = UROT * SIN( ALPHA )
          END DO

          CALL ROTJR( ALPHA )
          DO K = 1, NPAN
          UROT = UY0 - ZETA0 * YI( K )
          U0( K ) = UROT * COS( ALPHA )
          V0( K ) = UROT * SIN( ALPHA )
          END DO

        END IF

        RETURN
        END

        SUBROUTINE EXPSFL
************************************************************************
*                                                                      *
*       SPECIFY ONSET EXPONENTIAL SHEAR FLOW                           *
*                                                                      *
********************      NOMENCLATURE      ****************************
*                                                                      *
*       U0, V0         ONSET VELOCITY ON THE CONTROL POINT             *
*                                                                      *
*       UG0, VG0       ONSET VELOCITY IN THE DOMAIN                    *
*                                                                      *
*       UG01, VG01     ONSET VELOCITY IN THE FAR UPSTREAM              *
*                                                                      *
************************************************************************
*                                                                      *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' AUXI.FIL '
        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' VLTY.FIL '

        DO J = 1, NGRID
        DO I = 1, NGRID
        UREF = YG( I, J )/ AREF
          UG0( I, J ) = UY0 * ( 1.0D0 + CREF * EXP( - UREF** 2 ) )
          VG0( I, J ) = 0.0
        END DO
        END DO

        DO K = 1, NG101
          UREF = YG1( K )/ AREF
          UG01( K ) = UY0 * ( 1.0 + CREF * EXP( - UREF** 2 ) )
          VG01( K ) = 0.0
          UREF = YG2( K )/ AREF
          UG02( K ) = UY0 * ( 1.0 + CREF * EXP( - UREF** 2 ) )
          VG02( K ) = 0.0
        END DO

        DO K = 1, NPAN
          UREF = Y( K )/ AREF
          U0( K ) = UY0 * ( 1.0 + CREF * EXP( - UREF** 2 ) )
          V0( K ) = 0.0
        END DO
```

```
      RETURN
      END


      SUBROUTINE DFWSFL
*********************************************************************
*                                                                   *
*     READ ONSET FLOW VELOCITY AT 4 CORNERS FROM TASS DATA AND THEN *
*     INTERPOLATE THE ONSET FLOW VELOCITY AT ANY POINT IN THE DOMAIN *
*     USING SUB-AREA AS WEIGHTING PARAMETER :                       *
*                                                                   *
*                  4 __________________ 3                           *
*                    l   A2    l  A1  l                             *
*                    l_________x______l                             *
*                    l        pl      l                             *
*                    l   A3    l  A4  l                             *
*                    l_________l______l                             *
*                  1                    2                           *
*                                                                   *
*                    Wp = ( Wi * Ai ) / ( Aj )  ( i, j = 1, 4 )     *
*                                                                   *
*******************    NOMENCLATURE    *****************************
*                                                                   *
*     U0, V0         ONSET VELOCITY ON THE CONTROL POINT            *
*                                                                   *
*     UG0, VG0       ONSET VELOCITY IN THE DOMAIN                   *
*                                                                   *
*     UG01, VG01     ONSET VELOCITY IN THE FAR UPSTREAM             *
*                                                                   *
*********************************************************************
*                                                                   *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' DFWS.FIL '
      INCLUDE ' VLTY.FIL '
      REAL A( 4 )

      XEDGE = S / 2.0
      YEDGE = S / 2.0
      DO  K = 1, NG101

         PX = XG1( K )
         PY = YG1( K )
         CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
         UG01( K ) = WINTPL( UCN1, A )
         VG01( K ) = WINTPL( VCN1, A )

         IF( ALPHA .NE. 0.0 )THEN
         CALL DFWSAL( UG01( K ), VG01( K ), ALPHA )
         END IF

         PX = XG2( K )
         PY = YG2( K )
         CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
         UG02( K ) = WINTPL( UCN1, A )
         VG02( K ) = WINTPL( VCN1, A )

         IF( ALPHA .NE. 0.0 )THEN
         CALL DFWSAL( UG02( K ), VG02( K ), ALPHA )
         END IF

      END DO
```

```
      DO  I = 1, NGRID
      DO  J = 1, NGRID

         PX = XG( I, J )
         PY = YG( I, J )
         CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
         UG0( I, J ) = WINTPL( UCN1, A )
         VG0( I, J ) = WINTPL( VCN1, A )

         IF( ALPHA .NE. 0.0 )THEN
         CALL DFWSAL( UG0( I, J ), VG0( I, J ), ALPHA )
         END IF

      END DO
      END DO

      DO  K = 1, NPAN

         PX = X( K )
         PY = Y( K )
         CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
         U0( K ) = WINTPL( UCN1, A )
         V0( K ) = WINTPL( VCN1, A )

         IF( ALPHA .NE. 0.0 )THEN
         CALL DFWSAL( U0( K ), V0( K ), ALPHA )
         END IF

      END DO

      RETURN
      END

      SUBROUTINE DFWSFL1
************************************************************************
*                                                                      *
*     SPECIFY THE ONSET VELOCITY FOR STREAM FUNCTION CALCULATION       *
*                                                                      *
********************      NOMENCLATURE      ****************************
*                                                                      *
*     UGINT, VGINT    VELOCITY FOR STREAM FUNCTION CALCULATION         *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' DFWS.FIL '
      INCLUDE ' STRM.FIL '
      REAL A( 4 )

      XEDGE = S / 2.0
      YEDGE = S / 2.0

      DO J  = 1, 2
      DO I  = 1, NG100
         PX = XINTG( I, J )
         PY = YINTG( I, J )

      CALL SUBAREA( XEDGE, YEDGE, PX, PY, A)
      UU = WINTPL( UCN1, A )
      VV = WINTPL( VCN1, A )

