Titre: Title:	Improvement of a tridimensional transonic panel method
Auteur: Author:	Richard Clairoux
Date:	1991
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Clairoux, R. (1991). Improvement of a tridimensional transonic panel method [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/57025/

UNIVERSITÉ DE MONTRÉAL

Improvement of a Tridimensional Transonic Panel Method

par

Richard CLAIROUX

DÉPARTEMENT DE GÉNIE MÉCANIQUE

ÉCOLE POLYTECHNIQUE

RAPPORT DE PROJET PRÉSENTÉ EN VUE DE L'OBTENTION

DU GRADE DE MAÎTRE EN INGÉNIERIE (M.Ing.)

GÉNIE AÉRONAUTIQUE

DÉCEMBRE 1991

SOMMAIRE

L'étude des écoulements transsoniques autour d'ailes tridimensionnelles est un domaine d'étude important des grandes firmes en aéronautique. Pendant plusieurs années, cette étude a été limitée à l'emploi de méthodes de différences finies ou d'éléments finis. Les résultats obtenus par ces méthodes sont satisfaisants mais la construction des maillages dans le cas de configurations complexes demeure une tâche compliquée et longue à réaliser.

La méthode de panneaux tridimensionnelle transsonique s'avère une alternative économique à ces approches. La méthode de panneaux bien connue en écoulement incompressible est modifiée pour modéliser la non-linéarité des écoulements transsoniques. Une nouvelle grille de champs est créée autour de l'aile et est divisée en un nombre limité de volumes. On place au centre de chacun de ces volumes une source ponctuelle de force inconnue. La valeur de la force de ces sources change avec le type d'écoulement pour représenter les variations de compressibilité autour de l'aile. Ces sources sont inclues dans une nouvelle intégrale de volume. L'équation de base devient non-linéaire puisqu'on retrouve plusieurs inconnues et doit donc être résolue de façon itérative. Comme il n'y a pas de conditions frontières à respecter, la grille de champs peut pénétrer le volume intérieur de l'aile sans se conformer à la surface. L'étude d'écoulements autour de configurations à plusieurs composantes peut donc être réalisée plus facilement et plus rapidement que dans le cas des méthodes mentionnées précédemment.

Ce rapport présente l'ensemble des tâches effectuées pour améliorer un programme de panneaux transsonique tridimensionnel. La première partie décrit la théorie employée par la méthode. On montre également comment on peut discrétiser numériquement les intégrales obtenues. Nous présentons ensuite une

série de modifications qui peuvent diminuer le temps de calcul du programme aux dépends d'une baisse de précision des résultats acceptable. Par la suite, le programme lui-même est décrit. L'usager peut apprendre aisément comment le programme est utilisé. Nous décrivons en détail l'algorithme itératif qui résout la non-linéarité de l'écoulement.

La deuxième partie du rapport énumère l'ensemble des cas qui ont été exécutés pour valider les modifications et pour vérifier la précision du programme. Nous avons découvert que les avantages de l'écriture en simple précision ne peuvent être utilisés puisque l'écriture en double précision est requise dans certains calculs. Toutefois, une nouvelle méthode de calcul simplifiée dans l'évaluation du potentiel des vitesses des noeuds éloignés de la grille de champs a un impact important sur le temps de calcul. Celui-ci est diminué de près de 50% pour une baisse de précision négligeable.

Les résultats du programme sont comparés avec ceux d'un programme Euler pour un cas particulier. Il y a un bon accord en général entre les deux courbes de distribution de pression à diverses sections d'aile. Toutefois, nos essais montrent qu'il apparaît des discontinuités de vitesse au bord de fuite lorsque l'incidence de l'aile est différente de zéro. Ceci demeure donc un problème à régler avant que ce programme ne soit complètement validé.

Donc, ce rapport permet au lecteur de bien comprendre le fonctionnement d'une méthode de panneaux tridimensionnelle transsonique. Ces méthodes possèdent un avenir intéressant étant donné que la simulation numérique des écoulements en aéronautique devient de plus en plus populaire aux dépends des coûteux essais en soufflerie.

ABSTRACT

The study of transonic flows over three-dimensional (3-D) wings is a major field of interest among aircraft manufacturers. For years, the analysis was limited to finite-difference or finite-element methods. While these formulations give acceptable results, the generation of suitable grids around complex configurations remains a major task.

An economic alternative is the transonic 3-D integral method. The well-known incompressible panel method is modified to include the non-linearity of transonic flows. A new field grid is created around the wing and is divided in a limited number of volumes with a point source at their center. These sources represent the compressibility variations around the wing. With a new volume integral, the basic equation becomes non-linear and is solved iteratively. The flow around a multi-body configuration can be studied more easily than the previous methods because there are no boundary requirements on the field grid. The grid doesn't need to fit the wing surface and is thus easier to generate.

The report presents the work done to improve a transonic 3-D integral code. The first part describes the integral theory in great detail and how this is numerically discretized in the program. A list of modifications is then given in order to lower the computing time at the expense of an acceptable loss of accuracy. The program itself is presented: we show how to use the program and how the iterative algorithm is accounted for.

The second part of the report describes the validation of the modifications and the verification of the program's accuracy. We found out that the advantages of single precision accuracy cannot be used since some calculations require the double precision accuracy. However, a simpler method for the evaluation of the

velocity potential of far field nodes makes a remarkable impact on computing time. In some cases, the CPU time is cut by almost 50% with a negligible loss of accuracy.

The program's results are compared with those from an Euler code for a specific transonic case. There is good agreement between the two pressure distribution curves for different wing stations. However, this can be predicted only at zero degrees incidence since the program's testing showed that there are velocity discontinuities at the wing trailing edge for incidences different than zero. Thus, the program needs some modifications to correct this problem.

Thus, this report explains accurately the operation of a transonic 3-D panel method. The text can be used by the reader to understand how to use the program and to learn what kind of results should be expected. The 3-D integral method should be very useful in the near future because the aircraft manufacturers are depending more and more on computational fluids dynamics (CFD) studies with the rising costs of wind tunnel testing.

RÉSUMÉ

Le développement fulgurant des ordinateurs à grande vitesse ainsi que l'amélioration des méthodes numériques ont grandement contribué à l'étude des écoulements transsoniques autour d'ailes tridimensionnelles. Pendant plusieurs années, la non-linéarité des équations de base en régime transsonique a retardé l'écriture de programmes efficaces et précis. De nos jours, les grandes avionneries possèdent des programmes qui sont basés sur les méthodes de différences finies, d'éléments finis ou de volumes finis. L'étude par simulation numérique des écoulements transsoniques devient de plus en plus intéressante car les coûts d'opération des souffleries aérodynamiques à haute vitesse sont devenus exorbitants. Avec l'usage des programmes numériques, les aérodynamiciens peuvent modifier les caractéristiques des ailes et en voir les résultats sur les performances transsoniques peu de temps après sans qu'il ne soit nécessaire de construire un modèle et d'employer une soufflerie.

Toutefois, les méthodes précédentes comportent certaines lacunes. Ainsi, la création des maillages tridimensionnels (3-D) qui s'adaptent au corps est une tâche longue et ardue. C'est pour cette raison que l'étude de l'écoulement autour de profils multi-corps (aile avec volets au bord de fuite) ou de l'interaction entre l'aile et les nacelles moteur est très difficile à réaliser. De plus, des centaines d'itérations sont souvent nécessaires pour avoir une représentation valable de l'écoulement. Ceci mène à des coûts d'opération de super-ordinateur parfois exorbitants.

La méthode de panneaux 3-D pour écoulements transsoniques s'avère donc une nouvelle approche fort intéressante. Pendant plusieurs années, la méthode de panneaux a été limitée au cas particulier des écoulements incompressibles. Les méthodes de panneaux sont basées sur une équation du potentiel de la vitesse

qui provient de la théorie de la fonction de Green. Or, cette fonction s'applique exclusivement pour des écoulements linéaires (équation de Laplace) et donc, pour des écoulements incompressibles uniquement.

Récemment, une version transsonique de la méthode de panneaux 3-D a été développée par Sinclair de British Aerospace (Réf.1 et 2). On ajoute à l'équation de base une intégrale de volume. Celle-ci intègre une distribution de sources ponctuelles de force σ_i placées au centre de petits volumes (appelés éléments) qui entourent l'aile. Cette intégrale transforme l'équation de base en une équation non-linéaire. En fait, la compressibilité de l'écoulement autour de l'aile est tenue pour compte avec la nouvelle intégrale. Le système est résolu de manière itérative puisque les valeurs des sources de champs σ_i sont initialement inconnues.

Les conditions frontières sur la grille de champs qui forme le volume ne sont pas aussi sévères que pour les méthodes mentionnées précédemment. Ainsi, cette grille peut pénétrer l'espace intérieur de l'aile: certains éléments de champs peuvent être inclus dans l'aile. Donc, il n'est pas nécessaire de créer une grille qui s'adapte au contour de la surface. De plus, un nombre limité d'éléments est requis autour de l'aile. Ceci est dû au fait que lorsqu'on s'éloigne de l'aile, l'écoulement redevient non-perturbé et donc la valeur des sources σ_i y est nulle. Ceci permet donc d'analyser beaucoup plus facilement l'écoulement autour d'aile multi-corps ou encore de combinaisons aile-fuselage. Aussi, un nombre restreint d'itérations (de l'ordre de 10) est nécessaire pour que la valeur des sources converge.

Donc, la méthode de panneaux pour écoulements transsoniques peut s'avérer très utile pour l'industrie aéronautique. Depuis quelques années, plusieurs projets concernant ce sujet ont été réalisés par le département de Génie Mécanique de l'École Polytechnique de Montréal. Après les ouvrages de Masson (Réf.3, bi-dimensionnel transsonique) et de Normandin (Réf.4, 3-D incompressible), Fang

(Réf.5) a réalisé en 1990 un programme transsonique 3-D employant la méthode de panneaux. Ce programme a été réalisé dans le cadre d'une entente de recherche avec la firme Bombardier-Canadair. Toutefois, ce programme nécessite plusieurs modifications avant de pouvoir être employé sur des cas pratiques. Ainsi, le temps de calcul semble très grand et le programme n'a pas été validé sur un cas transsonique typique. On note également qu'il n'existe pas une version commentée du programme et que certains détails sur son fonctionnement demeurent obscurs.

Donc, ce rapport présente les différentes étapes qui ont été réalisées pour améliorer ce programme de panneaux 3-D transsonique. Le rapport débute par l'énumération au Chapitre 1 des trois buts principaux du projet:

1-Réaliser une étude complète du programme pour bien comprendre son fonctionnement et écrire une version bien documentée;
2-Déterminer des modifications qui permettront d'améliorer le temps de calcul du programme tout en conservant une variation de précision acceptable;
3-Vérifier la précision du programme en comparant ses résultats avec des valeurs expérimentales dans le cas bien connu de l'aile ONERA M6 à Mach M_∞=0.84 et à un angle d'attaque α=3.06°.

On démontre ensuite au Chapitre 2 la théorie de base de la méthode. Nous mentionnons comment la théorie de la fonction de Green nous permet d'obtenir la forme de base en écoulement incompressible. Comme l'écoulement est irrotationnel, on peut dire que la vitesse dérive d'un potentiel Φ. Donc, l'inconnue à résoudre est la distribution de ce potentiel sur l'aile. Lorsque cette distribution est connue, on peut déterminer les valeurs de la vitesse et du coefficient de pression (Cp) sur l'aile. On ajoute à l'équation deux nouvelles intégrales qui tiennent compte des sources de champs et des surfaces de discontinuité causées

par les ondes de choc (s'il y a lieu). Nous montrons comment cette forme de base peut être simplifiée à l'équation suivante:

$$\Phi(x_p,y_p,z_p) = \Phi_\infty - \int_{Sa} \Phi(\bar{n}\cdot\nabla\Phi_L)ds - \int_{Sb} \Phi(\bar{n}\cdot\nabla\Phi_L)ds + \int_{Sc} \Phi_L(\bar{n}\cdot\nabla\Phi)ds + \int_V \sigma\,\Phi_L\,dv$$

avec

$$\Phi_L = -1/(4\pi)\cdot[(x-x_p)^2 + (y-y_p)^2 + (z-z_p)^2]^{-1/2}$$

où (x_p,y_p,z_p) sont les coordonnées du point évalué, Φ_∞ est le potentiel de l'écoulement non perturbé, ∇ est l'opérateur du gradient, $\bar{n}$ est le vecteur normal unitaire et σ est la valeur inconnue de la force des sources ponctuelles de champs. Les intégrales sont réalisées sur les surfaces de discontinuité, soit S_a: l'aile, S_b: le sillage de l'aile et S_c: les surfaces de choc. V représente le volume mentionné précédemment. Donc, les intégrales sur S_c et V sont les nouveaux termes pour la formulation transsonique de la méthode des panneaux. On voit bien que l'équation est non-linéaire puisque Φ et σ à la fois sont inconnus.

Le Chapitre 3 montre comment ces intégrales sont discrétisées sur les différentes surfaces et le volume. Ainsi, la surface de l'aile est divisée en triangles qui joignent les noeuds. La position de ces noeuds est fournie par l'usager lors de la spécification de la géométrie de l'aile (les points autour du profil de quelques sections d'aile). On suppose une variation linéaire du potentiel de vitesse Φ sur ces panneaux triangulaires. Le sillage est représenté par des bandes qui quittent le bord de fuite de l'aile vers l'infini. Ces bandes sont situées entre les sections d'aile fournies par l'usager. Le volume est représenté par des éléments parallélépipèdes qui entourent l'aile. Ces éléments deviennent plus volumineux en s'éloignant de l'aile puisque la compressibilité devient moins importante.

On détermine les surfaces de chocs de la manière suivante: lorsque le

nombre de Mach d'un élément de champs est supérieur à un et que celui de l'élément suivant est sous l'unité, alors la surface de contact des deux éléments est considérée comme un choc. Les intégrales sont représentées par des sommations. Les intégrales sur S_a et S_b deviennent des fonctions de Φ seulement: on peut donc les déplacer du côté gauche de l'équation et on obtient le système suivant à résoudre lorsqu'on répète l'équation de base aux n noeuds de la surface de l'aile:

$$[A(n,n)]\ [\Phi(n)] = [\Phi_\infty + \Phi_{VOL} + \Phi_{CHOC}] = [RHS(n)]$$

où les n inconnues Φ_i sont les valeurs du potentiel de vitesse sur l'aile, Φ_{VOL} représente l'intégrale de volume et Φ_{CHOC} celle sur les surfaces de choc S_c. Une méthode itérative est employée pour résoudre le problème. On commence par solutionner le système lorsque M_∞=0.0: les intégrales Φ_{VOL} et Φ_{CHOC} s'annulent et on retrouve le système linéaire pour les écoulements incompressibles. On obtient les valeurs de Φ aux noeuds de l'aile. Ces valeurs sont ensuite utilisées pour calculer les valeurs de Φ des noeuds de champs, permettant de déterminer les surfaces de choc et les valeurs des sources σ_i de champs. On peut alors calculer les intégrales Φ_{VOL} et Φ_{CHOC} et obtenir un nouveau membre de droite [RHS(n)]. Le système est encore résolu pour calculer les nouveaux Φ aux noeuds de l'aile, menant à de nouveaux σ_i et surfaces de chocs. Ce cycle est répété jusqu'à ce que la différence entre les valeurs de deux itérations devienne négligeable. On remarque que la matrice [A] demeure inchangée au cours des itérations.

Par la suite, nous présentons les modifications proposées pour réduire le temps de calcul (Chapitre 4). Entre-autres, le passage de l'écriture en double précision à celle en simple précision est suggéré. On propose aussi une méthode simplifiée pour le calcul des potentiels Φ des noeuds de champs qui sont éloignés de l'aile. Cette méthode fait appel à une version simplifiée de l'intégrale sur S_a. De

plus, nous suggérons de retirer des calculs de potentiel certains noeuds de champs situés au plan inférieur de l'aile puisque les valeurs de Φ y sont identiques à celles des noeuds équivalents au plan supérieur. Les autres modifications sont l'emploi de nouveaux vecteurs (pour économiser des calculs), une nouvelle boucle de calcul du coefficient de portance de l'aile et l'annulation de certains calculs inutiles.

On passe ensuite au chapitre 5 à une présentation du programme. Le fichier d'entrée est d'abord décrit. Ceci est suivi par l'analyse du programme principal et des 16 sous-routines. L'usager peut ainsi comprendre rapidement les fonctions et les méthodes employées par les diverses composantes du programme. Les paramètres d'entrée à changer et le type d'information présenté à la sortie sont clairement définis. Le listing du programme est d'ailleurs disponible en annexe.

Nous présentons ensuite au Chapitre 6 les résultats des validations de nos modifications ainsi que de la comparaison pour le cas de l'aile ONERA M6 en écoulement transsonique. Nous avons découvert que le changement à l'écriture en simple précision n'entraîne pas une réduction du temps de calcul puisque les résidus à chaque itération sont plus élevés. Les autres modifications fonctionnent bien toutefois. Ceci est particulièrement le cas pour la version simplifiée de calcul du potentiel pour les noeuds de champs éloignés: une réduction du temps de calcul de 40% est obtenue pour une perte de précision acceptable.

Toutefois, les essais de validation pour le calcul du coefficient de portance nous montrent que pour des angles d'attaques α différents de zéro, des discontinuités importantes apparaissent dans les valeurs des coefficients de pression au bord de fuite. Ceci ce produit également pour des écoulements incompressibles et ne peut donc pas être lié au calcul des σ_i ou des surfaces de choc. L'apparition de ce problème nous oblige à réaliser une comparaison des

résultats pour l'aile ONERA M6 pour α=0.0° et M_∞=0.84. La comparaison avec les résultats d'un programme Euler employant la méthode des volumes finis montre une grande similitude entre les résultats. Les courbes de distribution de pression ont un comportement semblable sauf lors des pics où un léger décalage est présent.

Ce rapport permet donc à l'utilisateur du programme d'avoir une bonne compréhension du logiciel. Les modifications apportées réduisent efficacement le temps de calcul. Toutefois, il existe un problème à résoudre pour le cas des écoulements dont l'incidence est différente de zéro.

ACKNOWLEDGEMENTS

First, I would like to thank professor Ion Paraschivoiu for giving me the opportunity to work on an interesting aerodynamics project. Also, I sincerly thank professor B.C. Basu for guiding me all along this work.

I have to mention the much needed assistance of Dr. Zhigang Fang: he helped in explaining obscure sections of his program. Also, I would like to thank fellow M.Ing. student Stéphane Major for bringing me the Euler transonic code results used in the comparison of Chapter 6. Finally, I would like to mention the help of Adrian Hinca at the beginning of the project and the aid of my father during the writing of the report.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The recent development of very powerful computing facilities and the rapid improvement in numerical methods have resulted in many new transonic flow programs in the aircraft design community. For many years, the non-linearity of the governing flow equations in the transonic regime had prevented the development of reliable codes. The calculation methods are based on the finite-difference, finite-element and finite-volume approaches. This field of study is very important for the commercial aircraft companies since jet transport aircrafts and business jets cruise more economically at high altitude and at speeds in the Mach 0.7-0.85 range. These cruising Mach numbers lead to supersonic regions on the aircraft's wing, hence the transonic flow regime.

The main objective of the aerodynamics studies group is to design a wing that will maximise the drag rise Mach number for good range performance. Since wind tunnel testing is more and more expensive, the use of accurate transonic flow codes becomes a welcome alternative. However, in both finite-difference and finite-element approaches, the generation of suitable grids for flow around complex configurations is a major task.