         IF( ALPHA .NE. 0.0 )THEN
         CALL DFWSAL( UU, VV, ALPHA )
```

```
            END IF

            IF( J .EQ. 1 )THEN
            UGINT( I ) = UU
            ELSE
            VGINT( I ) = VV
            END IF

         END DO
         END DO

         RETURN
         END

         SUBROUTINE DFWSAL( U, V, A )
************************************************************************
*                                                                      *
*        ADD ANGLE OF ATTACK                                           *
*                                                                      *
************************************************************************
*                                                                      *
         IF( U .EQ. 0.0 )THEN
         B = ACOS( -1.0 ) / 2.0

         ELSEIF( ( U .EQ. 0.0 ) .AND. ( V .EQ. 0.0 ) )THEN
         TYPE*,'ATAN(V/U), U, V = 0'
         STOP 999

         ELSE
         B = ATAN( V / U )
         END IF

         IF( ABS( U ) .LT. 1.0E-6 )THEN
         AU = V

         ELSEIF( ABS( V ) .LT. 1.0E-6 )THEN
         AU = U

         ELSE
         AU = SQRT( U** 2 + V** 2 )
         END IF

         U = AU * COS( B + A )
         V = AU * SIN( B + A )

         RETURN
         END

         SUBROUTINE ONSET1
************************************************************************
*                                                                      *
*        SPECIFY THE ONSET VELOCITY FOR STREAM FUNCTION CALCULATION    *
*                                                                      *
*******************     NOMENCLATURE      *****************************
*                                                                      *
*        UGINT, VGINT    VELOCITY FOR STREAM FUNCTION CALCULATION      *
*                                                                      *
*                                                                      *
************************************************************************
*                                                                      *
         PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
         PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

         INCLUDE ' AUXI.FIL '
         INCLUDE ' CNST.FIL '
         INCLUDE ' CORD.FIL '
```

```
      INCLUDE ' DBLE.FIL '
      INCLUDE ' JAWS.FIL '
      INCLUDE ' JOUK.FIL '
      INCLUDE ' STRM.FIL '
      INCLUDE ' VLTY.FIL '
      REAL YY1( NG100 ), YY2( NG100 )

      GOTO( 100, 100, 200, 300, 400 ), IFLOW

100   IF ( ALPHA .EQ. 0.0 )THEN
        DO K = 1, NG100
          UGINT( K ) = UY0 - ZETA0 * YINTG( K, 1 )
          VGINT( K ) = 0.0
        END DO

      ELSE
        DO I = 1, NG100
        YY1(I) = -XINTG(I,1) * SIN(ALPHA) + YINTG(I,1) * COS(ALPHA)
        YY2(I) = -XINTG(I,2) * SIN(ALPHA) + YINTG(I,2) * COS(ALPHA)
        END DO
        DO K = 1, NG100
          UROT1 = UY0 - ZETA0 * YY1( K )
          UROT2 = UY0 - ZETA0 * YY2( K )
          UGINT( K ) = UROT1 * COS( ALPHA )
          VGINT( K ) = UROT2 * SIN( ALPHA )
        END DO
      END IF
      GOTO 500

200   DO K  = 1, NG100
      UROT1 = YINTG( K, 1 )/ AREF
          IF( ABS( UROT1 )  .GT. 7.0 )THEN
          UGINT( K ) = UY0
          ELSE
          UGINT( K ) = UY0 * ( 1.0 + CREF * EXP( - UROT1** 2 ) )
          END IF
          VGINT( K ) = 0.0
      END DO
      GOTO 500

300   CALL JAWSFL1
      GOTO 600

400   CALL DFWSFL1
      GOTO 600

 500  IF( IDBL .EQ. 0 )THEN
      GOTO 600
      ELSEIF( ( IDBL .EQ. 1 ) .OR. ( IDBL .EQ. 2 ) )THEN
      CALL DBLFL1
      ELSEIF( IDBL .EQ. 3 )THEN
      CALL DBLNS1
      ELSEIF( IDBL .EQ. 4 )THEN
      CALL DBLCS1
      ELSEIF( IDBL .EQ. 5 )THEN
      CALL DBLSN1
      END IF

      DO K = 1, NG100
        IF( IDIR .EQ. 1 )THEN
        UGINT( K ) = UGINT( K ) - DUGINT( K )
        VGINT( K ) = VGINT( K ) - DVGINT( K )
        ELSE
        UGINT( K ) = UGINT( K ) + DUGINT( K )
        VGINT( K ) = VGINT( K ) + DVGINT( K )
        END IF
```

```
        END DO

600     RETURN
        END

        SUBROUTINE ROTATE( ANGL )
***********************************************************************
*                                                                     *
*       ROTATE THE COORDINATES OF GRID PTS. OF A SQUARE DOMAIN        *
*       FOR CALCULATING THE ONSET VELOCITY FIELD W. ALPHA             *
*                                                                     *
***********************************************************************
*                                                                     *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' ROTE.FIL '

        DO J = 1, NGRID
        DO I = 1, NGRID
            YIJ( I,J ) = -XG(I,J) * SIN(ANGL) + YG(I,J) * COS(ANGL)
        END DO
        END DO

        DO K = 1, NG101
          YIJ1( K ) = -XG1( K ) * SIN( ANGL ) + YG1( K ) * COS( ANGL )
          YIJ2( K ) = -XG2( K ) * SIN( ANGL ) + YG2( K ) * COS( ANGL )
        END DO

        RETURN
        END


        SUBROUTINE ROTJR( ANGL )
***********************************************************************
*                                                                     *
*       ROTATE THE COORDINATES OF CONTROL PTS. ON THE SURFACE OF      *
*       THE AIRFOIL FOR CALCULATING THE ONSET VELOCITY W. ALPHA       *
*                                                                     *
***********************************************************************
*                                                                     *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' ROTE.FIL '

        DO I = 1, NPAN
            YI( I ) = -X( I ) * SIN(ANGL) + Y( I ) * COS(ANGL)
        END DO

        RETURN
        END

        SUBROUTINE DBLFL
***********************************************************************
*                                                                     *
*       SPECIFY THE CROSS FLOW FIELD ( 2ND ONSET FLOW FIELD ):        *
*       UNIFORM OR LINEAR SHEAR                                       *
*                                                                     *
********************        NOMENCLATURE      *************************
*                                                                     *
*       DU, DV            DRAFT FLOW ON THE CONTROL POINT             *
```

```
*                                                                        *
*       DUG, DVG        DRAFT FLOW ON THE GRID POINT                     *
*                                                                        *
*       DUG1, DVG1      DRAFT FLOW ON THE FAR UPSTREAM POINT             *
*                                                                        *
**************************************************************************
*                                                                        *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' DBLE.FIL '

        DO J = 1, NGRID
        DO I = 1, NGRID
        DUG( I, J ) = 0.0
        DVG( I, J ) = VX0 + ZETA1 * XG( I, J )
        END DO
        END DO

        DO K = 1, NG101
        DUG1( K ) = 0.0
        DVG1( K ) = VX0 + ZETA1 * XG1( K )
        DUG2( K ) = 0.0
        DVG2( K ) = VX0 + ZETA1 * XG2( K )
        END DO

        DO K = 1, NPAN
        DU( K ) = 0.0
        DV( K ) = VX0 + ZETA1 * X( K )
        END DO

        RETURN
        END


        SUBROUTINE DBLFL1
**************************************************************************
*                                                                        *
*       SPECIFY THE CROSS FLOW FIELD ( 2ND ONSET FLOW FIELD ):           *
*       UNIFORM OR LINEAR SHEAR                                          *
*                                                                        *
********************       NOMENCLATURE      *****************************
*                                                                        *
*       DUGINT, DVGINT  DRAFT FLOW FOR THE STREAM FUNCTION CALCULATION *
*                                                                        *
**************************************************************************
*                                                                        *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' DBLE.FIL '
        INCLUDE ' STRM.FIL '

        DO K = 1, NG100
        DUGINT( K ) = 0.0
        DVGINT( K ) = VX0 + ZETA1 * XINTG( K, 2 )
        END DO

        RETURN
        END

        SUBROUTINE DBLNS
**************************************************************************
```

```
*                                                                     *
*       SPECIFY THE CROSS FLOW FIELD ( 2ND ONSET FLOW FIELD ):        *
*       EXPONENTIAL SHEAR FLOW                                        *
*                                                                     *
********************     NOMENCLATURE     ******************************
*                                                                     *
*       DU, DV          DRAFT FLOW ON THE CONTROL POINT               *
*                                                                     *
*       DUG, DVG        DRAFT FLOW ON THE GRID POINT                  *
*                                                                     *
*       DUG1, DVG1      DRAFT FLOW ON THE FAR UPSTREAM POINT          *
*                                                                     *
***********************************************************************
*                                                                     *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' DBLE.FIL '

        DO J = 1, NGRID
        DO I = 1, NGRID
          UROT = XG( I, J )/ AREF1
          DVG( I, J ) = VX0 * ( 1.0 + CREF1 * EXP( - UROT** 2 ) )
          DUG( I, J ) = 0.0
        END DO
        END DO

        DO K = 1, NG101
          UROT = XG1( K )/ AREF1
          DVG1( K ) = VX0 * ( 1.0 + CREF1 * EXP( - UROT** 2 ) )
          DUG1( K ) = 0.0
          UROT = XG2( K )/ AREF1
          DVG2( K ) = VX0 * ( 1.0 + CREF1 * EXP( - UROT** 2 ) )
          DUG2( K ) = 0.0
        END DO

        DO K = 1, NPAN
          UROT = X( K )/ AREF1
          DV( K ) = VX0 * ( 1.0 + CREF1 * EXP( - UROT** 2 ) )
          DU( K ) = 0.0
        END DO

        RETURN
        END

        SUBROUTINE DBLNS1
***********************************************************************
*                                                                     *
*       SPECIFY THE CROSS FLOW FIELD ( 2ND ONSET FLOW FIELD ):        *
*       EXPONENTIAL TYPE CROSS FLOW                                   *
*                                                                     *
********************     NOMENCLATURE     ******************************
*                                                                     *
*       DUGINT, DVGINT  DRAFT FLOW FOR THE STREAM FUNCTION CALCULATION *
*                                                                     *
***********************************************************************
*                                                                     *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' DBLE.FIL '
        INCLUDE ' STRM.FIL '
```

```
      DO K = 1, NG100
       UROT1 = XINTG( K, 2 )/ AREF1
       DVGINT( K ) = VX0 * ( 1.0 + CREF1 * EXP( - UROT1** 2 ) )
       DUGINT( K ) = 0.0
      END DO

      RETURN
      END


      SUBROUTINE DBLCS
**********************************************************************
*                                                                    *
*     SPECIFY THE CROSS FLOW FIELD ( 2ND ONSET FLOW FIELD ):         *
*     COSINE SHEAR FLOW                                              *
*                                                                    *
********************      NOMENCLATURE     ***************************
*                                                                    *
*     DU, DV           DRAFT FLOW ON THE CONTROL POINT               *
*                                                                    *
*     DUG, DVG         DRAFT FLOW ON THE GRID POINT                  *
*                                                                    *
*     DUG1, DVG1       DRAFT FLOW ON THE FAR UPSTREAM POINT          *
*                                                                    *
**********************************************************************
*                                                                    *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' CORD.FIL '
      INCLUDE ' DBLE.FIL '

      DO J = 1, NGRID
      DO I = 1, NGRID
        AA = ( XG( I, J ) + SEDGE ) * PI / 20.0
        DVG( I, J ) = VX0 * ( 1.0 - COS( AA ) )
        DUG( I, J ) = 0.0
      END DO
      END DO

      DO K = 1, NG101
        AA = ( XG1( K ) + SEDGE ) * PI / 20.0
        DVG1( K ) = VX0 * ( 1.0 - COS( AA ) )
        DUG1( K ) = 0.0
        AA = ( XG2( K ) + SEDGE ) * PI / 20.0
        DVG2( K ) = VX0 * ( 1.0 - COS( AA ) )
        DUG2( K ) = 0.0
      END DO