The integral method based on the Green's function theory has been used to solve aerodynamic problems for several decades. However, this method was always restricted to incompressible flows because of the Green's function limitation to linearized flows. Recently, a compressible version has been developed by Sinclair of British Aerospace (Ref.1 and 2). A volume integral is added to the basic incompressible formulation to solve the non-linearity of the compressible flow. The computational grids consists of boundary grids and a field grid. The boundary grids are used for the implementation of the boundary integrals while the field grids are used for the evaluation of the volume integral. The field grid is divided in a limited number of volumes with a point source at their center. The strength of theses

sources is initially unknown and thus, the non-linear equation is solved by using an iterative procedure.

Since the field grid does not need to fit the aircraft boundary surface, the integral method can be easily applied to solve flow problems over complex multi-body configurations such as wings with flaps and slats deployed or wing-mounted engine nacelles. Also, tests have shown that only a small number of iterations (ten or so) are required to solve the numerical system with reasonable accuracy. Thus, the transonic integral formulation presents some very interesting features that could be very useful for aircraft designers.

The Sinclair approach has been the subject of several graduate students projects at the École Polytechnique's Department of Mechanical Engineering. In 1989, Masson (Ref.3) developed a two-dimensional (2-D) transonic flow program. This was followed by the work of Normandin in 1990 (Ref.4) who wrote a three-dimensional (3-D) incompressible flow program studying the wing and wake interaction. Finally, Fang in 1990 (Ref.5) developed a 3-D integral method for transonic flows.

This M.Ing project involves the improvement of Fang's 3-D program. Some modifications and testing are required before the program can be used efficiently on practical problems. Also, a complete analysis of the program is needed to describe accurately the operation of the code. This report presents the different steps that were made in order to realize these goals. The first Chapter talks about the purpose of the report. This is followed in Chapter 2 by the program theory and in Chapter 3 by the numerical discretization. Then, Chapter 4 lists the program modifications that were made and Chapter 5 describes the program itself. Finally, Chapter 6 presents the important results of the validation process. A conclusion ends the report.

CHAPTER 1
PURPOSE OF REPORT

1.1 PROGRAM HISTORY

The purpose of this project was the improvement of a three-dimensional (3-D) transonic panel code. As part of a research convention between Bombardier-Canadair and the École Polytechnique, such a program was finished in the summer of 1990 by a research associate, Dr. Zhigang Fang.

The 3-D program was an extension of a two-dimensional (2-D) code on which work started in May, 1989. This program was used to study transonic flows over a NACA 0012 section with rectangular and body-fitted grids. In the period of March to May, 1990, a 3-D incompressible flow program was developed from the 2-D code and was successfully tested on the analytical case of a sphere. Finally, the 3-D transonic version was made in the period of May to August, 1990. The code was run on a very coarse grid over an isolated swept wing of constant chord and NACA 0012 section. A report (Ref.5) describing these developments was submitted to Bombardier-Canadair in October, 1990.

This report was studied by Canadair and some comments were addressed about the operation of the code and the accuracy of the results. Actually, the report did not detail how the program worked and the listing did not include comprehensive comments. Also, the results were not compared with any known results. Finally, there were some concerns about the program memory requirements and computing time.

1.2 PROJECT REQUIREMENTS

Three main requirements are to be studied in this project. First, there has to be a complete analysis of the code from which a fully commented version will be made. Then, this study will help in finding modifications that could speed-up computing time and reduce memory needs at the expense of an acceptable loss of accuracy. Finally, the new version will be tested on a well-known transonic case, the ONERA M6 wing at Mach 0.84 at the angle of incidence of 3.06°, and its results compared with experimental values. These three steps will be detailed in this report to facilitate future use of the program by École Polytechnique students or Canadair users.

CHAPTER 2
PROGRAM THEORY

2.1 INTEGRAL FORMULATION

This chapter presents the basic theory that was used in the development of the program. For compressible flows, full potential theory can be written in the following form:

$$\nabla^2\Phi=\sigma \tag{1}$$

where ∇^2 is the Laplacian operator, Φ is the potential of velocity and σ is the source term that expresses the compressibility of the flow. If σ=0.0, we get the Laplacian equation for incompressible flows. σ is defined as:

$$\sigma=M^2\Phi_{ss} \tag{2}$$

where M is the flow Mach number and Φ_{ss} the rate of velocity change in the flow direction:

$$\Phi_{ss} = (u^2\Phi_{xx} + v^2\Phi_{yy} + w^2\Phi_{zz} + 2uv\Phi_{xy} + 2vw\Phi_{yz} + 2uw\Phi_{xz})/|\overline{V}|^2 \tag{3}$$

where u,v and w are the x, y and z components of velocity, $\overline{V}$ is the velocity vector and $\Phi_{xx},\Phi_{xy},...$ are the second derivatives of the velocity potential.

The integral formulation is based on the Green's function theorem. This approach specifies that the potential at a point (x_p,y_p,z_p) in a Laplacian field of flow can be represented by a sum of boundary integrals:

$$\Phi(x_p,y_p,z_p) = \int_S [(\overline{n}\cdot\nabla\Phi)\Phi_L - \Phi(\overline{n}\cdot\nabla\Phi_L)]ds \tag{4}$$

where $\bar{n}$ is the unit normal vector at the boundary surface S and Φ_L represents the potential of an unit strenght source defined in 3-D problems by:

$$\Phi_L = -1/4\pi\cdot[(x-x_p)^2 + (y-y_p)^2 + (z-z_p)^2]^{-1/2} \quad (5)$$

or, using $\bar{r}$ as the vector between (x,y,z) and (x_p,y_p,z_p):

$$\Phi_L = -1/(4\pi|\bar{r}|) \quad (6)$$

The expression (4) is only valid for incompressible flows. The compressibility effect is included by adding a volume integral that uses σ strength sources in the field volume:

$$\Phi(x_p,y_p,z_p) = \int_S [(\bar{n}\cdot\nabla\Phi)\Phi_L - \Phi(\bar{n}\cdot\nabla\Phi_L)]ds + \int_V \sigma\,\Phi_L\,dv \quad (7)$$

Equation (7) shows that the solution Φ can be represented in terms of unknown distribution of sources in the volume V and of sources and doublets on the boundary surface S (Ref.6). It is clear from the equation (7) that an iterative method is required for a solution.

2.2 SURFACE INTEGRALS AND SIMPLIFICATIONS

The surface integral is performed on the boundaries of a simply connected region that surrounds the wing. The volume integral is done on the inside of that same region. The total surface S includes the surfaces S_a, S_b, S_c and S_∞ (Fig.1). S_a represents body surfaces such as wing main section, flaps or slats for an isolated wing. S_b includes the cuts between bodies and the wing wake. The shock surfaces (if any) on the wing are included in S_c and the infinity is represented by S_∞. Because of the discontinuity of potential, the wake surface is required to have

a simply connected region. The shock surfaces also need to be included for the reason mentioned above. Normal unit vectors convention is shown on the figure.

On body surfaces, we have:

$$\int_{Sa} [(\bar{n}\cdot\nabla\Phi)\Phi_L]ds = 0.0 \qquad (8)$$

because of the tangential speed ($\bar{n}\cdot\nabla\Phi$=0.0) condition over body surfaces. For wake surfaces, we also have:

$$\int_{Sb} [(\bar{n}\cdot\nabla\Phi)\Phi_L]ds = 0.0 \qquad (9)$$

This happens because the normal component of $\nabla\Phi$ is always continuous. Since the unit normals are equal but opposite on the two sides of S_b, the contributions of any element "ds" on one side is cancelled by the corresponding element on the opposite side of S_b, thus cancelling the integral (Ref.6).

We also have the following result for shock surfaces:

$$\int_{Sc} [(\bar{n}\cdot\nabla\Phi_L)\Phi]ds = 0.0 \qquad (10)$$

In an oblique shock, the tangential speeds remain identical on both sides. The normal speed will be reduced, causing the overall Mach number reduction. Thus, tangential variation of potential will be the same on both sides of the shock even if potential values are different (Fig.2). Because of the path of integration along the surface, the variation of potentials will be used. The $\bar{n}\cdot\nabla\Phi_L$ terms will only change signs for points across the shock (normal vector direction) since the shock thickness is infinitesimal. This results in the fact that the integral of (10) over one side of S_c will be the exact opposite of the one on the other side, thus cancelling

the total integral.

Finally, it can be easily shown that for points far from the wing, the potential can be expressed as:

$$\Phi(x_p,y_p,z_p)= V_\infty(x_p\cdot\cos\alpha + y_p\cdot\cos\beta + z_p\cdot\cos\gamma) \qquad (11)$$

where V_∞ is the free flow speed normalized to 1.0 and α, β and γ are the flow angles with the x, y and z axes (Fig.3).

We use this result to simplify the S_∞ integral: by inspection, we see that for (x_p,y_p,z_p) at the infinity:

$$\int_{S\infty} [(\bar{n}\cdot\nabla\Phi)\Phi_L - \Phi(\bar{n}\cdot\nabla\Phi_L)]ds = (x_p\cdot\cos\alpha+y_p\cdot\cos\beta+z_p\cdot\cos\gamma) \equiv \Phi_\infty \qquad (12)$$

since all other integrals cancel at the infinity. With equations (8), (9), (10) and (12), equation (7) becomes:

$$\Phi(x_p,y_p,z_p) = \Phi_\infty - \int_{Sa} \Phi(\bar{n}\cdot\nabla\Phi_L)ds - \int_{Sb} \Phi(\bar{n}\cdot\nabla\Phi_L)ds + \int_{Sc} \Phi_L(\bar{n}\cdot\nabla\Phi)ds + \int_V \sigma\,\Phi_L\,dv \qquad (13)$$

This equation is used in the program to solve the flow potential Φ on the wing.

CHAPTER 3
NUMERICAL DISCRETIZATION

3.1 WING SURFACE

The wing surface is divided into a series of triangular panels (Fig.4). This type of panel represents the actual contours of the wing better than the square ones. The panels are fitted to the wing points inputted by the user. A wing tip grid is created to include the influence of this surface. Since only symmetrical flows are usually evaluated (ß=90.0°), the flow will be identical on both sides of the wing and thus only one half will actually be used.

In this program, linear variation of potential on the panels is assumed (Fig.5). Thus, the variation of potential over a panel can be represented by another triangle surface. A local system of coordinates (Fig.6) is necessary to simplify calculations in the S_a and S_b integrals of equation (13). The origin is set at the node #1. The axis x_R is located along the line joining nodes #1 & #2 while axis y_R is in the same plane as the panel. Thus, the axis z_R is perpendicular to the panel surface.

The linear variation of potential is given by:

$$\Phi(x_R,y_R) = \Phi_1 + (\Phi_2-\Phi_1)/x_2 \cdot x_R + [\Phi_3 \cdot x_2 - \Phi_2 \cdot x_3 + \Phi_1(x_3-x_2)]/(x_2 \cdot y_3) \cdot y_R \qquad (14)$$

where 1,2,3 are subscripts for the three panel nodes and (x_R,y_R) are the relative coordinates of the evaluated point located inside the triangle.

3.2 WING WAKE

The wing wake is represented by infinite strips starting at the wing trailing

edge [T.E.] (Fig.4). In fact, there are in theory two wake surfaces separated by an infinitesimal distance. The surfaces join at node #1 (at the corner of the wing's T.E. and tip) to close the wake discontinuity. A local system of coordinates is also used (Fig.7). The line joining the T.E. points is used as a reference. The origin is at node #1, the axis x_R is in the same direction as general axis X, the axis y_R is rotated to be in the same plane as x_R while z_R is perpendicular to both axes. The figure also shows the numbering convention of the four nodes.

3.3 FIELD ELEMENTS

The volume surrounding the wing has to be divided into small parallelepipeds elements to solve the volume integral (Fig.8). There are "NO" such elements in front, behind, over, under and to the left of the wing. Inside each element, the value of the source strength σ (Eq.13) is assumed constant. At a certain distance from the wing, nonlinear effects become negligible and thus values of σ become very small. Thus, only a limited number of elements "NO" is required in each directions around the wing. These combine to the elements on the wing giving the field dimensions MX by MY by MZ. Also, since the nonlinear effects are more important near the wing, the size of the elements becomes larger when moving further away from the wing. In order to avoid special numerical cases that would add up computing time, the field elements are body-fitted to the wing surface. The volume integral is discretized as (Fig.9):

$$\int_V \sigma \, \Phi_L \, dv \approx -1/4\pi \, \Sigma[\sigma_i \cdot VOL_i / |\bar{r}_{Gi}|] \equiv \Phi_{VOL} \qquad (15)$$

with $|\bar{r}_{Gi}| = \sqrt{[(x_{Gi}-x_p)^2+(y_{Gi}-y_p)^2+(z_{Gi}-z_p)^2]}$

and where (x_{Gi},y_{Gi},z_{Gi}) are the general coordinates of the center, VOL_i the volume and σ_i the source strength of the ith element of the summation. The calculation of

the vector position $\bar{r}$ in the integral is thus simplified to the radius $\bar{r}_G$ to the center of the element. This makes sense because distances of a field point to each point of the element generally do not vary much.

3.4 SHOCK SURFACES

Shock surfaces are found by the following way: the average Mach number of each field element is evaluated at their centers. When the Mach number of an element is over or equal to 1.0 and that the following element (along the X axis) Mach number is under 1.0, there is a shock (Fig.10). The meeting side of the two elements is the shock surface. The number and location of shock surfaces depend on the free stream Mach number M_∞, the angles of attack, the wing geometry, etc. Usually, these surfaces will appear at M_∞ values of about 0.80-0.85 and over. The shock integral is discretized as:

$$\int_{Sc} \Phi_L(\bar{n}\cdot\nabla\Phi)ds \approx 1/4\pi\ \Sigma[\Phi_X(IE)-\Phi_X(IE+MY-1)]\cdot SS_i\ /|\bar{r}_s| \equiv \Phi_{SHOCK} \qquad (16)$$

with $|\bar{r}_s| = \sqrt{[(x_{Ci}-x_p)^2+(y_{Ci}-y_p)^2+(z_{Ci}-z_p)^2]}$

and where $[\Phi_X(IE)-\Phi_X(IE+MY-1)]$ is the difference of the x derivatives of the velocity potentials of the elements on both sides of the shock, SS_i is the shock surface area and (x_{Ci},y_{Ci},z_{Ci}) are the general coordinates of the center of the ith shock surface. Basically, this simplification uses the same assumption as with the volume integral concerning far distances radii. In the program, the summation is divided between the "upper" shocks (elements over the wing) and the "lower" shocks (elements under the wing).

3.5 SYSTEM OF EQUATIONS

When the equation (14) is introduced in equation (13), the integrals of the wing surface and wake (S_a and S_b) will be discretized in functions of the wing potentials Φ_i at the nodes. These discretizations are exact summations and due to their enormous size, they are not presented here. We note that these two summations are the only ones which include the unknowns Φ_i: they can then be moved to the left side of the equation. When the equation (13) is repeated for the n wing nodes (including the tip), a set of n simultaneous equations results:

$$[\ A(n,n)\ ]\ [\ \Phi(n)\ ]\ =\ [\ RHS1(N)\ ] \tag{17}$$

A(n,n) is the matrix of coefficients: it shows the influence of wing and wake panels on wing node potentials; $\Phi(n)$ is the vector of the wing potential solutions; RHS1(N) is the right side vector: it includes the potentials of equations (12), (15) and (16):

$$RHS1(N) = \Phi_\infty + \Phi_{VOL} + \Phi_{SHOCK} \tag{18}$$

A Kutta condition is applied on lower trailing edge nodes: it equates the tangential speeds of the upper and lower surfaces. In the incompressible case, Φ_{VOL} and Φ_{SHOCK} vanish (σ_i=0 for all elements and no shocks) and we get the familiar system of integral panel methods. Since the exact values of σ_i for each field element and the number and location of shock surfaces are initially unknown, an iterative procedure is needed to solve the system. Basically, the incompressible case is evaluated first, giving wing node potentials. These are used to find the potential values of field nodes. Then, the iterations begin: σ_i values are computed from field nodes potential as are shock surfaces. A new RHS1 term is calculated with these values.

The residue is then calculated: [RES] = [A] [Φ^{OLD}] - [RHS1^{NEW}]. If the residue is small, we quit the iterative loop; if not, we calculate the new wing potentials [Φ^{NEW}] = [A]$^{-1}$ [RHS1^{NEW}] and then new field node potentials and start the loop again. We see that the matrix [A] and its inverse [A]$^{-1}$ never change. The field potentials Φ_F are computed with a similar version of equation (13):

$$\Phi_F(x_p,y_p,z_p) = \Phi_\infty - \Phi_{WING} - \Phi_{WAKE} + \Phi_{VOL} + \Phi_{SHOCK} \qquad (19)$$

where Φ_{WING} and Φ_{WAKE} are discretizations of the S_a and S_b integrals that were included in [A] for wing nodes. Each of these steps will be detailed in Chapter 5.0.

CHAPTER 4
PROGRAM MODIFICATIONS

During the program analysis, some modifications that can improve the computing time at the expense of a negligible loss of accuracy were found. Also, there were changes to clarify the program structure and new output calculations were added. The modifications can be grouped into six main sections:

4.1 DOUBLE TO SINGLE PRECISION ACCURACY

The original program was written completely in double precision accuracy. While this double precision increased accuracy of numerical results by reducing round-off errors, more memory was needed to run the program and computing time became very large. Thus, we wanted to test a single precision version that would still give acceptable results while reducing memory needs and CPU time. This was done by changing all the concerned variables and the FORTRAN intrinsic functions to single precision status. Also, it was required to change the test values SMALL (main program) and ABSV (XYZTRANS subroutine) to 0.00001 to account for larger round-off errors. Finally, the IMPLICIT statements at the beginning of all the subroutines were deleted.

4.2 SIMPLER INTEGRAL VERSION FOR FAR FIELD NODES

The discretization of the integral:

$$-\int_{Sa} \Phi(\overline{n}\cdot\nabla\Phi_L)ds \equiv -\Phi_{WING} \qquad (20)$$

can be simplified for far field nodes in equation (19). This presents an excellent opportunity of cutting computing time since the discretization of (20) is made of a

lot of program lines and operations. This complexity is not required for far field points, especially with a sizeable field grid. Using equation (6), it can be shown that:

$$\nabla\Phi_L = -1/4\pi\ \nabla(1/|\bar{r}|) = \bar{r}/[4\pi|\bar{r}|^3] \tag{21}$$

Discretizing equation (20) for each wing panel and using (21), we get:

$$-\int_{Sa} \Phi(\bar{n}\cdot\nabla\Phi_L)ds = -1/4\pi\Sigma\{\ \int_{Si} [\Phi(\bar{n}\cdot\bar{r})/|\bar{r}|^3]\ ds\ \} \tag{22}$$

Equation (22) is further simplified by supposing a constant potential value Φ_C over the panel surface. Φ_C is quickly calculated as the average of the three panel node potentials:

$$\Phi_C = [\Phi_1 + \Phi_2 + \Phi_3]/3.0 \tag{23}$$

Φ_C could also have been taken as the potential value at the center of the panel, using equation (14) with relative coordinates. However, the resulting potential is almost similar to Φ_C at the expense of a few more calculations. Using the definition of $\bar{r}$ [equation (6)], the vector $\bar{r}$ can be expressed in terms of the field point relative coordinates:

$$\bar{r} = x_R\,\bar{i} + y_R\,\bar{j} + z_R\,\bar{k} \tag{24}$$

Since $\bar{n} = \bar{k}$, the scalar product $\bar{n}\cdot\bar{r}$ gives:

$$\bar{n}\cdot\bar{r} = z_R \tag{25}$$

where z_R is the z coordinate of the evaluated point in local panel coordinates

(x_R,y_R,z_R). This value is constant for all panel points (they all lie in the plane z_R=0.0). Finally, the term $|\bar{r}|^3$ can be simplified in (22) by assuming that the radius $\bar{r}$ is measured between the field node and the center of the panel $(x_{CPi},y_{CPi},z_{CPi})$ in general coordinates:

$$|\bar{r}| \approx |\bar{r_c}| = \sqrt{[(x_{CPi}-x_p)^2+(y_{CPi}-y_p)^2+(z_{CPi}-z_p)^2]} \qquad (26)$$

Again, equation (26) applies for far field nodes only. With equations (23), (25) and (26), the integral in (22) now reduces to the area of the evaluated panel SP_i and we get:

$$-\int_{Sa} \Phi(\bar{n}\cdot\nabla\Phi_L)ds \approx -1/4\pi\ \Sigma[\ \Phi_C\cdot z_R\cdot SP_i\ /|\bar{r_c}|^3\ ] \qquad (27)$$

From the Fig.6, we see that the surface SP_i can be evaluated by $(x_2\cdot y_3)/2.0$. For each field point, the vector $\bar{r_c}$ will be calculated. If the vector's norm is bigger than the specified value, then the simplified version is used; if not, we choose the complete expression. In the program, the specified value is set by the user as a fraction of the half wing span (FRAC variable).