      DO K = 1, NPAN
        AA = ( X( K ) + SEDGE ) * PI / 20.0
        DV( K ) = VX0 * ( 1.0 - COS( AA ) )
        DU( K ) = 0.0
      END DO

      RETURN
      END

      SUBROUTINE DBLCS1
**********************************************************************
*                                                                    *
*     SPECIFY THE CROSS FLOW FIELD ( 2ND ONSET FLOW FIELD ):         *
*     COSINE TYPE CROSS FLOW                                         *
*                                                                    *
********************      NOMENCLATURE     ***************************
```

```
*                                                                       *
*       DUGINT, DVGINT  DRAFT FLOW FOR THE STREAM FUNCTION CALCULATION *
*                                                                       *
*************************************************************************
*                                                                       *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' DBLE.FIL '
        INCLUDE ' STRM.FIL '

        DO K = 1, NG100
          AA = ( XINTG( K, 2 ) + SEDGE ) * PI / 20.0
          DVGINT( K ) = VX0 * ( 1.0 - COS( AA ) )
          DUGINT( K ) = 0.0
        END DO

        RETURN
        END


        SUBROUTINE DBLSN
*************************************************************************
*                                                                       *
*       SPECIFY THE CROSS FLOW FIELD ( 2ND ONSET FLOW FIELD ):          *
*       SINE SHEAR FLOW                                                 *
*                                                                       *
********************     NOMENCLATURE     ******************************
*                                                                       *
*       DU, DV          DRAFT FLOW ON THE CONTROL POINT                 *
*                                                                       *
*       DUG, DVG        DRAFT FLOW ON THE GRID POINT                    *
*                                                                       *
*       DUG1, DVG1      DRAFT FLOW ON THE FAR UPSTREAM POINT            *
*                                                                       *
*************************************************************************
*                                                                       *
        PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
        PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

        INCLUDE ' CNST.FIL '
        INCLUDE ' CORD.FIL '
        INCLUDE ' DBLE.FIL '

        DO J = 1, NGRID
        DO I = 1, NGRID
          AA = ( XG( I, J ) + SEDGE ) * PI / 20.0
          DVG( I, J ) = VX0 * ( 1.0 - SIN( AA ) )
          DUG( I, J ) = 0.0
        END DO
        END DO

        DO K = 1, NG101
          AA = ( XG1( K ) + SEDGE ) * PI / 20.0
          DVG1( K ) = VX0 * ( 1.0 - SIN( AA ) )
          DUG1( K ) = 0.0
          AA = ( XG2( K ) + SEDGE ) * PI / 20.0
          DVG2( K ) = VX0 * ( 1.0 - SIN( AA ) )
          DUG2( K ) = 0.0
        END DO

        DO K = 1, NPAN
          AA = ( X( K ) + SEDGE ) * PI / 20.0
          DV( K ) = VX0 * ( 1.0 - SIN( AA ) )
          DU( K ) = 0.0
```

```
      END DO

      RETURN
      END

      SUBROUTINE DBLSN1
************************************************************************
*                                                                      *
*     SPECIFY THE CROSS FLOW FIELD ( 2ND ONSET FLOW FIELD ):           *
*     SINE TYPE CROSS FLOW                                             *
*                                                                      *
********************    NOMENCLATURE    ********************************
*                                                                      *
*     DUGINT, DVGINT  DRAFT FLOW FOR THE STREAM FUNCTION CALCULATION  *
*                                                                      *
************************************************************************
*                                                                      *
      PARAMETER ( NPAN = 100, NPAN1 = 101, NPAN2 = 102, NPANH1 = 51 )
      PARAMETER ( NGRID = 41, NG101 = 101, NG100 = 100 )

      INCLUDE ' CNST.FIL '
      INCLUDE ' DBLE.FIL '
      INCLUDE ' STRM.FIL '

      DO K = 1, NG100
        AA = ( XINTG( K, 2 ) + SEDGE ) * PI / 20.0
        DVGINT( K ) = VX0 * ( 1.0 - SIN( AA ) )
        DUGINT( K ) = 0.0
      END DO

      RETURN
      END
99
```

APPENDIX E

INPUT AND OUTPUT FILES FOR THE COMPUTER PROGRAM

The following files are the input (AIRFOIL.DAT) and output (AIRFOIL.OUT) files for a camber Joukowski airfoil in Gaussian type shear flow,

$U(y) = 3.5\ [1 + 2.765\ exp(-y^2)]$

```
$INPUT1
 IFLOW = 3,
  IDBL = 0,
  IAIR = 5,
  IOUT = 1,
 ALPHA = 0.0,
 ZETA0 = 0.0,
   UY0 = 3.5,
     S = 10.0,
  IMAX = 100,
 ERROR = 1.0E-4,
   RCP = 0.6,
  IDIR = 0
$END
$INPUT2
    AM = 0.0,
    AP = 0.0,
    AT = 0.0
$END
$INPUT3
  AREF = 1.0,
  CREF = 2.765
$END
$INPUT4
 AREF1 = 0.0,
 CREF1 = 0.0,
 ZETA1 = 0.0,
   VX0 = 0.1405408347
$END
$INPUT5
  YREF = -0.368
$END
$INPUT6
    XTOT = 50.0,
    YTOT = 50.0,
UCN( 1 ) = -12.53,
UCN( 2 ) = -12.53,
UCN( 3 ) = -15.02,
UCN( 4 ) = -15.02,
VCN( 1 ) = -0.17,
VCN( 2 ) = -0.17,
VCN( 3 ) = -0.33,
VCN( 4 ) = -0.33,
    VREF = 194.2
$END
$INPUT7
UCN1( 1 ) = -13.96,
UCN1( 2 ) = -13.73,
UCN1( 3 ) = -13.72,
UCN1( 4 ) = -13.71,
VCN1( 1 ) = 0.0,
VCN1( 2 ) = 0.0,
VCN1( 3 ) = -0.60,
VCN1( 4 ) = 0.24,
    VREF1 = 194.2
$END
```

```
$INPUT1
IFLOW    =          3,
IDBL     =          0,
IAIR     =          5,
IOUT     =          1,
ALPHA    =  0.0000000E+00,
ZETA0    =  0.0000000E+00,
UY0      =   3.500000     ,
S        =   10.00000     ,
IMAX     =        100,
ERROR    =  9.9999997E-05,
RCP      =  0.6000000     ,
IDIR     =          0
$END
$INPUT3
AREF     =   1.000000     ,
CREF     =   2.765000
$END
NPAN =    100
NGRID =    41
1
```

PANEL CONFIGURATION & STRENGTH :

I	X(I)	Y(I)	XB(I)	YB(I)	PL(I)	THETA(I)	SIGAMA	AGAMA	CP
1	0.9992	0.0002	1.0000	0.0000	0.0020	-3.3313	-1.8003	0.0296	0.2629
2	0.9963	0.0007	0.9980	0.0004	0.0042	-3.3206	-1.9445	0.0431	0.2526
3	0.9913	0.0016	0.9938	0.0012	0.0064	-3.3087	-2.0010	0.0530	0.2757
4	0.9841	0.0028	0.9875	0.0022	0.0087	-3.2961	-2.0490	0.0607	0.2987
5	0.9746	0.0042	0.9789	0.0036	0.0109	-3.2827	-2.0801	0.0673	0.3168
6	0.9630	0.0058	0.9681	0.0051	0.0131	-3.2687	-2.0911	0.0737	0.3282
7	0.9491	0.0075	0.9552	0.0068	0.0152	-3.2543	-2.1003	0.0800	0.3371
8	0.9332	0.0092	0.9401	0.0085	0.0173	-3.2394	-2.1160	0.0863	0.3447
9	0.9152	0.0108	0.9229	0.0102	0.0193	-3.2242	-2.1384	0.0926	[illegible]
10	0.8952	0.0123	0.9037	0.0118	0.0212	-3.2088	-2.1656	0.0990	0.3551
11	0.8734	0.0137	0.8825	0.0132	0.0230	-3.1933	-2.1962	0.1056	0.3576
12	0.8497	0.0147	0.8596	0.0144	0.0247	-3.1775	-2.2292	0.1124	[illegible]
13	0.8245	0.0155	0.8350	0.0153	0.0262	-3.1626	-2.2639	0.1195	[illegible]
14	0.7977	0.0159	0.8088	0.0158	0.0276	-3.1470	-2.2997	0.1269	[illegible]
15	0.7696	0.0159	0.7812	0.0160	0.0288	-3.1330	-2.3362	0.1348	[illegible]
16	0.7404	0.0155	[illegible]	0.0157	[illegible]	[illegible]	-2.3750	[illegible]	[illegible]
[illegible]	0.7101	[illegible]	[illegible]	0.0151	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	0.6791	[illegible]	[illegible]	0.0133	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

19	0.6473	0.0116	0.6601	0.0124	0.0321	-3.0807	-2.4807	0.1732	0.3158
20	0.6151	0.0095	0.6281	0.0104	0.0325	-3.0697	-2.5143	0.1844	0.3029
21	0.5826	0.0070	0.5956	0.0081	0.0327	-3.0598	-2.5459	0.1964	0.2883
22	0.5499	0.0042	0.5630	0.0054	0.0328	-3.0509	-2.5751	0.2091	0.2718
23	0.5173	0.0012	0.5303	0.0025	0.0327	-3.0433	-2.6015	0.2229	0.2537
24	0.4849	-0.0021	0.4978	-0.0007	0.0325	-3.0370	-2.6246	0.2379	0.2342
25	0.4527	-0.0055	0.4655	-0.0041	0.0321	-3.0322	-2.6440	0.2543	0.2132
26	0.4211	-0.0091	0.4336	-0.0076	0.0315	-3.0288	-2.6593	0.2724	0.1909
27	0.3900	-0.0126	0.4023	-0.0112	0.0309	-3.0270	-2.6700	0.2919	0.1670
28	0.3596	-0.0161	0.3716	-0.0147	0.0301	-3.0269	-2.6756	0.3129	0.1415
29	0.3301	-0.0195	0.3417	-0.0182	0.0292	-3.0286	-2.6758	0.3355	0.1144
30	0.3014	-0.0227	0.3126	-0.0215	0.0283	-3.0321	-2.6700	0.3602	0.0859
31	0.2737	-0.0257	0.2845	-0.0245	0.0272	-3.0377	-2.6577	0.3873	0.0562
32	0.2471	-0.0284	0.2575	-0.0274	0.0260	-3.0455	-2.6382	0.4175	0.0258
33	0.2217	-0.0307	0.2316	-0.0299	0.0248	-3.0556	-2.6110	0.4513	-0.0052
34	0.1974	-0.0327	0.2068	-0.0320	0.0236	-3.0683	-2.5752	0.4894	-0.0364
35	0.1744	-0.0342	0.1833	-0.0337	0.0223	-3.0839	-2.5295	0.5319	-0.0677
36	0.1528	-0.0353	0.1611	-0.0350	0.0209	-3.1027	-2.4728	0.5802	-0.0988
37	0.1324	-0.0359	0.1402	-0.0358	0.0195	-3.1251	-2.4032	0.6342	-0.1288