4.3 LOWER WING PLANE NODES POTENTIAL EVALUATION

As it can be seen on Fig.8, the field grid was designed such that some field nodes have exactly identical coordinates. This particularity occurs at the upper and lower wing planes that split and then meet again to body-fit the upper and lower wing surfaces. During program operation, the field nodes body-fitted to the wing are assigned the same potential values that were calculated for the corresponding wing nodes. However, for the remaining field nodes around the wing (zones 1,2,3,4 & 5 on Fig.11), the program repeats the potential calculations for both planes. This is unnecessary because since these nodes both have the same coordinates, the

potential of equation (19) will come out exactly the same.

Thus, the potentials are still evaluated for the upper wing plane nodes around the wing but for the lower wing plane, potentials values of the associated upper node are given thus saving computing time. There is an exception though: in the wake region (zone 5 on Fig.11), it was found that upper and lower potentials are slightly different even if they have the same coordinates. This is explained by the wake discontinuity mentioned in section 2.2. Thus, complete calculations are made for these lower wing plane wake nodes.

4.4 NEW VECTORS

In the original program, some calculations were unnecessarily repeated several times. During the evaluation of equation (16), the calculation of the values of x_{ci}, y_{ci}, z_{ci} and SS_i were repeated for each field node. Since these values don't vary from one node to another, they are now calculated only once right after shock capture and their values are stored in new vectors part of COMMON groups SHOPUP and SHOPLO.

Similarly, some calculations in subroutine UV were done for each node of each field element (during σ_i evaluation) while they were only eight possible cases altogether. We changed the dimension of the corresponding array (DNLOC1) from (8,3) to (8,3,8). The eight cases are now evaluated only once and stored in the third dimension. This saves a lot of calculations at the expense of very few extra memory.

Finally, the dimension of the vectors of the second derivatives of potential at the field nodes Φ_{XX}, Φ_{YY}, Φ_{XY} , ...(PSXX, PSYY, PSXY, ...) was changed from NGLOB (total number of field nodes) to eight. This can be done since only the

derivatives at the eight nodes of the current element are required (σ calculation, subroutine UV). There is no reason to keep the other (NGLOB-8) values in memory unless someone would want to see these values in a printout. With this change, memory requirements are greatly reduced.

4.5 WING LIFT COEFFICIENT

We decided to implement in the program a wing lift coefficient (CL) calculation loop. This was thought to be convenient for transonic studies. References 7 and 8 were used in the development of the related equations:

$$\begin{aligned} CF_X &= -1/S_{REF}\ \Sigma(CP_i \cdot SP_i/COSX_i) \\ CF_Z &= -1/S_{REF}\ \Sigma(CP_i \cdot SP_i/COSZ_i) \\ CL &= CF_Z \cdot \cos\alpha - CF_X \cdot \sin\alpha \end{aligned} \qquad (28)$$

where CF_X and CF_Z are the total forces coefficients along the X and Z axes, S_{REF} is the wing planform reference area, CP_i are the individual panel coefficients of pressure, SP_i the panel area and $COSX_i$, $COSZ_i$ the cosines of the angle between the general and panel relative axes. Negative signs are required to respect the Cp and cosines sign conventions. These equations are only valid for symmetrical flows (β=90°). The reference area S_{REF} is calculated in the program by using a Computer Drafting equation applicable for n-sides polygons (from Ref.9).

4.6 OTHER MODIFICATIONS

A few subroutines were grouped together: subroutine FI initially consisted of subroutines FI1, FI2 and FI3, while SHAP1 was divided into subroutines SHAP1, DETM1 and GLOB1. This way, some repeated lines are deleted and the structure is simpler. Also, we deleted unnecessary calculations that were repeated when the

influence of the other side of the wing is evaluated for node potentials. In these cases, only the y coordinates change, so it is not required to calculate again the same values of x and z. Cutting a few lines of operations repeated a few thousand times can show up in the long run. New variables were added to better monitor the program (ITMAX, TEST,...) as were some output messages. Finally, the incompressible case (M_∞=0.0) can now be run by the program: an IF was added for that purpose in the panel CP calculation loop (CP = 1 - [panel speed]^2 when M_∞=0 and V_∞=1.0). The main iterative loop (calculation of σ_i and field nodes potentials) is also skipped completely in that case.

CHAPTER 5
PROGRAM PRESENTATION

This section intends to present the main features of the program. A fully commented version is presented in appendice 1 and can be consulted for more details. The program is written in the FORTRAN 77 (version VSFORT) language.

5.1 INPUT FILE

This file includes the wing grid points only. The data should be given in the following form:

x(1)	y(1)	z(1)	
.	.	.	
x(i)	y(i)	z(i)	(29)
.	.	.	
x(n)	y(n)	z(n)	

There are no special writing format required. The wing root must be located at y=0.0 because of the calculations concerning the influence of the other side of wing. Other y values are negative. Equal y values are required for a wing section. The grid numbering is shown on Fig.12. There are NH wing sections and NV points around a section. NV must be an odd number. Wing node numbering starts from the tip trailing edge and is increased going inboard. There is one exception: node 1 is repeated on the lower trailing edge. This is done in order to join the wake upper and lower surfaces at the wingtip (Fig.4). Finally, tip grid nodes are added by the program as shown on Fig.4 and don't need to be included in the input file.

5.2 MAIN PROGRAM

A few parameters have to be set by the user. Variables NH,NV,NO and NSHO are set in the PARAMETER section of the main program and also at the start of each subroutine. This is required because of the use of COMMON groups. NH and NV are related to the input file. NO sets the number of field elements around the wing (Fig.8) and is chosen by experience. NSHO is the maximum number of shock surfaces over/or under the wing; since this is initially unknown, it may need to be changed later (appropriate message in subroutine UV). These vector dimensions are set in order to reduce memory requirements.

The other important variables are set after the COMMON declarations. DMM is the free stream Mach number M_∞; DG0 sets the size of the field elements: it becomes larger as they are further away from the wing (Fig.13); FRAC is used to determine whether the simplified version of the wing panel integral is used or not for field nodes (section 4.2); ALPHA,BETA AND GAMMA are the flow angles of attack α, β and γ inputted in degrees; ITMAX sets the maximum number of iterations allowed (safety measure) and TEST is the value under which the total residue must be to stop the iterations.

The main program follows an iterative algorithm to solve the non-linear compressible cases (s/r stands for subroutine):

1-set important parameters (M_∞,α,β,γ,DG0,TEST,...);
2-create wing, wake and field grids:
-read wing points & create wing panels (s/r XYZL);
-create wake panels (s/r BNODE) and local coordinates (s/r XYZTRANW);
-evaluate parallel wing node grid (s/r GRIDB);
-calculate local wing panel coordinates (s/r XYZTRANS);

-create field nodes and elements (s/r XYZGN);

3-build coefficients matrix [A] (s/r BUILDA);

4-build right side [RHS1] for M_∞=0.0: [RHS1] = Φ_∞ (s/r BUILDA);

5-apply Kutta condition at trailing edge (s/r BC);

6-evaluate matrix inverse $[A]^{-1}$ (s/r LUDC);

7-compute incompressible solution $[\Phi]=[A^{-1}][RHS]$ (s/r LUBS and DO loop 53);

8-set all σ_i=0.0 and the shock surface counters (KSU for upper, KSL for lower) to zero;

9-evaluate initial Φ values of field nodes (s/r PHYGLOB);

10-compute σ_i of field elements and capture shocks (s/r UV);

11-for each wing node, calculate new right side term:

$$[RHS^{NEW}] = [\Phi_\infty + \Phi_{VOL} + \Phi_{SHOCK}] \quad \text{(DO loop 902);}$$

12-evaluate the residue = $[A^{-1}]$ $[\Phi^{OLD}]$ - $[RHS^{NEW}]$ for each wing node (DO LOOP 930);

13-if the $\Sigma[residue]^2$ is under the test value (DO loop 939), go to 16;

14-if not, calculate new wing potentials $[\Phi^{NEW}]=[A]^{-1}$ $[RHS^{NEW}]$ (s/r LUBS & DO loop 57);

15-compute the new field nodes Φ (s/r PHYGLOB);

-go back to step 8;

16-calculate Cp of wing panels and nodes (s/r PHYBXYZ);

-print final results;

The steps 10 to 15 are included in the DO loop 9001. The use of "copy" arrays is required for A and RHS1 (BA1 and BAR, DO loops 671, 901 and 931). These copies are needed because the matrix inverse $[A]^{-1}$ is stored in A (subroutine LUDC) and the solution vector from subroutine LUBS uses the RHS1 vector too. Also, Φ values for field nodes in contact with the wing surface simply take the same potential values as their wing counterparts (DO loop 900 and 950).

Step 11 is a direct application of the discretizations of Φ_∞ (eq.12, DO loop 901), Φ_{VOL} (eq.15, DO loop 689) and Φ_{SHOCK} (eq.16, DO loops 681 & 682). Influence of the other side of wing is included. Incompressible flows can be evaluated by the program: in this case, iterative steps 10 to 15 are bypassed and step 16 is executed right away. Different types of information are printed in the output section at the end of the main program. Potential values at the nodes and upper and lower shocks are generally the only ones needed but the user can print information about field elements and nodes if he wants it to by removing comment labels "C" in the right places. Cp values are outputted in subroutine PHYBXYZ.

5.3 PROGRAM SUBROUTINES

5.3.1 SUBROUTINE XYZL

This subroutine reads the wing surface points. The wing is then scaled to a half-span of 1.0 for normalization purposes. The extra points on the camber line (Fig.4) are computed as part of the tip grid. This is followed by the panel/nodes associations. These are stored in the array NODEB and panel numbering can be seen on the Fig. 14. Triangles are inversed for the lower surface (for symmetry). After, the subroutine evaluates the panels associated with each node for node Cp purposes. These are stored in the array NEB. Finally, the subroutine equates upper and lower trailing edge nodes coordinates (if they are not already the same). This is done in order that the distance between wake surfaces is infinitesimal to better represent theory (section 2 and equation 9).

5.3.2 SUBROUTINE BNODE

The subroutine associates the trailing edge nodes with the NH-1 wake panels. Four nodes (two upper, two lower) are required (Fig.7) with the exception of wake

panel #1 because of the repetition of node 1. Wake panels are numbered from the tip to the wing root.

5.3.3 SUBROUTINE XYZTRANW

This subroutine calculates the local coordinates of each wake panel node (as described in section 3.2 & Fig. 7) and the relative cosines. These are the cosines of the angles between the general axes X, Y and Z and the relative ones x_R, y_R and z_R.

5.3.4 SUBROUTINE GRIDB

This subroutine computes a parallel grid on the wing surface. Each node has new coordinates (X2,Y2,Z2) close to the originals (arrays X, Y and Z). These coordinates are used in subroutine BUILDA when the A matrix is assembled. They are required because of the way the wing and wake panels integrals were discretized: they prevent special cases when calculating the effect of a panel on one of its own nodes. These new coordinates are computed by using the distances with neighbouring nodes.

5.3.5 SUBROUTINE XYZTRANS

This subroutine evaluates the local coordinates of the wing panel nodes and the cosines of the angles between the relative axes and the general ones. An example of these relative cosines is given in Fig.15. The example shows their use to calculate the local coordinate x_R. Similar operations are repeated for y_R and z_R. The relative axes are shown on Fig.6 as it was described in section 3.1. The reference area of the wing S_{REF} is calculated for CL purposes. This is followed by tests on each panel to verify if the node #3 x_R coordinate is located between the

nodes #1 and #2 x_R coordinates. This is required by the discretization of the wing panel integral. If node #3's x_R is negative or superior to node #2's, there is a renumbering of the nodes (the origin is changed). Usually, only a few panels will be renumbered. In the main program, a loop allows only two trips to subroutine XYZTRANS: the requirement of another renumbering shows an error in the input file.

5.3.6 SUBROUTINE XYZGN

This subroutine creates the field grid around the wing. Based on the choice of DG0, the distances between field nodes are evaluated. Then, there is the assembly of vectors NGL and NGLT: these are used to associate field and wing nodes during body-fitting. Loop 1312 and 2312 calculate the coordinates of the upper and lower wing plane nodes respectively.

The field is divided in 5 sections around the wing (Fig.11) and field values are based on wing perimeter coordinates. Then, loop 900 does the body-fitting of field nodes in contact with the wing as on Fig.8. The upper part of the figure also shows the numbering conventions for nodes and elements. All other field nodes are then evaluated based on the location of the wing plane nodes: only the z coordinate changes. Finally, field elements-nodes association is done. The numbering convention is shown on Fig.16. Properties of each element (center,volume) is then computed. Also, the array DNLOC1 (used in subroutine SHAP1) is assembled at the end of the subroutine. This is the vector in which we added a new dimension to save unneccessary calculations.

5.3.7 SUBROUTINE BUILDA

This subroutine builds the important A matrix. The influence of every wing and

wake panel is calculated for each node (one matrix row). First, wing panel effects are evaluated. The FI subroutine is used as part of the discretization. The influence of the other side of wing corresponding panel is included (only the Y coordinate changes). Then, the effects of wake panels are computed, using subroutine FI3W. Lower trailing edge nodes are removed from the calculations (special case in subroutine BC). Finally, the subroutine evaluates the right side term RHS1 for the incompressible case (Φ_∞).

5.3.8 SUBROUTINE BC

The subroutine applies the Kutta condition to the lower trailing edge nodes that were removed from evaluation in BUILDA. The A matrix and the vector RHS1 are modified for these nodes. Speeds are equal on upper and lower trailing edge panels. This can be expressed using wing potentials by:

$$[\Phi(up1)-\Phi(up2)]/ds1 = [\Phi(low1)-\Phi(low2)]/ds \qquad (30)$$

where ds1 and ds are distances between the last two upper (up1 and up2) and lower (low1 and low2) trailing edge nodes (see Fig.17). Rearranging (30) gives:

$$\Phi(low1) - \Phi(low2) - \Phi(up1)ds/ds1 + \Phi(up2)ds/ds1 = 0.0 \qquad (31)$$

This equation is repeated for each wing section. The coefficients of the related potentials are set to 1, -1, -ds/ds1 and ds/ds1 respectively in the A matrix row of the number of the node (low1) while the right side RHS1(low1) takes 0.0. Thus, all other values on the matrix row are set to 0.0.

5.3.9 SUBROUTINE LUDC

This subroutine computes the inverse $[A]^{-1}$ of the [A] matrix using the LU decomposition method. The inverse is stored in array A. For this reason, the [A] matrix was copied into array BA1 previously in the main program.

5.3.10 SUBROUTINE LUBS

This subroutine finds the solution $[\Phi]=[A]^{-1}[RHS1]$ using the LU back-substitution method. The solution vector is stored in RHS1: values of Φ_∞ were copied previously in BAR in the main program.

5.3.11 SUBROUTINE PHYGLOB

This subroutine calculates the potential values of every field nodes using equation 19. Nodes meeting wing points (NGL values different from zero) are removed from the study. Then, the special case for lower wing plane nodes is done (section 4.3). These nodes quit the main loop at this stage.

The main loop continues for the other nodes by calculating the influence of the wing panels (Φ_{WING} in Eq.19). As we mentioned in section 4.2, a special version is used for far field nodes. The distance from the node to the panel center "R" is the test value. R is compared with FRAC·YHSPAN, with FRAC set by the user and YHSPAN being the wing half-span. The following algorithm is then used:

```
IF R is greater than FRAC·YHSPAN THEN
-use simplified version for both wing sides panels;

ELSE
-use complete version for current wing side
 (negative y coordinates);

-calculate R for other side of wing panel;

    IF R is greater than FRAC·YHSPAN THEN
    -use simplified version for other side panel;

    ELSE
    -use complete version for other side panel;

    END

END
```

The simplified version utilizes equation 27. The other version repeats the relationships developed in subroutine BUILDA (except that the coordinates X2, Y2 & Z2 are changed for X3, Y3 & Z3). The algorithm shows that if R is already greater than the specified value on the first side, then the other side will always use the simplified version too. However, this is not always true for the second case.

Afterwards, the influence of wake panels (loop 8000), Φ_{∞}, field elements (loop 689) and upper and lower shocks (loop 681 and 682) is computed. They use the same discretized versions described in the main program and again, the effect of the other side of wing is always included.

5.3.12 SUBROUTINE UV

This subroutine computes the σ_i values for each field element. As we saw in equation 2 and 3, σ is a function of Mach number, local speed and first & second

derivatives of the velocity potential Φ. First, we need to evaluate the first derivatives.

Loop 581 calculates the first derivatives of the elements PSXE, PSYE & PSZE (Φ_X, Φ_Y and Φ_Z). A method based on the eight nodes of the element only is used. First, the derivatives at the nodes PSX, PSY and PSZ are evaluated using only the eight potential values at the nodes and a special grid (DNGLO1 terms). This special grid for each node is calculated in subroutine SHAP1. When all values at the nodes are known, the eight first derivatives are averaged to give the element values. These in turn are used in the calculation of the element Mach number (DMG) and square of the speed (VV2).

Knowing the elements's Mach numbers, we can capture the upper and lower shock surfaces (DO loop 121 and 122). The procedure described in section 3.4 is used, and shock surface parameters are computed. Then, we need to do a special treatment concerning the field nodes derivatives. We explained that these derivatives were calculated for the eight nodes of each element. Then, at the same node, there are the eight derivatives of the eight element into which this node is included (Fig.18). In DO loop 600, these eight derivatives are averaged in order to get a single node derivative. When there is a shock, there is a discontinuity of the potential derivatives: the average is then made on the four derivatives before (PSX, PSY & PSZ) and after (U1, V1 & W1) the shock surface.

But not all the nodes are in contact with eight elements. This is why special cases are needed for the nodes on each sides of the field grid and the upper and lower wing planes. This is done in DO loops 23 to 413. The average is made over four or two elements in these cases. The upper and lower wing planes are included because each derivative needs to be divided by two to exclude derivative discontinuity.

DO loop 1581 then evaluates the second derivatives of the elements PSXXE, PSYYE, ... (Φ_{XX}, Φ_{YY}, ...) by using the same approach as before. Values at the eight nodes are calculated and then averaged using a special grid. If there is a shock surface on one side of the element, derivatives U1, V1 & W1 will be used for the four nodes following the shock and PSX, PSY & PSZ for the ones preceding it. Finally, loop 879 computes the σ term. If the element Mach number is over one, a special equation is used: this formula uses information of the preceding element (σ, Mach number) to introduce artificial viscosity. In the other case, equations (2) and (3) are used directly.

5.3.13 SUBROUTINE FI

This subroutine evaluates the terms F1, F2 and F3 part of the discretization of the wing panel surface integral (Φ_{WING} in Eq.19).

5.3.14 SUBROUTINE FI3W

This subroutine calculates the F3W term related to the discretization of the wake surface integral (Φ_{WAKE} in Eq.19).