38	0.1134	-0.0361	0.1207	-0.0361	0.0181	-3.1518	-2.3183	0.6972	-0.1566
39	0.0959	-0.0357	0.1026	-0.0360	0.0167	-3.1835	-2.2151	0.7675	-0.1825
40	0.0798	-0.0348	0.0859	-0.0353	0.0153	-3.2213	-2.0889	0.8544	-0.2010
41	0.0651	-0.0333	0.0706	-0.0340	0.0139	-3.2666	-1.9335	0.9484	-0.2155
42	0.0519	-0.0314	0.0568	-0.0323	0.0125	-3.3210	-1.7402	1.0775	-0.2293
43	0.0402	-0.0290	0.0445	-0.0301	0.0112	-3.3890	-1.4958	1.2030	-0.1986
44	0.0300	-0.0261	0.0337	-0.0273	0.0098	-3.4729	-1.1816	1.4147	-0.1303
45	0.0213	-0.0227	0.0244	-0.0241	0.0086	-3.5792	-0.7709	1.5626	-0.0579
46	0.0141	-0.0189	0.0166	-0.0205	0.0075	-3.7153	-0.2287	1.9487	0.1500
47	0.0084	-0.0146	0.0103	-0.0164	0.0066	-3.8905	0.4786	1.9963	0.2912
48	0.0042	-0.0100	0.0055	-0.0119	0.0059	-4.1109	1.3496	2.7365	0.7285
49	0.0014	-0.0051	0.0022	-0.0071	0.0055	-4.3716	2.2843	2.0781	0.8243
50	0.0002	0.0002	0.0003	-0.0020	0.0054	-4.6483	3.0756	3.5774	0.9268
51	0.0008	0.0073	0.0005	0.0035	0.0065	1.3613	3.5559	1.3304	0.9734
52	0.0033	0.0139	0.0014	0.0099	0.0074	1.1223	3.8116	2.5791	0.3721
53	0.0076	0.0206	0.0046	0.0165	0.0085	0.9378	3.8459	1.5279	[illegible]
54	0.0136	0.0275	0.0096	0.0233	0.0097	0.7966	3.7768	1.6955	-0.2310
55	0.0215	0.0345	0.0164	0.0303	0.0110	0.6864	3.6677	1.2814	-0.1730
56	0.0310	0.0415	0.0249	0.0373	0.0124	0.5979	3.5471	1.2345	-0.3850
57	0.0423	0.0484	0.0351	0.0442	0.0137	0.5249	3.4263	1.0430	[illegible]
58	0.0551	0.0552	0.0470	0.0511	0.0151	0.4625	3.3099	0.9685	-0.5317
59	0.0696	0.0618	0.0605	0.0579	0.0164	0.4096	3.1990	0.8616	[illegible]
60	0.0855	0.0682	0.0756	0.0644	0.0177	0.3612	3.0940	0.7964	[illegible]
61	0.1030	0.0742	0.0922	0.0707	0.0190	0.3182	2.9945	0.7275	-0.7781
62	0.1218	0.0799	0.1102	0.0766	0.0202	0.2788	2.9003	0.6756	-0.8181
63	0.1420	0.0852	0.1296	0.0822	0.0214	0.2425	2.8109	0.6261	-0.8451
64	0.1635	0.0901	0.1503	0.0873	0.0225	0.2092	2.7261	0.5841	-0.8721
65	0.1862	0.0944	0.1723	0.0919	0.0235	0.1764	2.6456	[illegible]	-0.8887
66	0.2100	0.0982	0.1955	0.0961	0.0245	0.1461	2.5692	0.5087	-0.8972
67	[illegible]	[illegible]	[illegible]	[illegible]	0.0255	[illegible]	[illegible]	[illegible]	[illegible]

68	Ø.26Ø7	Ø.1Ø4Ø	Ø.245Ø	Ø.1Ø26	Ø.Ø263	Ø.Ø9ØØ	2.4282	Ø.4454	-Ø.8969
69	Ø.2875	Ø.1Ø6Ø	Ø.2712	Ø.1Ø5Ø	Ø.Ø271	Ø.Ø638	2.3634	Ø.419Ø	-Ø.8912
70	Ø.315Ø	Ø.1Ø74	Ø.2983	Ø.1Ø67	Ø.Ø279	Ø.Ø388	2.3Ø24	Ø.3953	-Ø.8822
71	Ø.3433	Ø.1Ø8Ø	Ø.3261	Ø.1Ø78	Ø.Ø285	Ø.Ø148	2.245Ø	Ø.3741	-Ø.8687
72	Ø.3721	Ø.1Ø81	Ø.3547	Ø.1Ø82	Ø.Ø291	-Ø.ØØ81	2.1912	Ø.3526	-Ø.8496
73	Ø.4Ø15	Ø.1Ø74	Ø.3838	Ø.1Ø8Ø	Ø.Ø296	-Ø.Ø3Ø1	2.1411	Ø.33Ø5	-Ø.8247
74	Ø.4314	Ø.1Ø62	Ø.4134	Ø.1Ø71	Ø.Ø3ØØ	-Ø.Ø512	2.Ø945	Ø.3Ø79	-Ø.794Ø
75	Ø.4616	Ø.1Ø42	Ø.4434	Ø.1Ø55	Ø.Ø3Ø4	-Ø.Ø712	2.Ø514	Ø.2859	-Ø.7612
76	Ø.492Ø	Ø.1Ø17	Ø.4737	Ø.1Ø34	Ø.Ø3Ø6	-Ø.Ø9Ø4	2.Ø119	Ø.266Ø	-Ø.7267
77	Ø.5225	Ø.Ø986	Ø.5Ø42	Ø.1ØØ6	Ø.Ø3Ø8	-Ø.1Ø85	1.9758	Ø.2492	-Ø.692Ø
78	Ø.5531	Ø.Ø95Ø	Ø.5348	Ø.Ø973	Ø.Ø3Ø8	-Ø.1257	1.943Ø	Ø.2344	-Ø.6567
79	Ø.5837	Ø.Ø9Ø8	Ø.5654	Ø.Ø934	Ø.Ø3Ø8	-Ø.1418	1.9136	Ø.22Ø2	-Ø.619Ø
80	Ø.614Ø	Ø.Ø862	Ø.5959	Ø.Ø891	Ø.Ø3Ø6	-Ø.157Ø	1.8875	Ø.2Ø54	-Ø.5784
81	Ø.6441	Ø.Ø812	Ø.6261	Ø.Ø843	Ø.Ø3Ø4	-Ø.1711	1.8645	Ø.1899	-Ø.[illegible]
82	Ø.6737	Ø.Ø758	Ø.656Ø	Ø.Ø791	Ø.Ø3ØØ	-Ø.1841	1.8447	Ø.174Ø	-Ø.[illegible]
83	Ø.7Ø29	Ø.Ø7Ø2	Ø.6855	Ø.Ø736	Ø.Ø295	-Ø.196Ø	1.8278	Ø.1583	-Ø.[illegible]
84	Ø.7314	Ø.Ø643	Ø.7144	Ø.Ø679	Ø.Ø289	-Ø.2Ø68	1.8139	Ø.143Ø	-Ø.3952
85	Ø.7591	Ø.Ø583	Ø.7427	Ø.Ø62Ø	Ø.Ø281	-Ø.2163	1.8Ø28	Ø.1284	-Ø.3477
86	Ø.786Ø	Ø.Ø523	Ø.77Ø1	Ø.Ø559	Ø.Ø272	-Ø.2247	1.7944	Ø.1142	-Ø.[illegible]
87	Ø.8119	Ø.Ø463	Ø.7966	Ø.Ø499	Ø.Ø262	-Ø.2318	1.7888	Ø.1ØØ3	-Ø.2531
88	Ø.8367	Ø.Ø4Ø3	Ø.8221	Ø.Ø438	Ø.Ø25Ø	-Ø.2377	1.7859	Ø.Ø867	-Ø.2Ø62
89	Ø.86Ø2	Ø.Ø345	Ø.8464	Ø.Ø38Ø	Ø.Ø237	-Ø.2422	1.7858	Ø.Ø734	-Ø.16Ø1
90	Ø.8824	Ø.Ø29Ø	Ø.8695	Ø.Ø323	Ø.Ø223	-Ø.2454	1.7889	Ø.Ø6Ø5	-Ø.11[illegible]
91	Ø.9Ø31	Ø.Ø238	Ø.8911	Ø.Ø269	Ø.Ø2Ø7	-Ø.2472	1.7955	Ø.Ø483	-Ø.Ø72Ø
92	Ø.9221	Ø.Ø19Ø	Ø.9111	Ø.Ø218	Ø.Ø189	-Ø.2477	1.8Ø6Ø	Ø.Ø366	-Ø.[illegible]
93	Ø.9394	Ø.Ø147	Ø.9294	Ø.Ø172	Ø.Ø17Ø	-Ø.2467	1.8237	Ø.Ø255	Ø.[illegible]
94	Ø.9547	Ø.Ø1Ø8	Ø.946Ø	Ø.Ø13Ø	Ø.Ø15Ø	-Ø.2444	1.8484	Ø.Ø148	Ø.Ø451
95	Ø.968Ø	Ø.ØØ75	Ø.96Ø5	Ø.ØØ94	Ø.Ø129	-Ø.24Ø7	1.8789	Ø.ØØ46	Ø.[illegible]
96	Ø.9792	Ø.ØØ48	Ø.973Ø	Ø.ØØ63	Ø.Ø1Ø6	-Ø.2356	1.9Ø2Ø	-Ø.ØØ53	Ø.11[illegible]
97	Ø.9881	Ø.ØØ27	Ø.9833	Ø.ØØ38	Ø.ØØ82	-Ø.2292	1.8966	-Ø.Ø141	Ø.15[illegible]
98	Ø.9945	Ø.ØØ12	Ø.9912	Ø.ØØ2Ø	Ø.ØØ56	-Ø.2215	1.8691	-Ø.Ø2Ø2	Ø.18[illegible]
99	Ø.9985	Ø.ØØØ4	Ø.9967	Ø.ØØØ7	Ø.ØØ3Ø	-Ø.2127	1.76Ø4	-Ø.Ø282	Ø.23[illegible]
100	Ø.9999	Ø.ØØØ1	Ø.9997	Ø.ØØØ1	Ø.ØØØ3	-Ø.2Ø95	1.7827	-Ø.1Ø51	Ø.2629

JOUKOWSKI AIRFOIL -Ø.5ØØØØØE-Ø1 Ø.5ØØØØØE-Ø1 Ø.5ØØØØØE+ØØ

SUM OF SOURCE DISTRIBUTION = Ø.Ø118
SUM OF VORTEX DISTRIBUTION = Ø.7857
ANGLE OF ATTACK = Ø.ØØØØDEG.
ONSET VORTICITY = Ø.ØØØØ
RATIO OF CNTL PT = Ø.6ØØØ
CL = Ø.729Ø
CD = -Ø.Ø111
ZERO LIFT ANGL = 5.7392
YINFY = -Ø.Ø185
UINFY = 13.1742
U[illegible] = 3.[illegible]
V[illegible] = Ø.[illegible]

VELOCITY AND VORTICITY ON AIRFOIL :

I	U0(I)	V0(I)	U(I)	V(I)
1	13.1775	0.0000	11.1080	-2.1330
2	13.1775	0.0000	11.2071	-2.0277
3	13.1775	0.0000	11.0557	-1.8654
4	13.1774	0.0000	10.9014	-1.6977
5	13.1773	0.0000	10.7811	-1.5316
6	13.1772	0.0000	10.7112	-1.3692
7	13.1770	0.0000	10.6582	-1.2059
8	13.1767	0.0000	10.6135	-1.0411
9	13.1764	0.0000	10.5789	-0.8757
10	13.1760	0.0000	10.5558	-0.7105
11	13.1757	0.0000	10.5448	-0.5462
12	13.1754	0.0000	10.5459	-0.3833
13	13.1752	0.0000	10.5590	-0.2223
14	13.1751	0.0000	10.5840	-0.0640
15	13.1751	0.0000	10.6204	0.0909
16	13.1752	0.0000	10.6679	0.2419
17	13.1754	0.0000	10.7263	0.3882
18	13.1758	0.0000	10.7958	0.5288
19	13.1762	0.0000	10.8771	0.6630
20	13.1766	0.0000	10.9708	0.7899
21	13.1770	0.0000	11.0772	0.9086
22	13.1773	0.0000	11.1958	1.0178
23	13.1775	0.0000	11.3258	1.1165
24	13.1775	0.0000	11.4659	1.2031
25	13.1772	0.0000	11.6156	1.2761
26	13.1767	0.0000	11.7750	1.3339
27	13.1760	0.0000	11.9450	1.3748
28	13.1750	0.0000	12.1262	1.3970
29	13.1738	0.0000	12.3184	1.3983
30	13.1725	0.0000	12.5202	1.3759
31	13.1711	0.0000	12.7295	1.3268
32	13.1697	0.0000	12.9434	1.2475
33	13.1684	0.0000	13.1598	1.1339
34	13.1672	0.0000	13.3758	0.9815
35	13.1662	0.0000	13.5901	0.7848
36	13.1654	0.0000	13.7972	0.5373
37	13.1650	0.0000	13.9943	0.2310
38	13.1649	0.0000	14.1675	-0.1440
39	13.1652	0.0000	14.3136	-0.5996
40	13.1658	0.0000	14.3916	-1.1490
41	13.1667	0.0000	14.4112	-1.8112
42	13.1680	0.0000	14.2536	-2.5934
43	13.1694	0.0000	13.9843	-3.5319
44	13.1709	0.0000	13.2446	-4.5564
45	13.1725	0.0000	12.3316	-5.7688
46	13.1741	0.0000	10.2009	-6.5926