5.3.15 SUBROUTINE PHYBXYZ

This subroutine computes the Cp coefficients for the wing panels and nodes. First, panels Cp are calculated in DO loop 7000. Then, the nodes Cp are evaluated: an average of the near panels's CP is made. This is usually done over 6 panels but there are exceptions on the perimeter. Tip panels are not used in taking the averages. When there are shocks, the average is done in front of the shock only. We then calculate the wing lift coefficient CL using equations (28). Finally, panel location & Cp are printed by wing sections and followed by tip panels

outputs. The subroutine ends with the nodes Cp output.

5.3.16 SUBROUTINE SHAP1

The last subroutine calculates the special grid used in subroutine UV at each node for derivatives computation. The results are stored in the DNGLO1 array.

CHAPTER 6
RESULTS AND DISCUSSION

In this section, we will present the results of the tests that were made to validate the modifications to the program. This will be followed by an evaluation of the precision of the program itself. Finally, we will present the main differences with Sinclair's method.

The tests were made on two types of wing. First, we used a simple wing for the validation tests. This low aspect ratio wing can be seen on the Fig.19. It is made of constant chord NACA 0012 sections. The aspect ratio of the complete wing is only 1.2 and the sweep angle is 14°. Four sections at y=0.0, -0.2, -0.4 and -0.6 are used, and the number of points around a section is 17 or 33 depending on the precision required.

The second wing used is the ONERA M6 (Fig.20). Six sections of 17 or 33 points can be used. The aspect ratio is 3.798 and the the sweep angle at 25% chord stands at 26°. All sections have the same symmetrical profile. We used this wing to test the program on a well known transonic case (M_∞=0.84, α=3.06°) and compare its results with experimental data.

6.1 COMPUTING FACILITY

All computing was made on the MUSIC system of the École Polytechnique. The main frame computer is of the IBM 7171 model. The jobs were sent using Job Control Language (JCL) files in batch processing. Interactive runs are not possible due to the memory requirements and computing time. We used the MUSIC execution classes presented in Table 1:

TABLE 1: MUSIC execution classes

CLASS #	MAX CPU (min)	REGION (Kbytes)
4	10	32000
5	20	32000
6	60	32000

Most of the runs were done using the execution class #4. Output retrieval was made using the OSJR program in MUSIC. The output files were stored in the D=7 output and CPU times were noted in output D=3.

6.2 SINGLE PRECISION MODIFICATION

We expected that this modification would reduce the memory needs and computing time. We tested the accuracy and CPU time by comparing the new results with the ones from the original program. We renamed the old code as DOUB.FOR and the new single precision version is referred to as SING.FOR.

First, we tested SING.FOR on the NACA 0012 wing with 17 points per sections. Right away, an error occurred in subroutine FI, stopping the program. This was due to the single precision accuracy: round-off errors produced some negative arguments for logarithms in the discretization. This occured for very small arguments only: new IFs were added to assure positive values in all cases. Also, SMALL and ABSV parameters were changed (see section 4.1).

The modification worked and we got significative reductions in CPU time. One of the tests that were done is presented in Table 2 for the NACA 0012 wing with 17 points sections:

TABLE 2: Single precision test for NV=17

NV=17, NH=4, NO=3, NACA 0012 wing
M_∞=0.8, α=1°, DG0=0.1, TEST=1X10^{-5}
DOUB.FOR: 6 iterations time: 2 min 19.01 sec CPU
SING.FOR: 6 iterations time: 1 min 42.11 sec CPU

In all cases, ß is set to 90° for longitudinal flow. We found that the value of TEST=1X10^{-5} was low enough to get accurate results. Smaller "TEST" values will keep the program in an never-ending loop (limited to ITMAX iterations) because the residues converge to a higher value. In double precision cases, smaller TEST values can be used but there is no apparent increase in accuracy. NO and DG0 are chosen by experience: here, the choice was not important because we only wanted to compare CPU time. Comparison of wing node potentials showed identical results up to the third-fourth digits, which is much acceptable. Such a difference will never show up on a pressure distribution curve. Thus, the change to single precision looked very promising.

Unfortunately, when we tried the 33 points per section wing, the results changed drastically. The Table 3 presents the results for a 33 points per section NACA 0012 wing:

TABLE 3: Single precision test for NV=33

NV=33, NH=4, NO=4, NACA 0012 wing
M_∞=0.7, α=0°, DG0=0.1, TEST=1X10^{-5}
DOUB.FOR: 6 iterations time: 11 min 06.71 sec CPU
SING.FOR: 11 iterations time: 14 min 39.54 sec CPU

We see that the CPU time now increased and nearly twice as much iterations were required (it took less time per iteration in the single precision case). Higher residue values appeared and this explains the lower convergeance rate. In addition, some discontinuities appeared in the results: three strong shocks surfaces occured near the trailing edge (upper and lower surfaces) while they were none in DOUB.FOR. These shocks may have been responsible for very large speeds on three panels at the trailing edge near the wingtip.

We tried to find the reason behind all these problems and only one explication seemed to make sense: the double precision accuracy is always needed by the program to work properly. Some subroutine use very small values (SHAP1, FI particularly) and round-off errors introduced by the single precision are simply too tough on the numerical system for 33 points (smaller panels leads to smaller distances, thus larger relative errors, etc.). It would seem that for the 17 points sections, these errors simply dissipated.

Because of this setback, we decided to keep on testing the other modifications on a double precision code. The single precision version could still be used for small input files.

6.3 OTHER MODIFICATIONS

We made a double precision version that includes the other proposed modifications (section 4), namely:

1) lower wing plane nodes potential evaluation;
2) new vectors;
3) other modifications: lines deleted, grouping of subroutines, etc.;
4) simpler integral for far field nodes;

5) wing lift coefficient calculation;

We verified these modifications in three steps:

6.3.1 MODIFICATIONS #1, #2 AND #3

The new code was named RIDOUB.FOR for test purposes. The first 3 modifications are not supposed to influence the final results: only the CPU time should change. In order to use the exact expression for all field nodes for the wing potential integral (and deactivate modification #4), we set FRAC=10.0 at the beginning of the program and ran several cases. The Table 4 presents the difference in CPU time with these 3 modifications for one of these cases:

TABLE 4: Test for modifications #1, #2 & #3

NV=33, NH=4, NO=4, NACA 0012 wing
M_∞=0.7, α=0°, DG0=0.1, TEST=1X10^{-5}
DOUB.FOR: 6 iterations time: 11 min 06.71 sec CPU
RIDOUB.FOR: 6 iterations time: 10 min 25.01 sec CPU

We see that the first three changes reduced CPU time by 41.7 sec. This reduction will get bigger on larger input files and high field grid NO numbers. Comparison of wing potentials showed identical results up to the ninth digit! This difference is very small, and it shows that the modifications #1-3 work well. The small difference is explained by the different number of round-off errors in some subroutines. Other tests at high Mach numbers showed an identical number of shock surfaces and similar wing potential values.

6.3.2 SIMPLER INTEGRAL FOR FAR FIELD NODES

The next step was the verification of the modification #4. Tests were made with different values of FRAC while keeping all other parameters constant. We compared the wing node potentials with the ones from DOUB.FOR and noted at which digit differences occured. The Table 5 resumes the results of these tests:

TABLE 5: Tests for simpler integral version

NV=33, NH=4, NO=4, NACA 0012 wing
M_∞=0.7, α=0°, DG0=0.1, TEST=1X10^{-5}

DOUB.FOR: 6 iterations time: 11 min 06.71 sec CPU

RIDOUB.FOR:

FRAC	iteration#	difference	CPU time
1.1	6	1st-2nd digit	4 min 50.32 sec
1.2	5	2nd-3rd digit	4 min 23.39 sec
1.4	6	2nd-3rd digit	6 min 00.97 sec

We see that an appropriate choice of FRAC can cut the computing time by about 50%! Also, the accuracy of the results is acceptable for FRAC greater than 1.1: again, such differences can hardly be seen on a pressure distribution curve. The gain on CPU time over the loss of accuracy is excellent.

We made other tests with lower values for FRAC. The lower limit seems to be 1.1. When FRAC=1.0, the program converges to a residue of 1x10^{-2} to 1x10^{-3}: it stops when the ITMAX number of iteration is attained. For lower values of FRAC, the program becomes unstable and actually diverges, residues being very large.

This can be explained by the way we discretized the problem (Eq.(27)): for a small radius $\bar{r}_c$, the denominator becomes very small and thus the induced potential grows very quickly. This phenomena starts to occur for field nodes near the wing root because of the proximity of a lot of panels. The high potential induce large speeds, resulting in negative arguments for square roots in element Mach number evaluation (subroutine UV). In these cases, a message will be outputted to inform the user before the program stops.

6.3.3 WING LIFT COEFFICIENT CALCULATION

Finally, we checked the results of the wing lift coefficient CL for different cases. At M_∞=0.0 and α=0.0°, we got CL=-0.002 for the NACA 0012 wing with 17 points sections. This value is perhaps acceptable considering the round-off errors (exact value is 0.0 for the symmetrical section). For 33 points sections, we increased α to verify the variation of CL: strange values were obtained. A check of the CP distributions explained these results: we found that at α=5°, the Cp at the trailing edge panels reached values in the order of -5 to -6, getting worse by increasing α to 10° and higher.

We proceeded on by testing a transonic case, M_∞=0.7 with α=5° and 33 points sections. This resulted in the program stopping after only a few iterations: the high speeds at the trailing edge resulted in a divergeance of the numerical system. The program stopped in exactly the same way at it was described in the previous section. This had never shown on previous tests because of the low α used. Also, it seems that the situation gets worse by increasing the number of points on the wing sections.

Since the incompressible case does not involve iterations and field calculations, we were able to test high α values and check the CL results. We had seen that most panels gave reasonable Cp's, so we excluded from the

summations (Eq.28) the trailing edge panels. This resulted in CL values that made sense with the changing α. Also, we compared the results for the reference surface S_{REF} (after half-wing normalization) with the analytical value of 1.667 for the NACA 0012 wing (Fig.19):

17 points sections -> S_{REF} = 1.698 relative error: 1.9%
33 points sections -> S_{REF} = 1.675 relative error: 0.5%

We see an increase in precision with the number of wing points. The small error is attributed to round-off operations. We conclude that the CL loop seems fine but that the trailing edge problem at α greater than 0.0° will need to be solved before final validation can proceed.

6.4 VALIDATION ON THE ONERA M6 WING

The next step was to test the program on the ONERA M6 wing for a well documented transonic case: M_∞=0.84 and α=3.06°. We used 6 sections of 33 points in order to get nice pressure distribution curves. When we ran the case, the program diverged completely and stopped: high element speeds produced negative square root arguments in subroutine UV. The same file was then tested for M_∞=0.0 and α=3.06°: Cp values were extremely high (-6,-8) as were the panel speeds (5,6) at T.E. nodes. For the other nodes, Cp values looked fine.

Thus, our testing showed that the original program already had a problem at the trailing edge nodes/panels for angles of attack different than zero. Since the problem also occurred at M_∞=0.0, we deduced that field potential and source σ calculations (subroutine PHYGLOB and UV) are not the cause. Again, the problem got worse by increasing α and the number of points on a wing section.

The problem occurs at trailing edge nodes and panels: thus, we can almost be certain that this is related to the Kutta condition or the interaction of this condition with the wing and wake panel discretizations. The next step would be to verify how well the Kutta condition applies to the [A] matrix and maybe test new ways of calculating the condition. Unfortunately, this problem appeared too late in our work to be analysed deeply and also was not part of the original requirements of our project.

However, when α is equal to zero, the pressure distribution around the wing seems adequate (there are no Cp discontinuities at the trailing edge). Thus, we decided to test the program on the ONERA M6 wing for M_∞=0.84 and α=0.0°. We set the field parameter NO to 4 in order to get very accurate results for the comparison. We made two tests with different values of the parameter FRAC: these tests confirmed previous results for the NACA 0012 wing:

For FRAC=1.2: CPU time is 20 min. 47 sec.;
For FRAC=10.0: CPU time is 33 min. 19 sec.;

We see that the use of the simplified version for far field nodes reduced the CPU time by 38%. This was done with a difference in wing node potentials and Cp values occuring only at the third-fourth digits. This shows very well the power of the simpler version. Also, we can note the influence of the parameters NV, NH and NO on the CPU time when we compare the results with the ones from previous tests on the NACA 0012 wing.

The comparison was made with an Euler solution using a finite-volume method on an 49x17x17 O-H grid. This code is accurate and gives a good indication on the validity of the program. We compared the pressure distributions of the two solutions at the 20%, 44% and 80% semi-span stations (Fig.21, 22 and

23). The integral program results follow the same general tendancy as the Euler solution. For all three semi-span stations, there are two Cp peaks: the sharp peak occurs at x/c≈0.05 while the flatter peak moves from x/c≈0.5 at 20% semi-span station to x/c≈0.3 at 80% semi-span station. In all cases, the Cp peaks are slightly higher for the Euler solution. Elsewhere, we see that both curves are very similar. This is especially true at the 20% semi-span station (Fig.21).

From this comparison, we can conclude that the 3-D transonic panel method gives results in the same range as finite-volume Euler programs for M_∞=0.84 and α=0.0°. We also know that the pressure peaks are lower in the case of the panel method. This difference seems to grow as we move toward the wing tip. However, the discontinuity at the trailing edge prevented us from testing the complete transonic case on which shocks appears. Thus, we cannot conclude about the accuracy of the shock representation.

6.5 DIFFERENCES WITH SINCLAIR'S METHOD

There are some different aspects between the program and Sinclair's developments (Ref.1 and 2). First of all, the field grids are different. We mentioned in the section 3.3 that a body-fitted grid was selected in order to simplify some calculations. For the present wing configuration, body-fitting is easily done and this prevents the necessity for special cases when some field elements are inside the wing body. Sinclair's method penetrates the wing body and needs these special tests. The program could be modified when the study of more complex configurations (wing with flaps deployed for example) becomes a requirement.

Another significant difference is the shock potential integral Φ_{SHOCK} (Eq.16). In Sinclair's methodology, the shock surface S_c (Fig.1) in not accounted for as part of the boundary surface. However, according to potential theory, this surface

should be included since if forms a discontinuity in the volume as is the wing and the wake. Noticeable differences between the outputs of the two methods could occur at high subsonic Mach numbers.

Finally, the third important difference is the way in which the first and second derivatives of the velocity potential of field nodes are calculated. In Sinclair's method, a finite-difference scheme is used. The potential values are calculated at the field nodes. These edge values are then used as boundary conditions for a finite-difference Approximate Factorisation (AF) method. The AF technique incorporates the required upwind influence in supersonic regions together with a stable fast solver.

Solutions are found by iteratively sweeping through the field in the direction of increasing X while solving in the Y direction (Successive Line Over Relaxation, or SLOR). This is required in order to capture accurately the shock waves and insure convergeance of the results. By comparison, our program does not solve iteratively for the derivatives of potential. Section 5.3.12 showed that these values are found directly by using only the potential values at the nodes of the volume elements and a special grid for each node calculation. There are special cases for the shock surfaces discontinuities since their location are known from the Mach numbers of the volume elements. It remains to show which one of these two methods is more efficient in terms of accuracy and computing time.

CONCLUSION AND RECOMMENDATIONS

At the beginning of this report, we set out the three main requirements of this project (Chapter 1). First, we needed to study in detail the original transonic integral program and from there, we wanted to make a well-commented version. Chapter 2 presented the integral formulation based on the Green's function approach and then showed how this formulation can be simplified to the form used by the program (Eq.13). Then, we explained in Chapter 3 how Eq.13 is discretized numerically. The discretizations of the wing surface, wing wake, field elements, shock surfaces and the resulting system of equations were presented.

The second requirement was to find modifications that could accelerate the computing time and reduce the memory needs at the expense of a negligible loss of accuracy. Chapter 4 showed six groups of modifications that were derived from the study of the program. The most important modifications were the change from single to double precision and the simpler integral version for far field nodes's potential calculation. We were then able to present the most important features of the program in Chapter 5. The input file and the basic tasks of the main program and subroutines were described. We showed how the iterative algorithm solves the non-linearity of the system of equations. The appendice 1 presents the commented version of the program.

The modifications were then validated in the first part of Chapter 6. The validation was done on a constant-chord, low aspect ratio wing of NACA 0012 section. Tests showed that the single precision modification could not be used on large input files (33 points per section). More iterations and a larger CPU time was required. This is explained by the fact that some calculations need the double precision accuracy. A new double precision version of the program using the other five modifications was written. Tests showed that these modifications worked well. The simpler version for far field nodes brought reductions of about 50% in CPU

time for differences of wing node potentials at the second-third digits. A good starting value for the related parameter FRAC is 1.2.

The third requirement was to test the program on a well-known transonic case, namely the ONERA M6 wing at M_∞=0.84 and α=3.06°. Unfortunately, we found out that for flows with α different than zero, some velocity discontinuities appear at the wing trailing edge. This causes high Cp values and complete divergeance of the numerical system in compressible cases. This problem is not related to the field elements or to the iterative scheme since it appears at M_∞=0.0. The problem gets worse with bigger input files. It seems that the problem is related with the Kutta condition or its interaction with the system of equations.

We then decided to test the program accuracy for a zero incidence (α=0.0°) ONERA M6 wing at the same Mach number (M_∞=0.84). The results were compared with the solution from an Euler code. We observed good general agreement between the two pressure distribution curves at different wing stations. The only remarkable differences are at the location of Cp peaks along the wing chord. Thus, the program seems quite accurate at zero incidence. We finished the report by comparing the program with Sinclair's method.

The next step is obviously to solve the trailing edge velocity discontinuity problem for α different than zero. The problem may be the way the Kutta condition is applied. The longitudinal speeds of the upper and lower trailing edge are equated for each wing section (section 5.3.8). This formulation works well for a 2-D wing but may not cover some 3-D aspects of the wing flow. When this problem is solved, the transonic test case (ONERA M6 wing at M_∞=0.84 and α=3.06°) should be tried. The shock capture efficiency will then be assessed and we will be able to conclude once and for all on the program accuracy.

While the CPU time has gone down appreciatively, some work could be done to lower it even more. Actually, most of the CPU time is related to the calculation of the source terms σ_i (subroutine UV). There are still a big number of calculations done concerning the special grids for each nodes (8) of each field element. However, these calculations are repeated at each iteration since the field grid never changes. Some results could be stored but the memory requirements will increase.

Finally, the program could be modified to study flows over complex configurations. The use of a field grid penetrating the different wing bodies would then be easier to use than body-fitted grids. Each subroutine would be modified to repeat calculations for the new surfaces. When this is done, we will truly be able to show the advantages of the 3-D transonic integral method.

REFERENCES

1. SINCLAIR, P.M., 1986, "An Exact Integral (Field Panel) Method for the Calculation of Two-Dimensional Transonic Potential Flow Around Complex Configurations", Aeronautical Journal (June-July), 227-236.

2. SINCLAIR, P.M., 1988, "A Three-Dimensional Field-Integral Method for the Calculation of Transonic Flows on Complex Configurations/ Theory and Preliminary Results", Aeronautical Journal (June-July), 235-243.

3. MASSON, C., 1989, "Méthode de panneaux 2D pour écoulements transsoniques", M.SC.A. Thesis, Department of Mechanical Engineering, École Polytechnique de Montréal.

4. NORMANDIN, F., 1990, "Étude 3D d'une aile d'avion et de son sillage en régime subsonique", M.SC.A. Thesis, Department of Mechanical Engineering, École Polytechnique de Montréal.

5. FANG, Z. and PARASCHIVOIU, I., 1990, "Final Report, Integral Method for Transonic Potential Flows", Department of Mechanical Engineering, École Polytechnique de Montréal.

6. MORAN, J., 1984, "Theoretical and Computational Aerodynamics", John Wiley & Sons, 2, 103-111, 260-277.

7. KAFYEKE, F., 1990, "Aérodynamique de l'avion II", Mechanical Engineering Course AE461 Notes, 1, 118-119.

8. HOUGHTON, E.L. and CARRUTHERS, N.B., 1988, "Aerodynamics for Engineering Students", Arnold Publishers, 3, 451-455.

9. DANSEREAU, J., 1988, "Éléments de CAO, Bloc 3: modélisation géométrique", Mechanical Engineering Course 2.420 Notes, 1, 3.168-3.169.

FIGURES

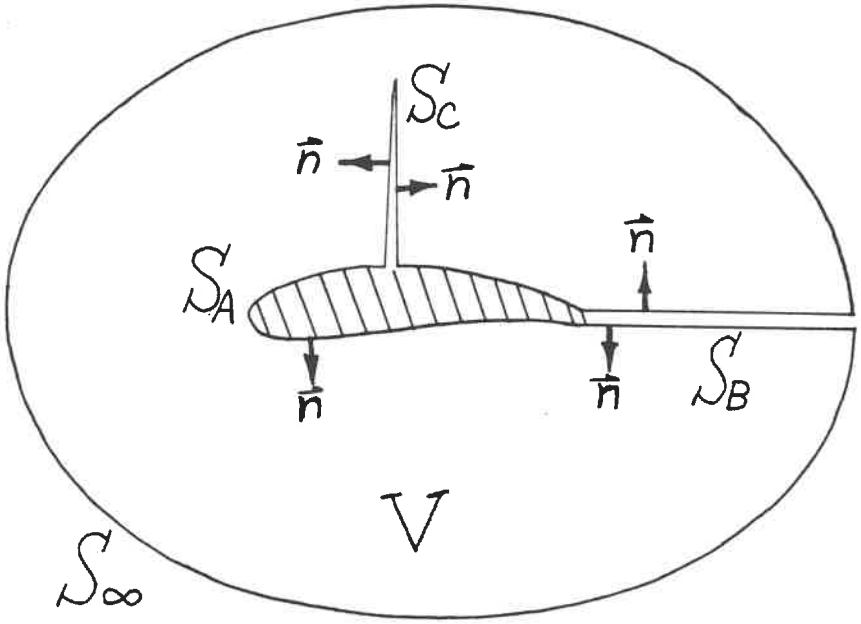


FIGURE 1: Surfaces and volume

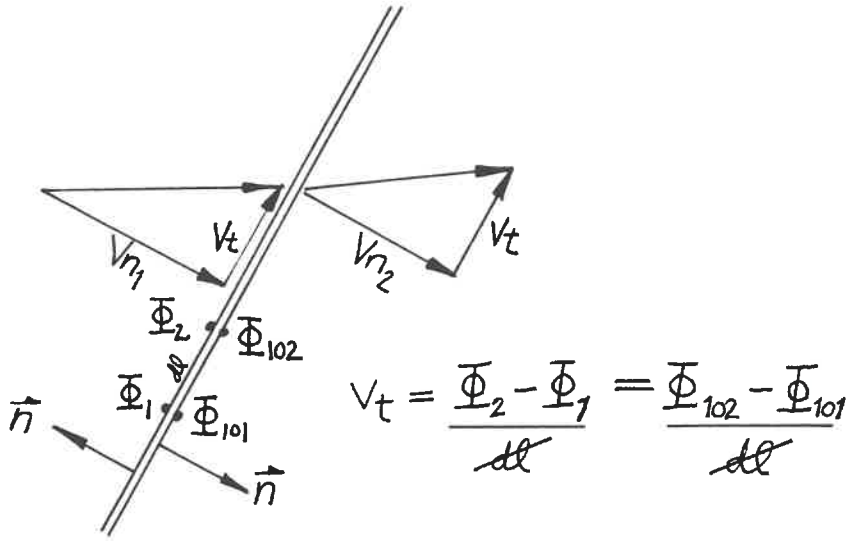


FIGURE 2: Shock potentials

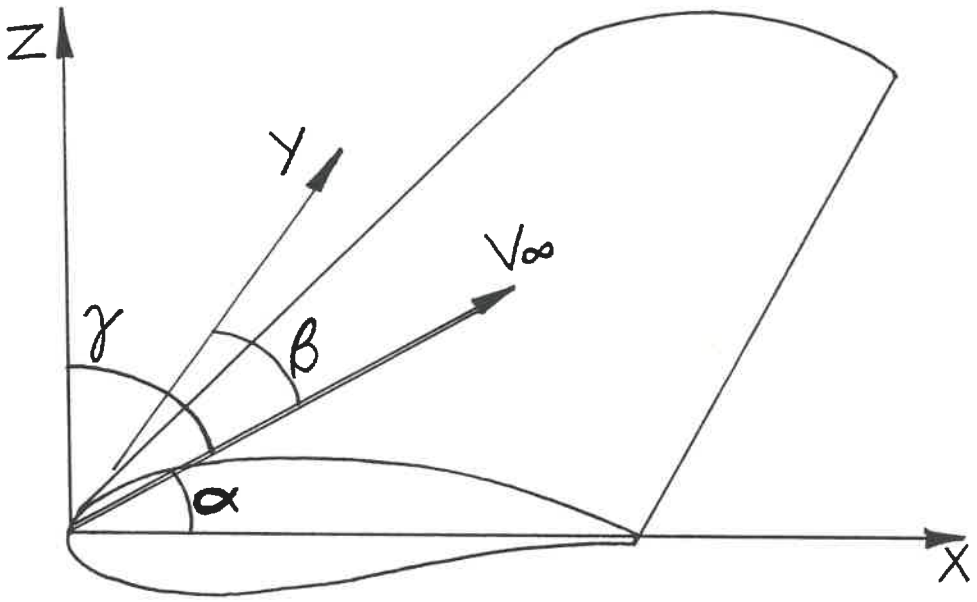


FIGURE 3: Flow angles

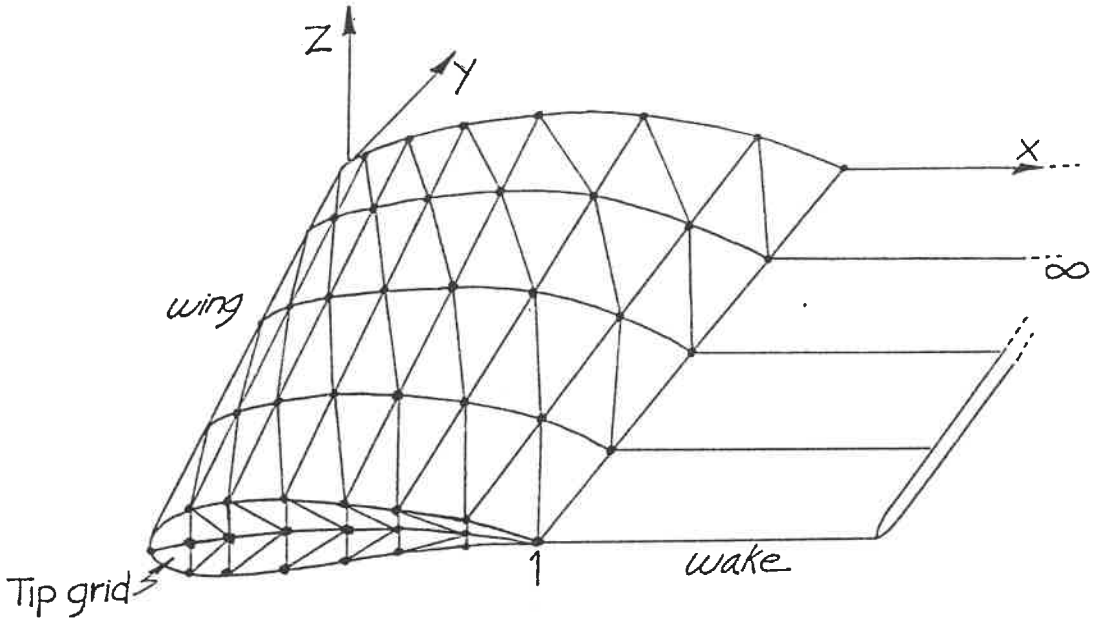


Figure 4: Wing and wake panels

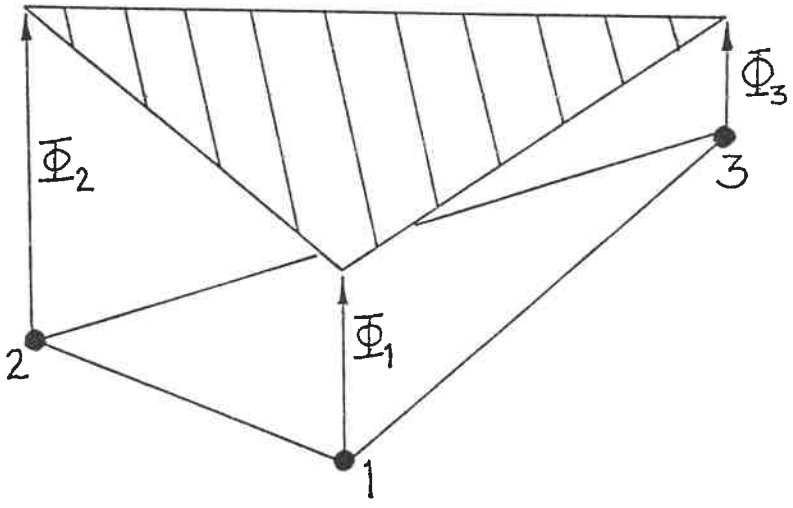


Figure 5: Linear variation of Φ

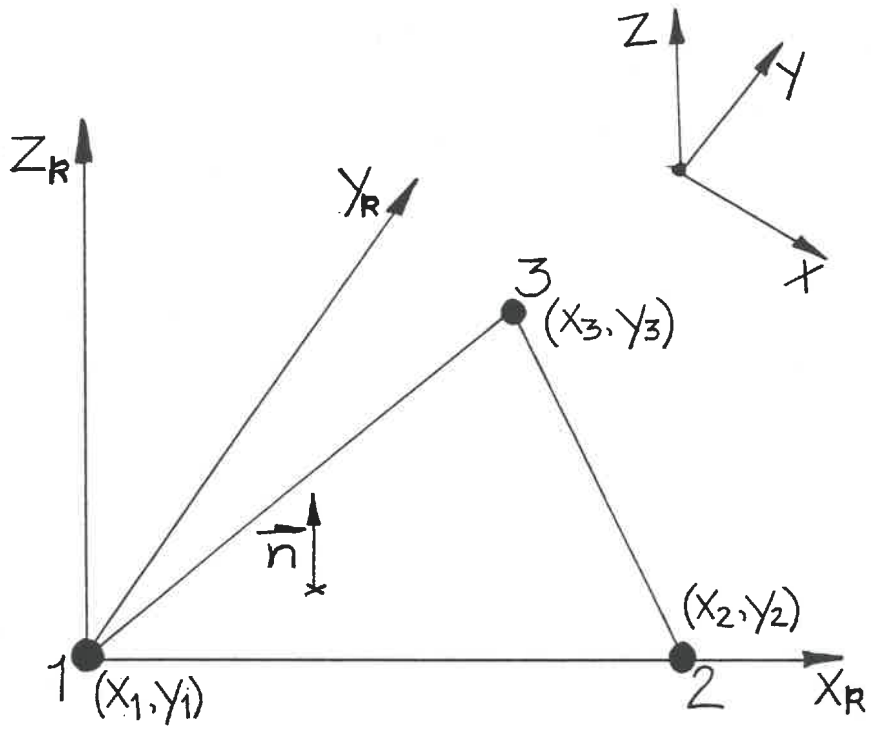


FIGURE 6: Local wing panel coordinates

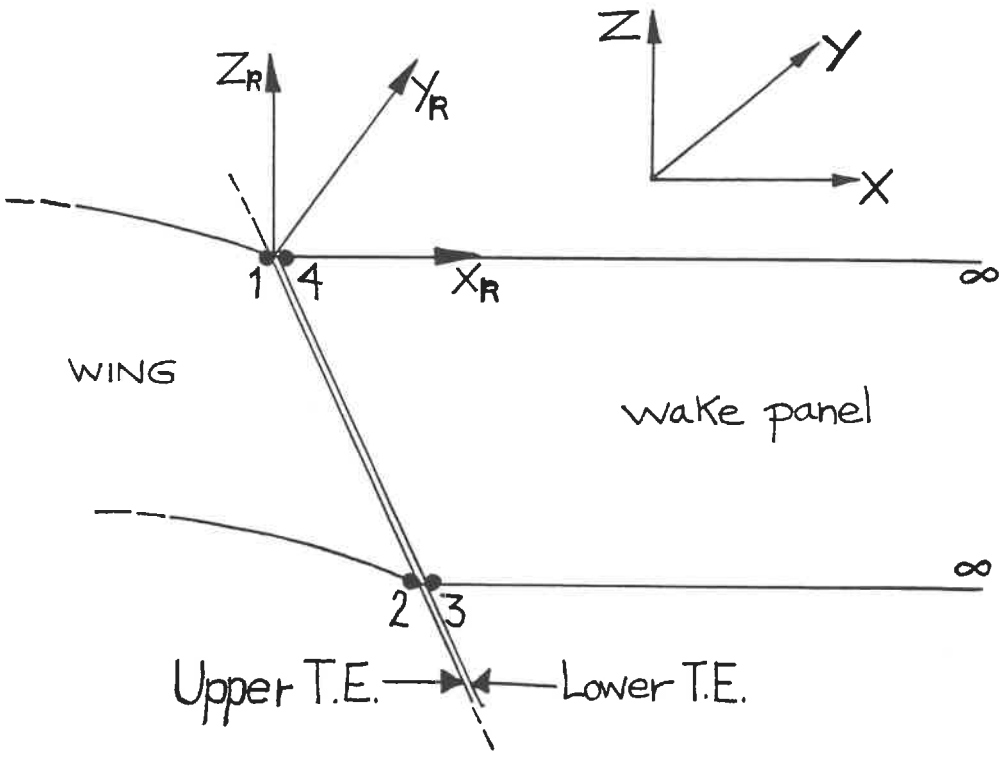


FIGURE 7: Local wake panel coordinates

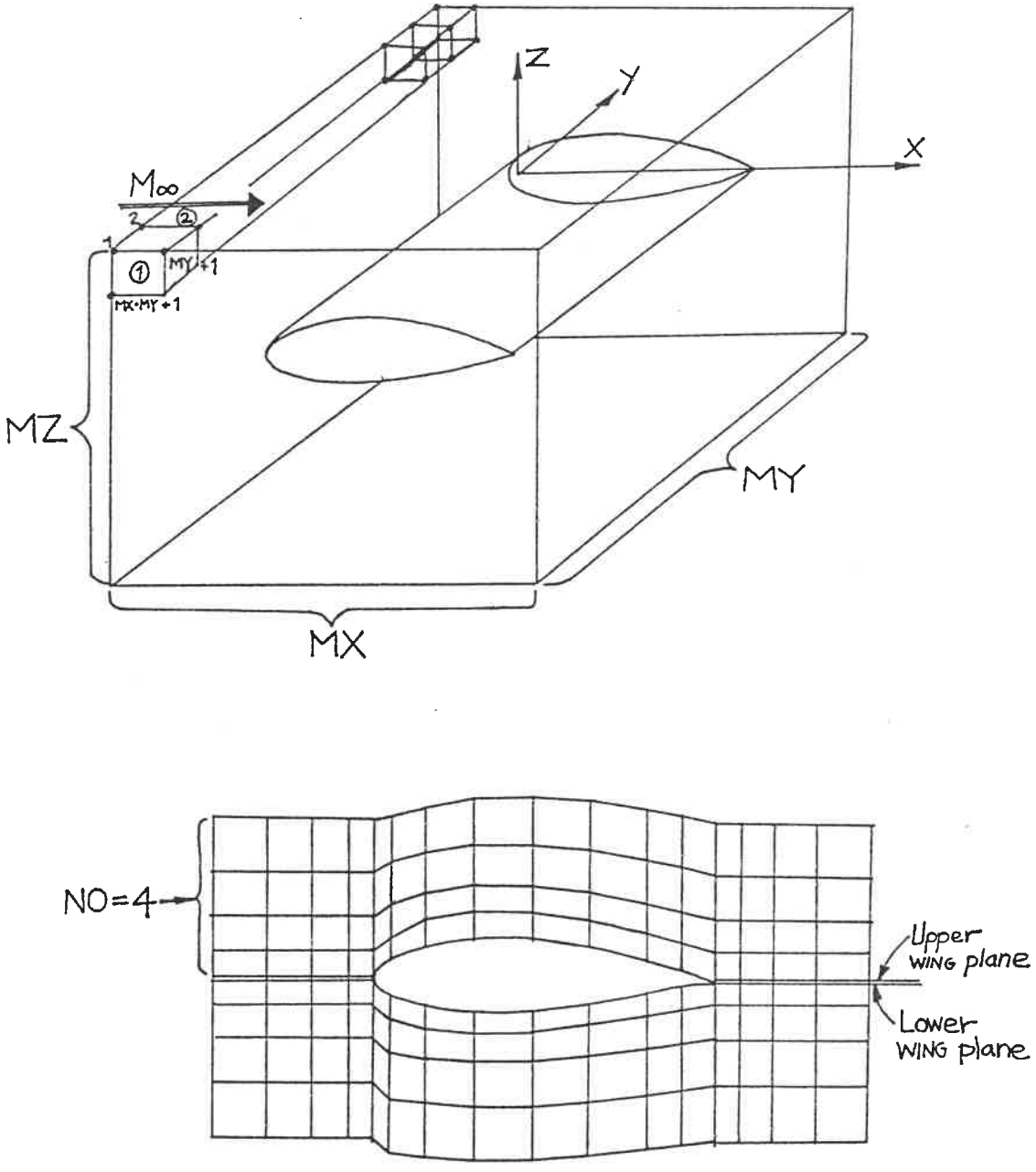