47	13.1754	0.0000	8.1243	-7.5514
48	13.1765	0.0000	3.8843	-5.6595
49	13.1773	0.0000	1.8455	-5.2047
50	13.1775	0.0000	-0.2290	3.5692
51	13.1770	0.0000	0.4472	2.1036
52	13.1756	0.0000	4.5265	9.4069
53	13.1734	0.0000	6.2396	8.5360
54	13.1702	0.0000	9.3771	9.7931
55	13.1660	0.0000	11.0423	9.0462
56	13.1600	0.0000	12.8034	8.[illegible]112
57	13.1540	0.0000	13.[illegible]510	[illegible].[illegible]014
58	13.1481	0.00[illegible]	1[illegible].[illegible]720	7.[illegible]020
59	13.1[illegible]	[illegible]	1[illegible]	5.7[illegible]
60	13.1[illegible]	[illegible]	1[illegible].1[illegible]74	[illegible].1[illegible]

61	13.1243	0.0000	16.6717	5.4906
62	13.1159	0.0000	17.0778	4.8879
63	13.1074	0.0000	17.3911	4.2988
64	13.0993	0.0000	17.6417	3.7293
65	13.0916	0.0000	17.8244	3.1764
66	13.0846	0.0000	17.9526	2.6420
67	13.0785	0.0000	18.0309	2.1264
68	13.0733	0.0000	18.0713	1.6309
69	13.0693	0.0000	18.0804	1.1559
70	13.0666	0.0000	18.0604	0.7012
71	13.0652	0.0000	18.0070	0.2669
72	13.0651	0.0000	17.9163	-0.1460
73	13.0664	0.0000	17.7377	-0.5362
74	13.0691	0.0000	17.6256	-0.9026
75	13.0729	0.0000	17.4393	-1.2444
76	13.0779	0.0000	17.2406	-1.5620
77	13.0838	0.0000	17.0360	-1.8557
78	13.0905	0.0000	16.8229	-2.1254
79	13.0980	0.0000	16.5946	-2.3697
80	13.1059	0.0000	16.3477	-2.5877
81	13.1139	0.0000	16.0833	-2.7788
82	13.1220	0.0000	15.8057	-2.9431
83	13.1300	0.0000	15.5198	-3.0815
84	13.1375	0.0000	15.2297	-3.1945
85	13.1446	0.0000	14.9377	-3.2829
86	13.1511	0.0000	14.6451	-3.3472
87	13.1568	0.0000	14.3527	-3.3881
88	13.1618	0.0000	14.0622	-3.4063
89	13.1660	0.0000	13.7758	-3.4030
90	13.1694	0.0000	13.4960	-3.3797
91	13.1720	0.0000	13.2253	-3.3377
92	13.1740	0.0000	12.9650	-3.2786
93	13.1754	0.0000	12.7207	-3.2041
94	13.1764	0.0000	12.4910	-3.1154
95	13.1770	0.0000	12.2712	-3.0122
96	13.1773	0.0000	12.0377	-2.8900
97	13.1774	0.0000	11.7920	-2.7525
98	13.1775	0.0000	11.5108	-2.6145
99	13.1775	0.0000	11.2191	-2.4296
100	13.1775	0.0000	11.0836	-2.3524

VITA

Jyh-Chyang Alex Wang was born in Kaohsiung, Taiwan, Republic of China, on December 12, 1958. He graduated from Chen Sheng Catholic Junior High School in 1974 and graduated from Taipei Municipal Chien Kuo Senior High School in 1977. The following October he enrolled in the National Cheng Kung University with a major in Aeronautical Engineering. In 1981 he received the Bachelor of Science degree in Aeronautical Engineering.

After the fulfillment of compulsory military service, he married Min-Huei Luo in Taiwan in June 1983. About a month later, they came to the United States together and he received a M.S. degree in Aerospace Engineering from the University of Maryland in 1984. In October 1984, he started to work on his Ph.D. degree at The University of Tennessee Space Institute. He was awarded the degree of Doctor of Philosophy with a major in Aerospace Engineering in December, 1988.

Mr. Wang is a member of The Honor Society of Phi Kappa Phi, American Institute of Aeronautics and Astronautics, The Society for Industrial and Applied Mathematics, and an associate member of Sigma Xi. He will be employed by McDonnell Douglas Astronautics Company in Huntsville, Alabama, after graduation.