FIGURE 8: Field nodes and elements

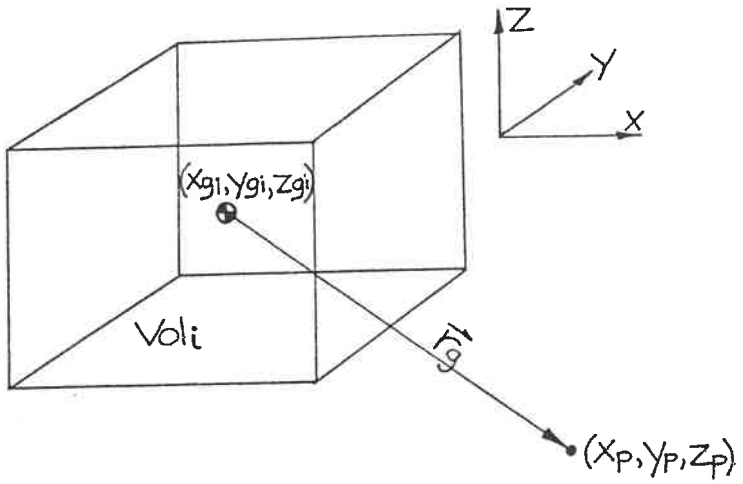


FIGURE 9: Volume integral

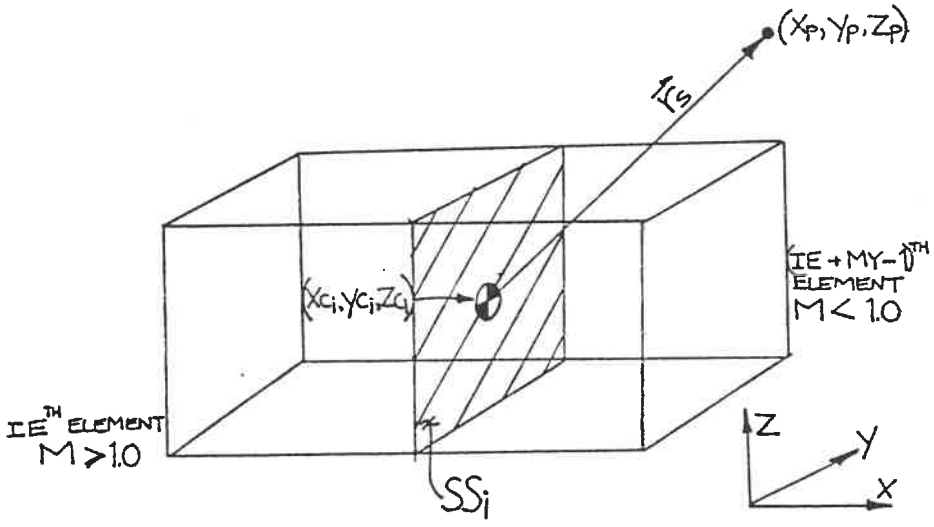


FIGURE 10: Shock integral

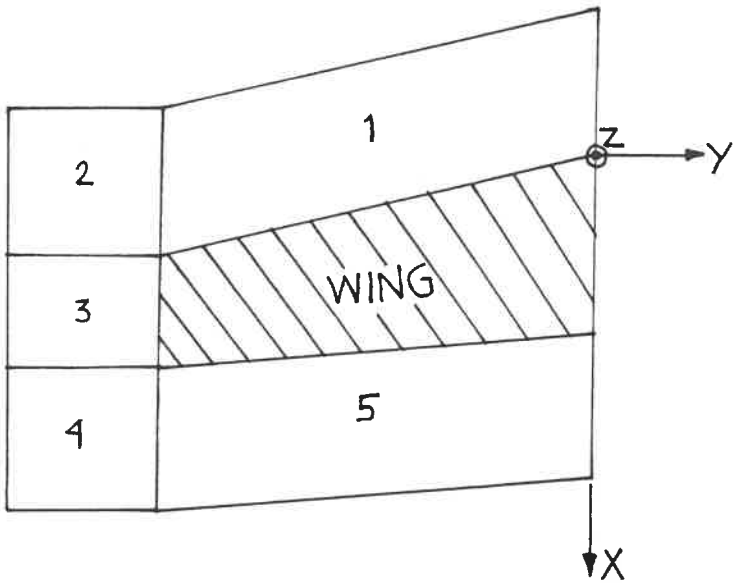


FIGURE 11: Field grid divisions

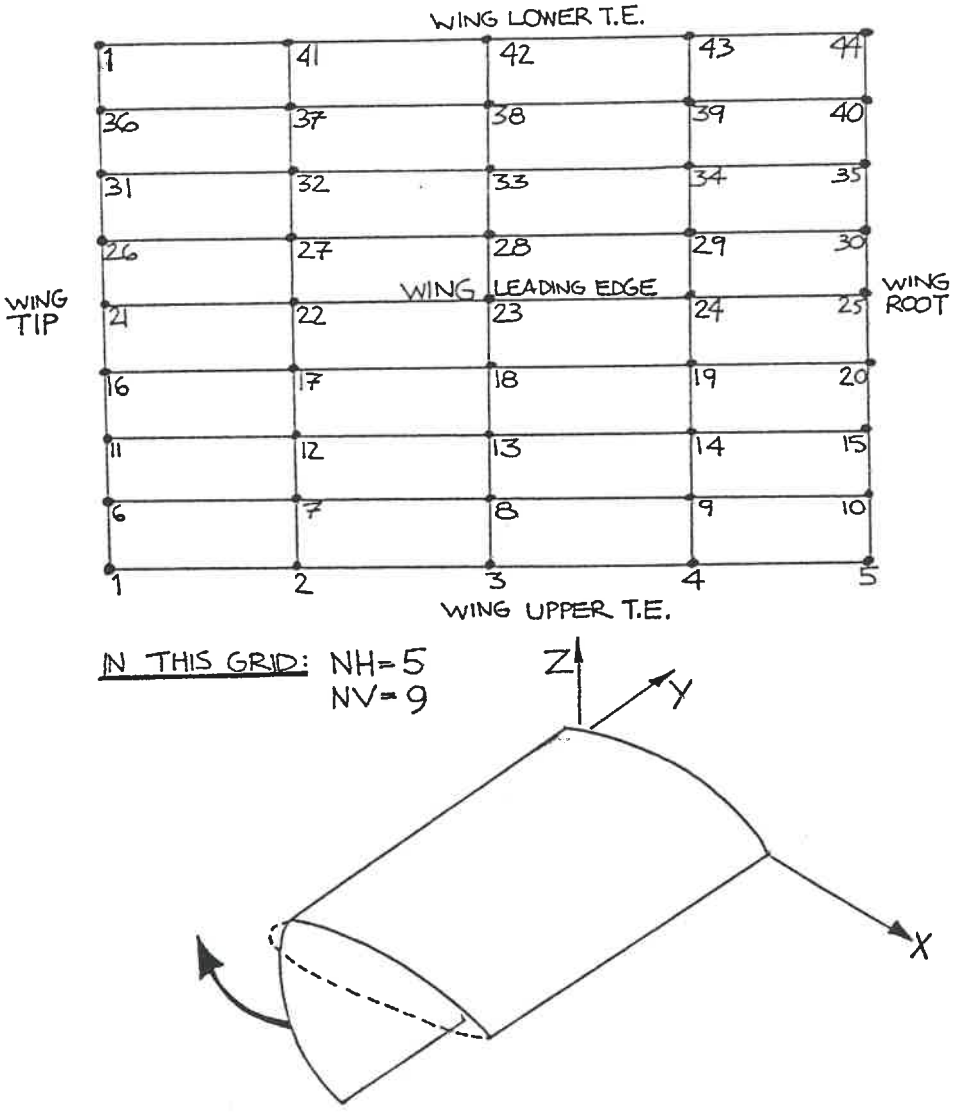


FIGURE 12: Wing grid numbering

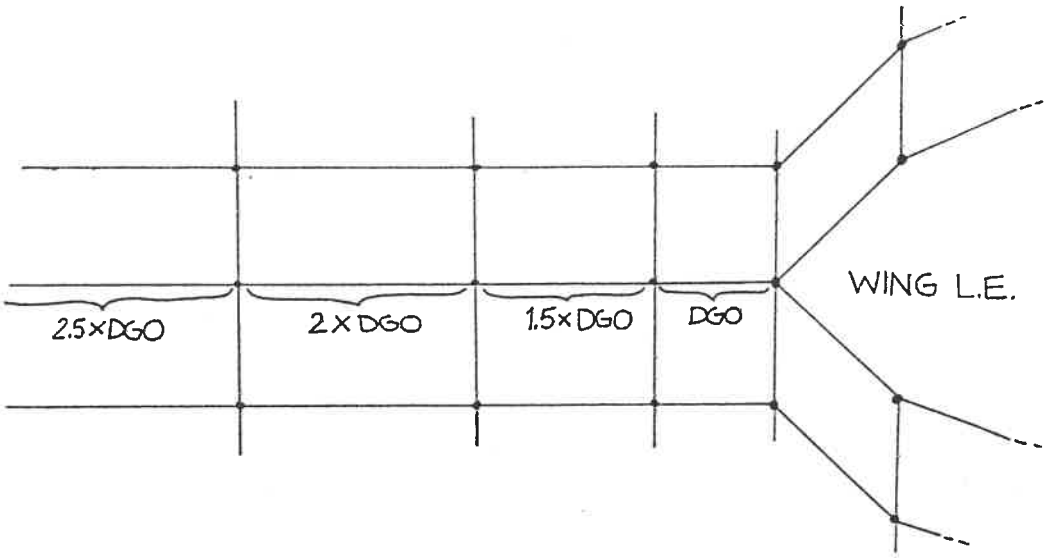
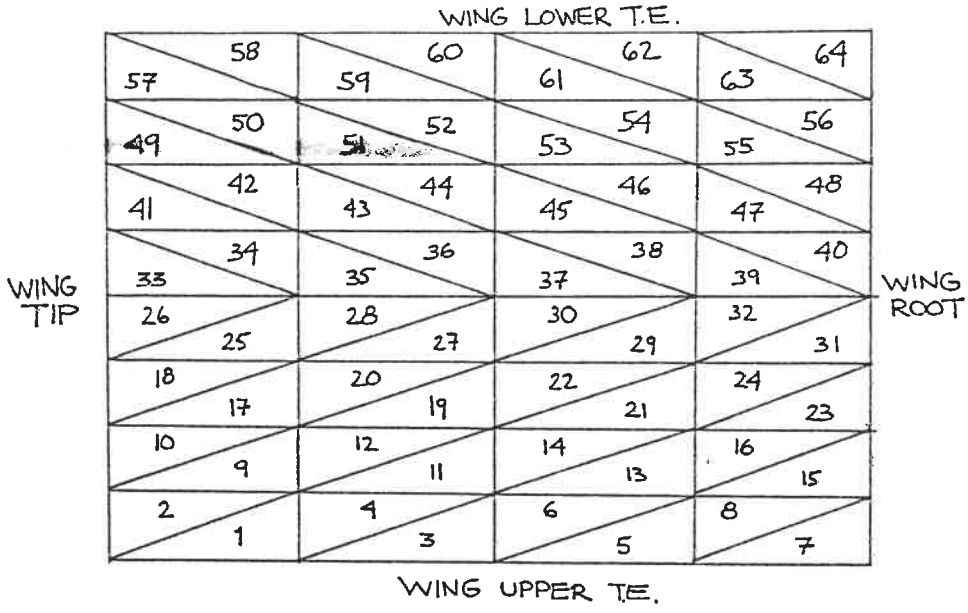


FIGURE 13: Influence of DGO



IN THIS GRID: NH=5
NV=9

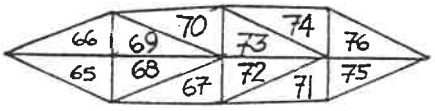
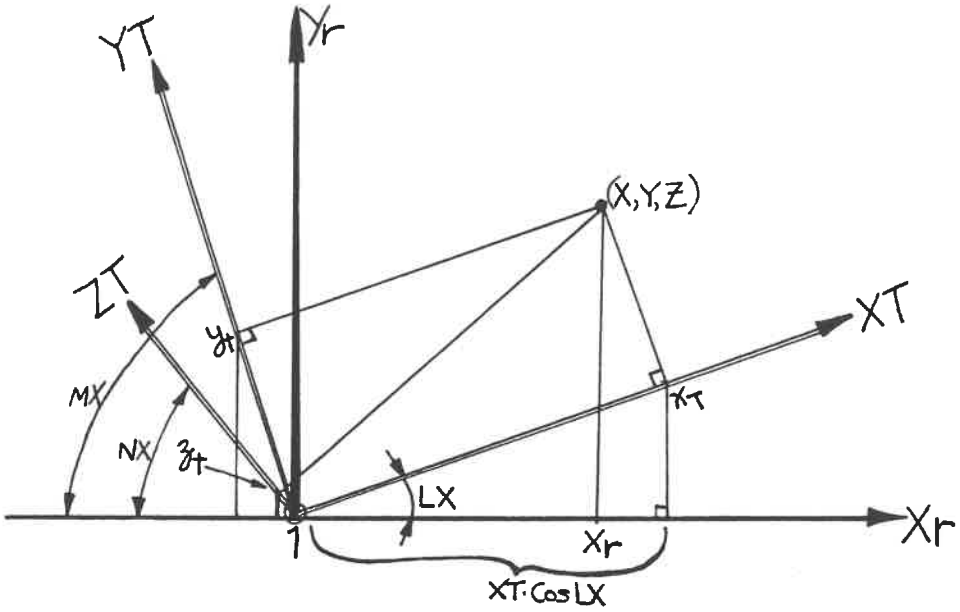


FIGURE 14: Wing panel numbering



XT, YT, ZT: general coordinates moved to panel origin 1;

x_T, y_T, z_T: point position in translated coordinates;

LX, MX, NX: angles between general axes and axis X_R;

$x_R = x_T \cdot \cos LX + y_T \cdot \cos MX + z_T \cdot \cos NX$

FIGURE 15: Relative cosines

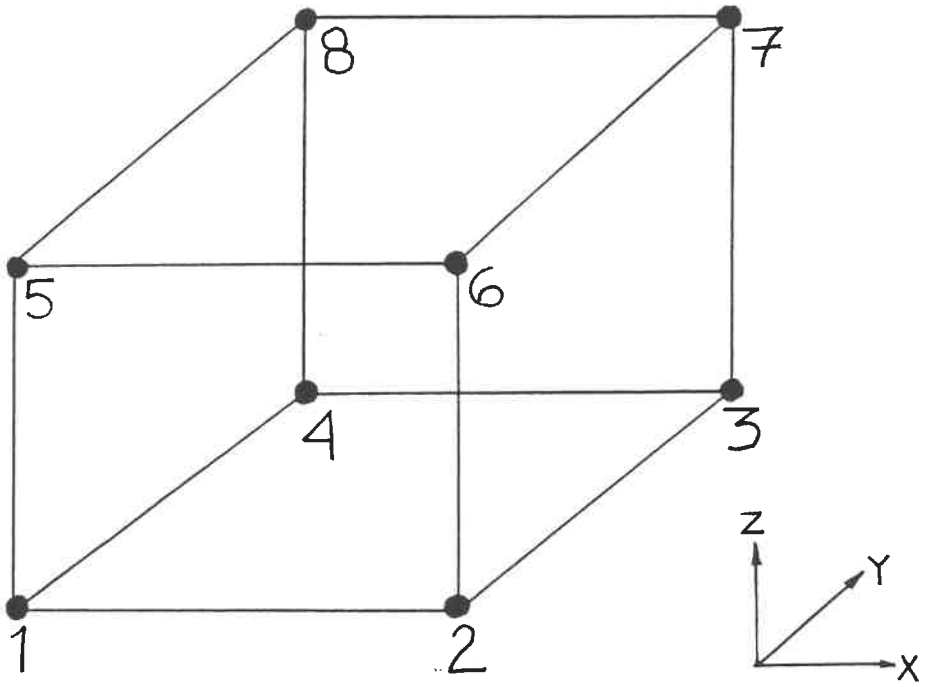


FIGURE 16: Element node numbering

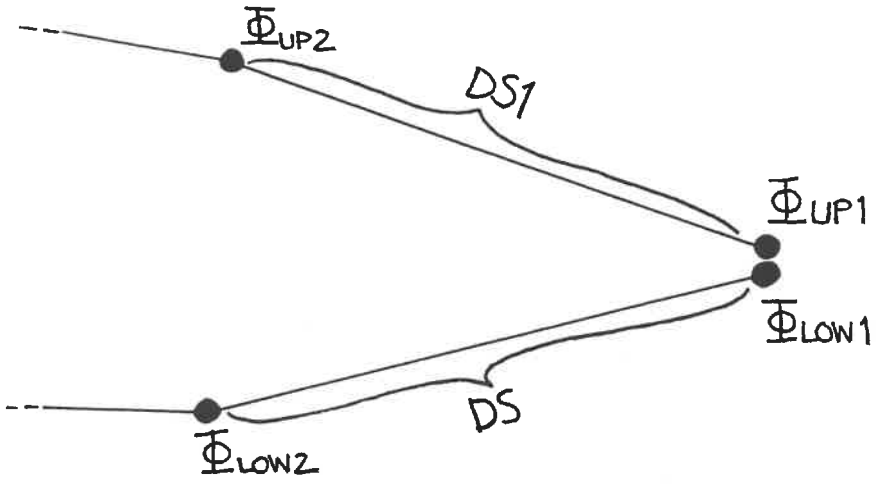


FIGURE 17: Kutta condition

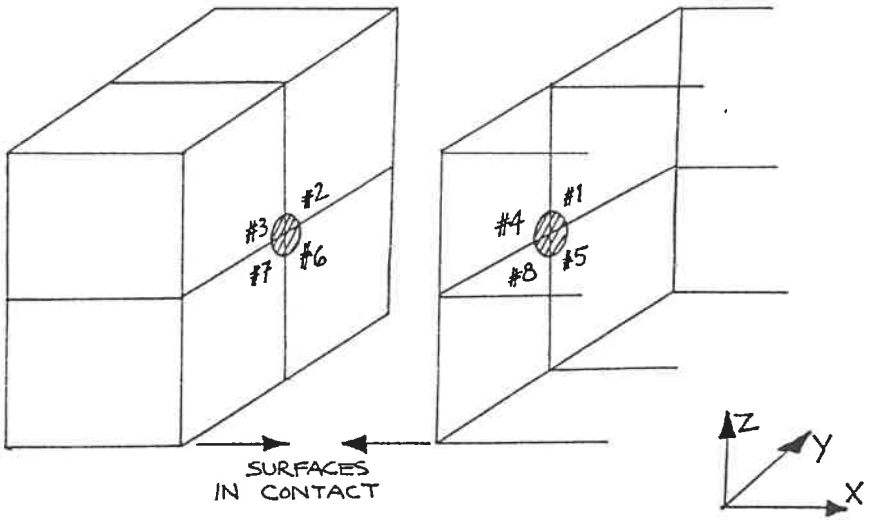
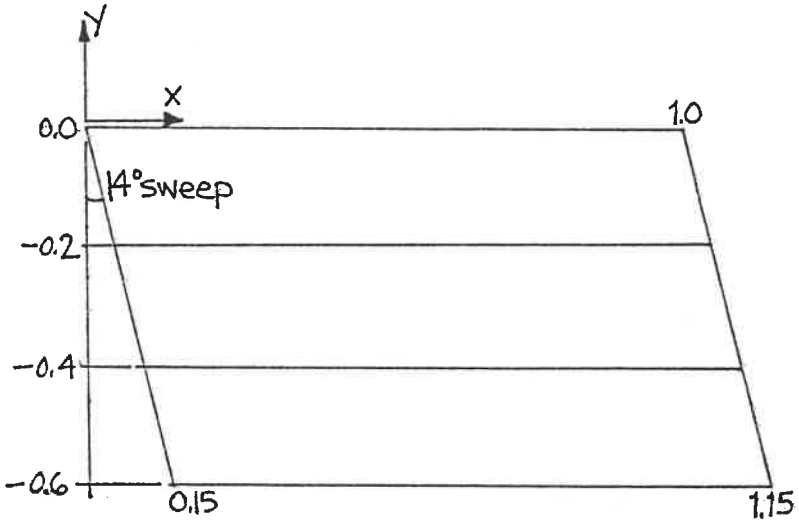


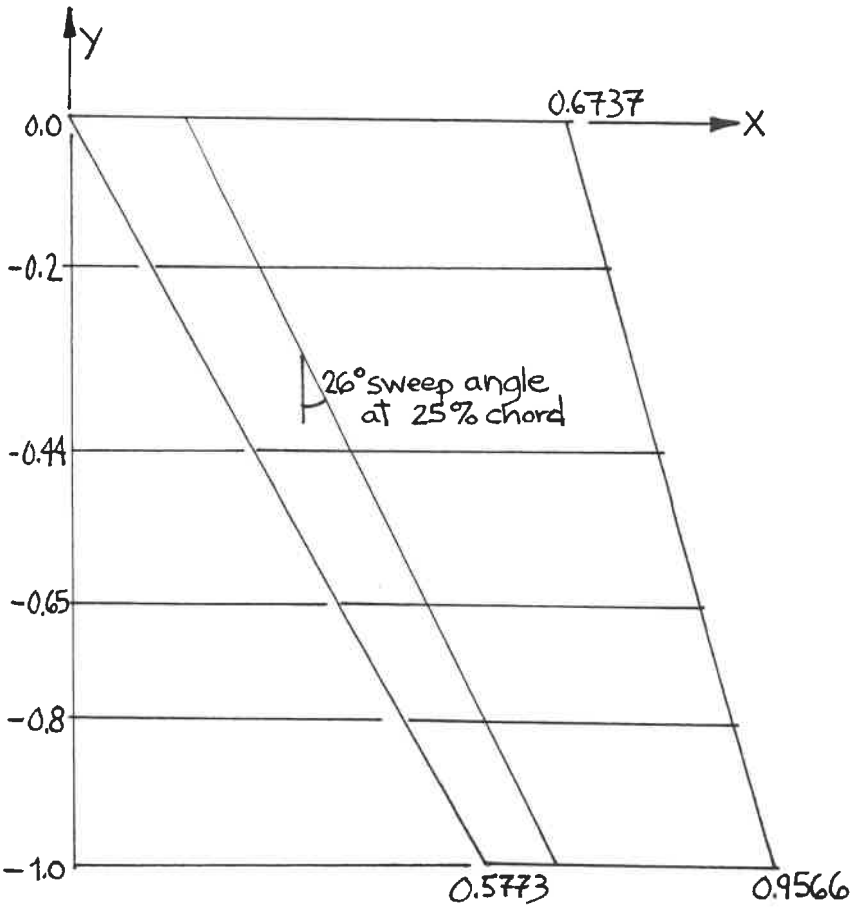
FIGURE 18: Field node derivative



Total wing area = 1.2

Wing aspect ratio = 1.2

FIGURE 19: NACA 0012 wing



Total wing area = 1.053

Wing aspect ratio = 3.798

FIGURE 20: ONERA M6 wing

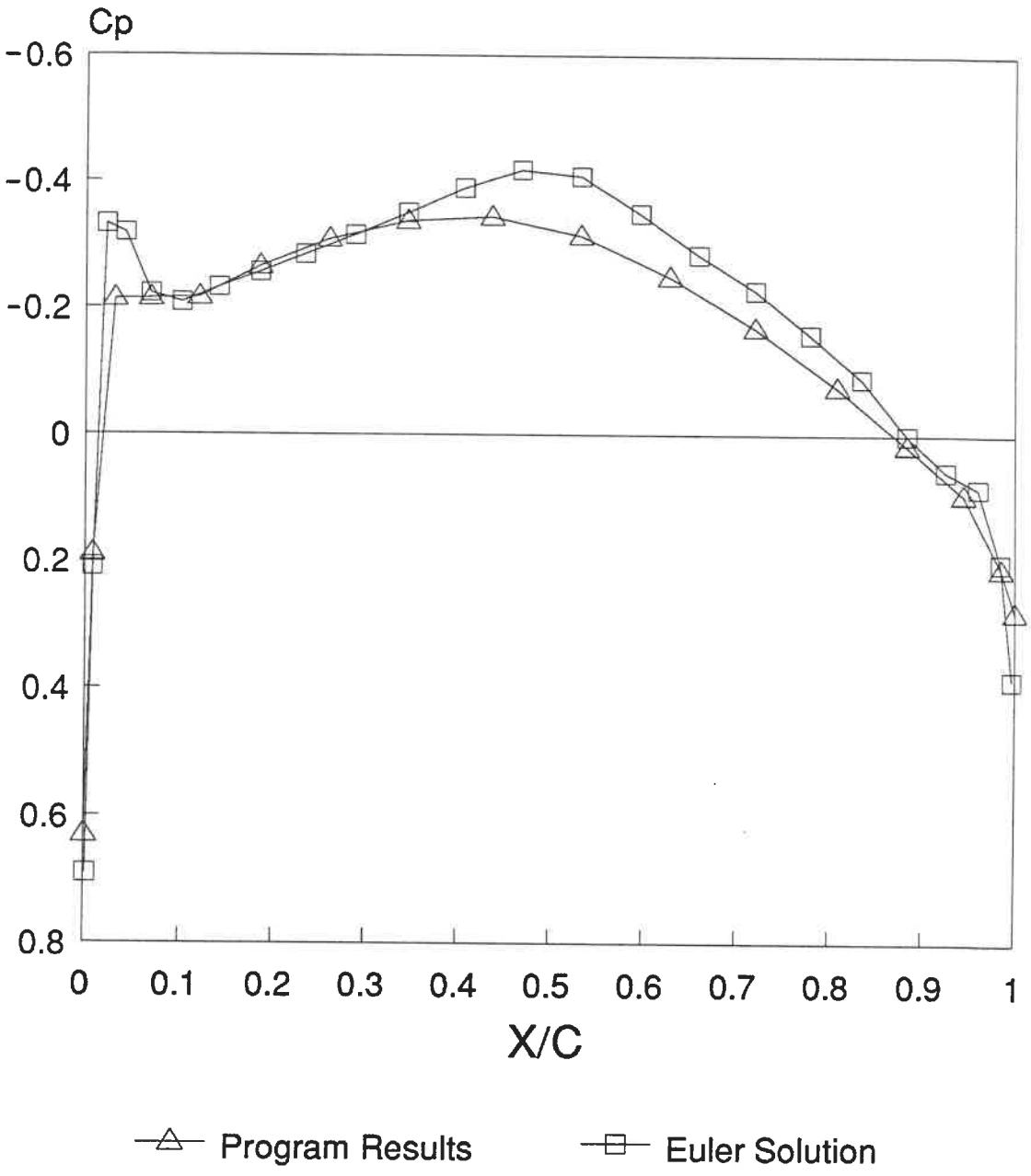


FIGURE 21: Pressure distribution on the ONERA M6 wing at 20% semispan station for M_∞=0.84 and 0.0° incidence

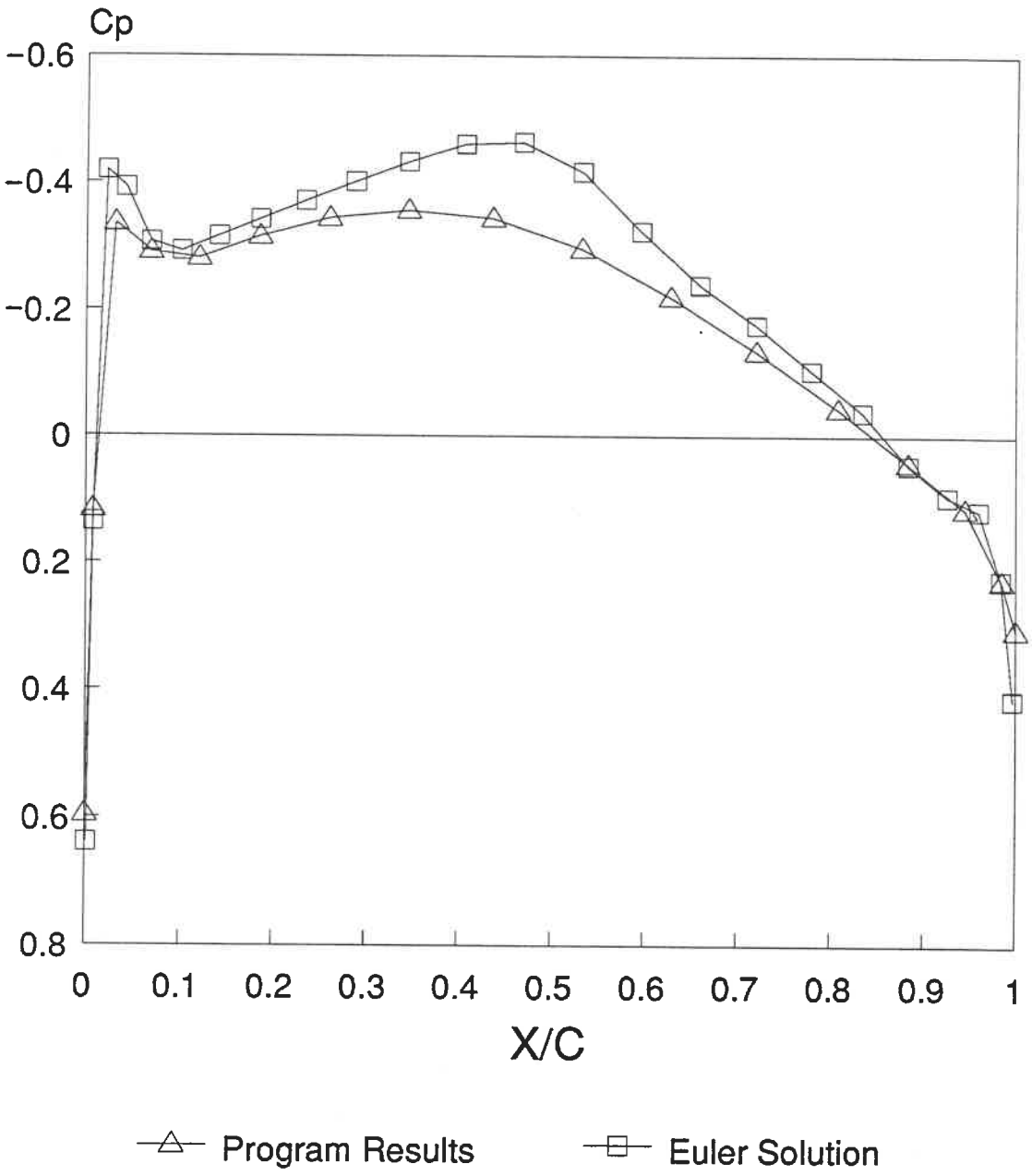


FIGURE 22: Pressure distribution on the ONERA M6 wing at 44% semispan station for M_∞=0.84 and 0.0° incidence

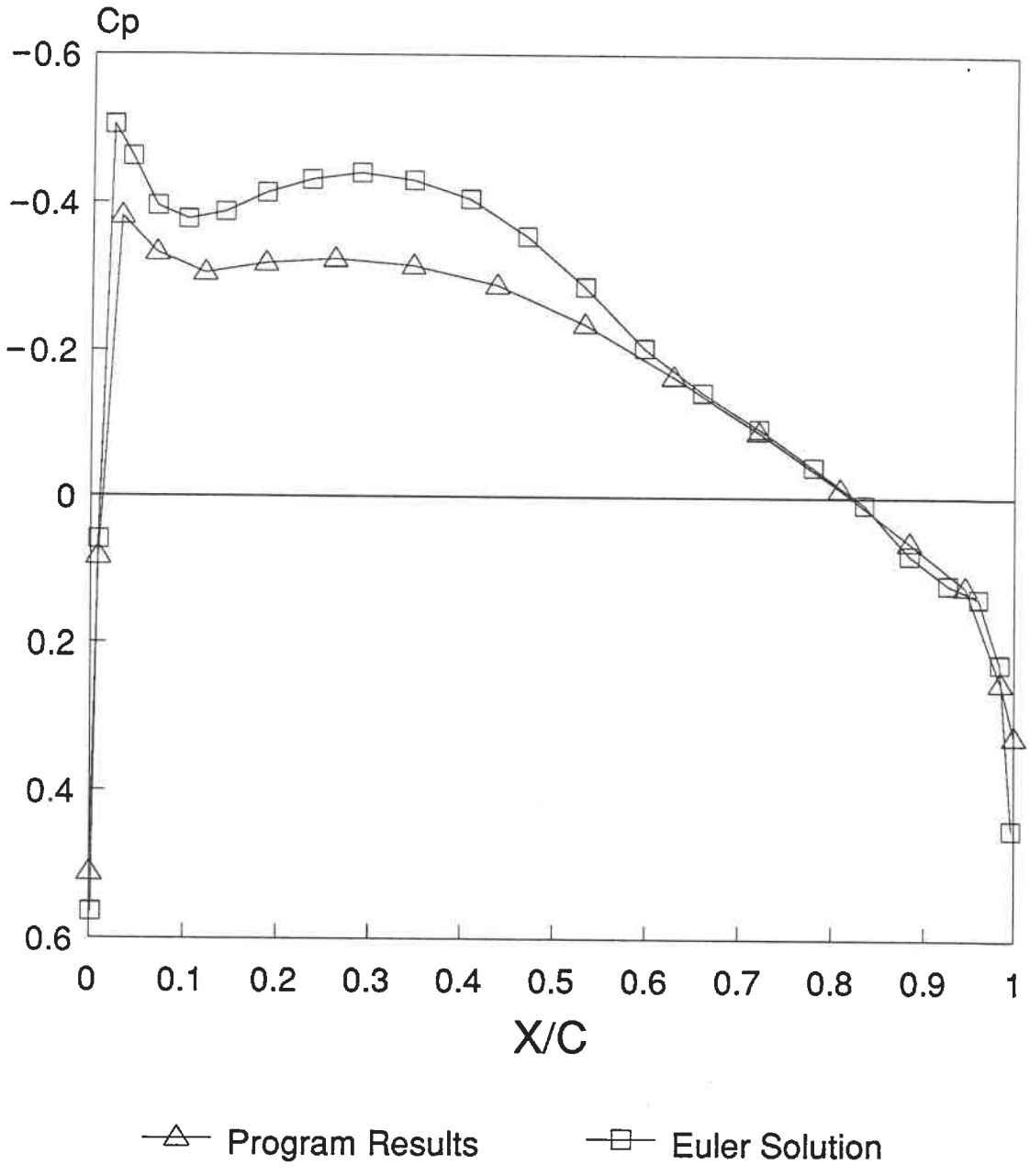


FIGURE 23: Pressure distribution on the ONERA M6 wing at 80% semispan station for M_∞=0.84 and 0.0° incidence

APPENDICE 1

```
      PROGRAM RIDOUB
C ***********************************************************************
C TRANSONIC FLOW PROGRAM USING PANEL & FIELD SOURCES INTEGRAL METHOD
C ONLY ONE SIDE OF 3-D WING IS REQUIRED IN INPUT FILE
C ***********************************************************************
C
C IMPORTANT PARAMETERS & VARIABLES OF THE PROGRAM
C
C PARAMETERS
C
C NH=NUMBER OF POINTS SPANWISE
C NV=NUMBER OF POINTS AROUND ONE SECTION (MUST BE AN ODD NUMBER)
C NO=NUMBER OF VOLUMES IN ONE DIRECTION AROUND HALF-WING
C NSHO=MAXIMUM NUMBER OF LOWER/AND UPPER SHOCK SURFACES(SET BY USER)
C NL=NH-1
C NNK=NUMBER OF POINTS ON WING SURFACE
C NN1=TOTAL NUMBER OF POINTS ON WING (INCLUDES TIP POINTS)
C NN=NUMBER OF PANELS ON WING (INCLUDING TIP)
C MX,MY,MZ=NUMBER OF FIELD POINTS IN X,Y & Z DIRECTIONS
C MZ2=NUMBER OF FIELD POINTS,Z DIRECTION, OVER & UNDER WING
C NGLOB=TOTAL NUMBER OF FIELD POINTS
C NELEM=TOTAL NUMBER OF FIELD ELEMENTS
C NDPL=NUMBER OF MEETING VOLUMES FOR A FIELD POINT (8)
C NPNO=NUMBER OF NODES PER WING PANEL (3)
C NWNO=NUMBER OF NODES PER WAKE PANEL (4)
C NWUNO=NUMBER OF COORDINATES PER WAKE PANELS (2 SINCE UPPER & LOWER
C     TRAILING EDGE POINTS HAVE THE SAME COORDINATE)
C NBQ=NUMBER OF MEETING PANELS AT A WING POINT
C
C VARIABLES
C
C X(NN1),Y(NN1),Z(NN1)=ARRAY OF WING POINTS IN GENERAL COORDINATES
C X2(NN1),Y2(NN1),Z2(NN1)=PARALLEL WING GRID (POTENTIAL CALCULATIONS)
C NODEB(NN,NPNO)=ARRAY OF PANELS AND CORRESPONDING NODES
C PI=PI CONSTANT->3.141516...
C XR(NN,NPNO),YR(NN,NPNO),ZR(NN,NPNO)=ARRAY OF THE RELATIVE COORDINATES
C     OF THE 3 NODES OF EACH PANEL
C COLX(NN),COMX(NN),CONX(NN)=COSINES OF GENERAL AXIS X,Y & Z WITH
C     RELATIVE AXIS XR FOR EACH PANEL
C COLY(NN),COMY(NN),CONY(NN)=COSINES OF GENERAL AXIS X,Y & Z WITH
C     RELATIVE AXIS YR FOR EACH PANEL
C COLZ(NN),COMZ(NN),CONZ(NN)=COSINES OF GENERAL AXIS X,Y & Z WITH
C     RELATIVE AXIS ZR FOR EACH PANEL
C XT(3),YT(3),ZT(3)=ARRAYS USED TO CALCULATE PANEL RELATIVE COORDINATES
C XP,YP,ZP=RELATIVE COORDINATES OF POINT FROM PANEL
C F1,F2,F3=RESULTS FROM "FI" S/R (EFFECT ON POINT FROM PANEL)
C STIF(4)=ARRAY OF POTENTIAL COEFFICIENTS FOR THE EFFECT OF 1 PANEL
C ALPHA,BETA,GAMA=FREE STREAM FLOW ANGLES IN XZ,XY & YZ PLANES
C A(NN1,NN1)=MATRIX OF COEFFICIENTS, EFFECT OF PANELS ON WING POINTS
C RHS1(NN1)=RIGHT-HAND SIDE VECTOR
```

```
C BA1(NN1,NN1)=COPY OF "A" MATRIX
C BAR(NN1)=COPY OF "RHS1"
C Q(NN1)=RESULTING POTENTIALS OF WING NODES
C SMALL=TEST VALUE FOR SOME CALCULATIONS
C XPW,YPW,ZPW=RELATIVE COORDINATES OF POINT FROM WAKE PANEL
C F3W=RESULT OF "FI3W" S/R (EFFECT ON POINT FROM WAKE PANEL)
C NODEW(NL,NWNO)=ARRAY OF WAKE PANELS AND CORRESPONDING NODES
C QX(NN),QY(NN),QZ(NN)=DERIVATIVES OF POTENTIALS ON PANEL SURFACES
C IB(NL),IBU(NL),ISBU(NN1),IBB(NN1)=VECTORS USED TO SEPARATE TRAILING
C       EDGE POINTS FROM OTHER WING POINTS
C XW(NL,NWUNO),YW(NL,NWUNO),ZW(NL,NWUNO)=ARRAY OF THE RELATIVE COORD-
C       NATES OF THE POINTS OF EACH WAKE PANELS
C CWLX(NL),CWMX(NL),CWNX(NL)=COSINES OF GENERAL AXIS X,Y & Z WITH
C       WAKE PANEL RELATIVE AXIS XR
C CWLY(NL),CWMY(NL),CWNY(NL)=COSINES OF GENERAL AXIS X,Y & Z WITH
C       WAKE PANEL RELATIVE AXIS YR
C CWLZ(NL),CWMZ(NL),CWNZ(NL)=COSINES OF GENERAL AXIS X,Y & Z WITH
C       WAKE PANEL RELATIVE AXIS ZR
C XIE(NN),YIE(NN),ZIE(NN)=ARRAYS OF GENERAL COORDINATES OF PANELS CENTER
C CP(NN)=PRESSURE COEFFICIENTS OF WING PANELS
C CPN(NN1)=PRESSURE COEFFICIENTS OF WING POINTS
C VC(NN)=ABSOLUTE RESULTING SPEED ON EACH PANEL
C DML(NN)=RESULTING MACH NUMBER ON EACH PANEL
C PHYG(NGLOB)=POTENTIAL OF FIELD POINTS
C DNLOC1(8,3,8),DNGLO1(8,3),DJACO1(3,3),DETER1=VECTORS USED IN THE
C       FINITE ELEMENT DISCRETIZATION, SIGMA CALCULATION
C NODE(NELEM,NDPL)=ARRAY OF FIELD ELEMENTS WITH CORRESPONDING NODES(8)
C PSXE(NELEM),PSYE(NELEM),PSZE(NELEM)=DERIVATIVES OF POTENTIAL FOR EACH
C       FIELD ELEMENT IN X, Y & Z DIRECTION
C PSY(NGLOB,NDPL),PSX(NGLOB,NDPL),PSZ(NGLOB,NDPL)=DERIVATIVES OF POTEN-
C       TIAL FOR FIELD NODES (1 PER MEETING ELEMENT) IN Y,X & Z DIR.
C X3(NGLOB),Y3(NGLOB),Z3(NGLOB)=GENERAL COORDINATES OF FIELD POINTS
C X3G(NELEM),Y3G(NELEM),Z3G(NELEM)=GENERAL COORDINATES OF FIELD
C       ELEMENT CENTERS
C V3G(NELEM)=VOLUME OF FIELD ELEMENTS
C XAL(MZ2),YAL(MY),YAT(MY),ZAU(MZ2)=VECTORS USED IN FIELD POINTS
C       COORDINATES CALCULATIONS
C NGL(NGLOB),NGLT(NN1)=VECTORS THAT ASSOCIATES MEETING FIELD AND WING
C       POINTS (BODY FITTING & POTENTIAL CALCULATION)
C NWAKE(NGLOB)=VECTOR USED TO EXCLUDE WAKE POINTS FROM POTENTIALCALC.
C SIGMA(NELEM)=VECTOR OF SOURCE TERMS FOR EACH FIELD VOLUMES
C RESIDU(NN1)=VECTOR OF RESIDUES FOR EACH WING POINT
C PSXX(NGLOB),PSXY(NGLOB),PSXZ(NGLOB)=SECOND DERIVATIVES OF POTENTIAL,
C       FIELD POINTS, XX,XY & XZ DERIVATIVES
C PSYX(NGLOB),PSYY(NGLOB),PSYZ(NGLOB)=SECOND DERIVATIVES OF POTENTIAL,
C       FIELD POINTS, YX,YY & YZ DERIVATIVES
C PSZX(NGLOB),PSZY(NGLOB),PSZZ(NGLOB)=SECOND DERIVATIVES OF POTENTIAL,
C       FIELD POINTS, ZX,ZY & ZZ DERIVATIVES
C PSXXE(8),PSXYE(8),PSXZE(8)=SECOND DERIVATIVES OF POTENTIAL,
C       FIELD ELEMENTS, XX,XY & XZ DERIVATIVES (S/R "UV")
C PSYXE(8),PSYYE(8),PSYZE(8)=SECOND DERIVATIVES OF POTENTIAL,
```

```
C       FIELD ELEMENTS, YX,YY & YZ DERIVATIVES (S/R "UV")
C PSZXE(8),PSZYE(8),PSZZE(8)=SECOND DERIVATIVES OF POTENTIAL,
C       FIELD ELEMENTS, ZX,ZY & ZZ DERIVATIVES (S/R "UV")
C DM0=VARIABLE F(DMM) USED TO CALCULATE ELEMENT & PANEL MACH NUMBER
C DMM=FREE STREAM FLOW MACH NUMBER
C DMG(NELEM)=MACH NUMBER AT THE CENTER OF FIELD ELEMENTS
C VV2(NELEM)=SQUARE VALUE OF FIELD ELEMENT SPEED
C STIFB(NPNO)=CALCULATED COEFFICIENTS OF POTENTIAL IN "PHYGLOB" S/R
C           EFFECT OF WING PANEL ON A FIELD POINT
C STIFW(NWNO)=CALCULATED COEFFICIENTS OF POTENTIAL IN "PHYGLOB" S/R
C           EFFECT OF WAKE PANEL ON A FIELD POINT
C KSU=NUMEROTATION OF SHOCK SURFACES, ELEMENTS OVER WING
C KSL=NUMEROTATION OF SHOCK SURFACES, ELEMENTS UNDER WING
C KSUE(NSHO)=VECTOR THAT ASSOCIATES SHOCK NUMBER WITH ELEMENT NUMBER
C           FOR OVER THE WING ELEMENTS
C KSLE(NSHO)=VECTOR THAT ASSOCIATES SHOCK NUMBER WITH ELEMENT NUMBER
C           FOR UNDER THE WING ELEMENTS
C KE(NELEM)=VECTOR NOTING THE NUMBER OF THE ELEMENT FOLLOWING THE
C           SHOCK SURFACE
C KN(NGLOB)=VECTOR THAT SPECIFIES WHETHER THE FIELD POINT IS PART OF A
C           SHOCK SURFACE OR NOT
C XCU(NSHO),YCU(NSHO),ZCU(NSHO)=GENERAL COORDINATES OF SHOCK SURFACES
C           CENTER, OVER THE WING SHOCKS ("UPPER" SCHOCKS)
C VSAU(NSHO)=SURFACE OF UPPER SHOCKS
C XCL(NSHO),YCL(NSHO),ZCL(NSHO)=GENERAL COORDINATES OF SHOCK SURFACES
C           CENTER, UNDER THE WING SHOCKS ("LOWER" SCHOCKS)
C VSAL(NSHO)=SURFACE OF LOWER SHOCKS
C U1(NGLOB),V1(NGLOB),W1(NGLOB)=SPEED COMPONENTS IN X,Y & Z DIRECTIONS
C           FOR FIELD POINTS
C DG0,DG1=FIELD GRID SIZE PARAMETERS, DG0 SET BY USER
C YMAX=HIGHEST WING Y COORDINATE AFTER NORMALIZATION
C YHSPAN=HIGHEST WING Y COORDINATE BEFORE NORMALIZATION (WING HALF-
SPAN)
C FRAC=FRACTION OF NORMALIZED HALF SPAN, USED TO DETERMINE WHETHER OR
C           NOT TO USE SIMPLIFIED VERSION OF PANEL EFFECT ON POINT
C SURFW=NORMALIZED HALF-WING REFERENCE SURFACE (USED IN CL
CALCULATION)
C NEB(NN1,NBQ)=ARRAY OF PANELS ASSOCIATED TO EACH WING POINT FOR POINT
C           CP CALCULATION ("CPN(NN1)")
C NMNM=FLAG USED WHEN WING PANEL-POINT RE-NUMEROTATION IS REQUIRED
C ITMAX=MAXIMUM NUMBER OF ITERATIONS (SET BY USER)
C TEST=COMPARISON VALUE FOR RESIDUE CONVERGEANCE (SET BY USER)
C
C ***************************************************************************
C ORIGINALLY WRITTEN BY DR.ZHIGANG FANG, ECOLE POLYTECHNIQUE, AUG.1990
C
C COMMENTED & MODIFIED BY RICHARD CLAIROUX, M.ING STUDENT
C           ECOLE POLYTECHNIQUE DE MONTREAL, SEPTEMBER 1991
C ***************************************************************************

      IMPLICIT REAL*8 (A-H,O-Z)
```

```
 PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
 PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)
 COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
 COMMON/NNNN/NODEB(NN,NPNO)
 COMMON/PI11/PI
 COMMON/XYZR1/XR(NN,NPNO),YR(NN,NPNO),ZR(NN,NPNO)
 COMMON/COLMN1/COLX(NN),COMX(NN),CONX(NN)
 COMMON/COLMN2/COLY(NN),COMY(NN),CONY(NN)
 COMMON/COLMN3/COLZ(NN),COMZ(NN),CONZ(NN)
 COMMON/XYZT1/XT(3),YT(3),ZT(3)
 COMMON/XYZ21R/XP,YP,ZP
 COMMON/F123/F1,F2,F3
 COMMON/ST11/STIF(4)
 COMMON/ABC11/ALPHA,BETA,GAMA
 COMMON/BKARHS/A(NN1,NN1),RHS1(NN1),BA1(NN1,NN1),BAR(NN1)
 COMMON/QQ1/Q(NN1)
 COMMON/SAML/SMALL
 COMMON/XYZ21W/XPW,YPW,ZPW
 COMMON/F12W/F3W
 COMMON/NNNW/NODEW(NL,NWNO)
 COMMON/QXYZ1/QX(NN),QY(NN),QZ(NN)
 COMMON/IBN/IB(NL),IBU(NL),ISBU(NN1),IBB(NN1)
 COMMON/XYZRW/XW(NL,NWUNO),YW(NL,NWUNO),ZW(NL,NWUNO)
 COMMON/CWLMN1/CWLX(NL),CWMX(NL),CWNX(NL)
 COMMON/CWLMN2/CWLY(NL),CWMY(NL),CWNY(NL)
 COMMON/CWLMN3/CWLZ(NL),CWMZ(NL),CWNZ(NL)
 COMMON/XYZIE/XIE(NN),YIE(NN),ZIE(NN)
 COMMON/VCP1/CP(NN),CPN(NN1),VC(NN),DML(NN)
 COMMON/PG1/PHYG(NGLOB)
 COMMON/DN11/DNLOC1(8,3,8)
 COMMON/DNG1/DNGLO1(8,3)
 COMMON/DJ1/DJACO1(3,3)
 COMMON/DE1/DETER1
 COMMON/NNODE1/NODE(NELEM,NDPL)
 COMMON/CPSIUV/PSXE(NELEM),PSYE(NELEM),PSZE(NELEM)
 COMMON/PXYSS/PSY(NGLOB,NDPL),PSX(NGLOB,NDPL),PSZ(NGLOB,NDPL)
 COMMON/XYZG/X3(NGLOB),Y3(NGLOB),Z3(NGLOB)
 COMMON/XYZGIE/X3G(NELEM),Y3G(NELEM),Z3G(NELEM),V3G(NELEM)
 COMMON/XYZA1/XAL(MZ2),YAL(MY),YAT(MY),ZAU(MZ2)
 COMMON/NGL1/NGL(NGLOB),NGLT(NN1),NWAKE(NGLOB)
 COMMON/PS111/SIGMA(NELEM)
 COMMON/RES11/RESIDU(NN1)
 COMMON/XYZSS4/PSXXE(NELEM),PSXYE(NELEM),PSXZE(NELEM)
 COMMON/XYZSS5/PSYXE(NELEM),PSYYE(NELEM),PSYZE(NELEM)
 COMMON/XYZSS6/PSZXE(NELEM),PSZYE(NELEM),PSZZE(NELEM)
 COMMON/DM010/DM0,DMM
 COMMON/DMGVV/DMG(NELEM),VV2(NELEM)
 COMMON/STGB/STIFB(NPNO)
```

```
      COMMON/STGW/STIFW(NWNO)
      COMMON/KE123/KSU,KSL,KSUE(NSHO),KSLE(NSHO),KE(NELEM),KN(NGLOB)
      COMMON/SHOPUP/XCU(NSHO),YCU(NSHO),ZCU(NSHO),VSAU(NSHO)
      COMMON/SHOPLO/XCL(NSHO),YCL(NSHO),ZCL(NSHO),VSAL(NSHO)
      COMMON/UVW11/U1(NGLOB),V1(NGLOB),W1(NGLOB)
      COMMON/DG10/DG0,DG1
      COMMON/YMAX11/YMAX,YHSPAN,FRAC,SURFW
      COMMON/NE1B/NEB(NN1,NBQ)
      COMMON/NMNM1/NMNM

C -------------------------------------------------------------
C SET IMPORTANT PARAMETERS
      DG0=0.10D00
      DMM=0.84D00
      ITMAX=15
      TEST=0.00001D00
      FRAC=1.2D00
      ALPHA=0.00D00
      BETA=90.0D00
      GAMA=90.0D00 -ALPHA

C -------------------------------------------------------------
C PRINT TITLE AND FIXED PARAMETERS
      WRITE(6,*)'TEST 33PTS MACH 0.84 R=1.20D00'
      WRITE(6,*)' FREE STREAM MACH NUMBER=',DMM
      WRITE(6,*)' NH =',NH
      WRITE(6,*)' NV =',NV
      WRITE(6,*)' NO =',NO
      WRITE(6,*)' NN =',NN
      WRITE(6,*)' NN1=',NN1
      WRITE(6,*)' MX =',MX
      WRITE(6,*)' MY =',MY
      WRITE(6,*)' MZ =',MZ
      WRITE(6,*)' NELEM=',NELEM
      WRITE(6,*)' NGLOB=',NGLOB
      WRITE(6,*)' ALPHA=',ALPHA
      WRITE(6,*)' BETA =',BETA
      WRITE(6,*)' GAMA =',GAMA
      WRITE(6,*)' DG0=',DG0
      WRITE(6,*)' ITMAX=',ITMAX
      WRITE(6,*)' TEST=',TEST
      WRITE(6,*)' FRAC =',FRAC

C -------------------------------------------------------------
C TRANSFORM ANGLES IN RADIANS AND EVALUATE OTHER CONSTANTS
      PI=2.0D00*DATAN2(1.0D00,0.0D00)
      ALPHA=ALPHA/180.0D00*PI
      BETA=BETA/180.0D00*PI
      GAMA=GAMA/180.0D00*PI
      DG1=DG0/2.0D00
      DM0=DMM*DMM/(1.0D00+0.2D00*DMM*DMM)
```

```
      SMALL=0.0000001D00

C -------------------------------------------------------------
C READ POINTS AND CREATE PANELS (S/R XYZL); CREATE WAKE PANELS (S/R
C BNODE); CALCULATE RELATIVE COORDS OF WAKE PANELS (S/R XYZTRANW);
C CALCULATE PARALLEL GRID FOR WING POINTS (S/R GRIDB)
      CALL XYZL
      CALL BNODE
      CALL XYZTRANW
      CALL GRIDB

C -------------------------------------------------------------
C CALCULATE PANEL RELATIVE COORDS(S/R XYZTRANS).IF A PANEL RE-NUMEROTA-
C TION IS REQUIRED (NMNM.GT.0.0), CALL XYZTRANS AGAIN. ONLY ONE RE-
C NUMEROTATION ALLOWED OR (KKK.GT.1) QUIT PROGRAM AND CHECK INPUT FILE
      KKK=0
 3000 WRITE(6,*)' '
      KKK=KKK+1
      NMNM=0
      CALL XYZTRANS
      IF(NMNM.GT.0)THEN
      WRITE(6,*)' PANEL NODES NUMEROTATION (1,2,3) WAS MODIFIED'
      IF(KKK.GT.1)THEN
      WRITE(6,*)' SECOND RE-NUMEROTATION REQUIRED=CHECK INPUT FILE'
      WRITE(6,*)'******** PROGRAM IS STOPPED ********'
      GO TO 9999
      END IF
      GO TO 3000
      END IF

C -------------------------------------------------------------
C CALCULATE FIELD GRID AND ELEMENTS (S/R XYZGN); BUILD MATRIX OF COEF-
C FICIENTS "A" (S/R BUILDA); APPLY KUTTA CONDITION AT THE T.E.(S/R BC)
      CALL XYZGN
      CALL BUILDA
      CALL BC

C -------------------------------------------------------------
C NOTE ARRAYS "A" (BAR) AND "RHS1" (BA1) BEFORE THEY ARE USED IN MATRIX
C INVERSION: "A" BECOMES THE INVERSE (A)-1 OF A AND "RHS1" BECOMES THE
C SOLUTION VECTOR
      DO 671 I=1,NN1
      BAR(I)=RHS1(I)
      DO 611 J=1,NN1
      BA1(I,J)=A(I,J)
  611 CONTINUE
  671 CONTINUE

C -------------------------------------------------------------
C INVERSE "A" (S/R LUDC) AND GET THE SOLUTION VECTOR (S/R LUBS). ASSO-
C CIATE THE INCOMPRESSIBLE SOLUTIONS WITH POTENTIALS "Q"
```

```
      CALL LUDC
      CALL LUBS
      DO 53 I=1,NN1
      Q(I)=RHS1(I)
   53  CONTINUE

C -------------------------------------------------------
C ASSOCIATE FIELD POINTS POTENTIAL WITH WING POTENTIALS FOR MEETING
C POINTS (NGL(I).NE.0.0). INITIALIZE KSU,KSL & SIGMA VECTORS (MACH=0.0)
C CALCULATE INITIAL FIELD NODES POTENTIALS (S/R PHYGLOB)
      DO 900 I=1,NGLOB
      IF(NGL(I).NE.0) PHYG(I)=Q(NGL(I))
  900 CONTINUE

      KSU=0
      KSL=0
      DO 468 I=1,NELEM
      SIGMA(I)=0.0D00
  468 CONTINUE

      CALL PHYGLOB

C -------------------------------------------------------------
C BEGINNING OF ITERATIVE LOOP 9001. IF MACH=0.0, GO STRAIGHT TO OUTPUT.
C # OF ITERATIONS LIMITED BY "ITMAX"
      IF(DMM.LT.0.001D00) GOTO 805
      DO 9001 KOUNT=1,ITMAX

C -------------------------------------------------------------
C CALCULATE SIGMA VALUES OF FIELD ELEMENTS (S/R UV). RIGHT HAND SIDE
C VECTOR TAKES INCOMPRESSIBLE VALUES (NOTED IN "BAR")

      CALL UV

      DO 901 I=1,NN1
      RHS1(I)=BAR(I)
  901 CONTINUE

C -------------------------------------------------------------
C FOR EACH WING NODE, CALCULATE NEW RIGHT SIDE TERM (LOOP 902)
C PARALLEL GRID NODE COORDS (X2,Y2,Z2) ARE USED.
C REMOVE T.E. POINTS FROM CALCULATION (IBB.EQ.I). FIRST, CALCULATE
C EFFECT OF FIELD ELEMENTS (LOOP 689), STORED IN "SUMP1". REPEAT
C CALCULATIONS FOR OTHER SIDE OF WING ELEMENTS (NEG. Y3G)
      DO 902 I=1,NN1
      IF(IBB(I).EQ.I)GO TO 902

      SUMP1=0.0D00
      DO 689 IE=1,NELEM
      XRS=X2(I)-X3G(IE)
      YRS=Y2(I)-Y3G(IE)
```

```
      ZRS=Z2(I)-Z3G(IE)
      SUMP1=SUMP1-SIGMA(IE)*V3G(IE)/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
      YRS=Y2(I)+Y3G(IE)
      SUMP1=SUMP1-SIGMA(IE)*V3G(IE)/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
  689 CONTINUE

C -------------------------------------------------------------
C NEXT, CALCULATE EFFECT OF UPPER SHOCKS (LOOP 681, STORED IN SUMU1)
C AND LOWER SHOCKS (LOOP 682, STORED IN SUML1).
C AGAIN, REPEAT CALCULATIONS FOR OTHER SIDE OF WING SHOCKS (NEG. YCU,YCL
C FINALLY, EVALUATE NEW RIGHT HAND SIDE TERM RHS1(I)
      SUMU1=0.0D00
      DO 681 K=1,KSU
      IE=KSUE(K)
      XRS=X2(I)-XCU(K)
      YRS=Y2(I)-YCU(K)
      ZRS=Z2(I)-ZCU(K)
      SUMU1=SUMU1+(PSXE(IE)-PSXE(IE+MY-1))*VSAU(K)
     1/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
      YRS=Y2(I)+YCU(K)
      SUMU1=SUMU1+(PSXE(IE)-PSXE(IE+MY-1))*VSAU(K)
     1/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
  681 CONTINUE

      SUML1=0.0D00
      DO 682 K=1,KSL
      IE=KSLE(K)
      XRS=X2(I)-XCL(K)
      YRS=Y2(I)-YCL(K)
      ZRS=Z2(I)-ZCL(K)
      SUML1=SUML1+(PSXE(IE)-PSXE(IE+MY-1))*VSAL(K)
     1/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
      YRS=Y2(I)+YCL(K)
      SUML1=SUML1+(PSXE(IE)-PSXE(IE+MY-1))*VSAL(K)
     1/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
  682 CONTINUE

      RHS1(I)=RHS1(I)+SUMP1+SUMU1+SUML1
  902 CONTINUE

C -------------------------------------------------------------
C CALCULATE (A)*(Q) TERM AND CALCULATE THE DIFFERENCE WITH THE NEW
C RIGHT HAND SIDE TERM= THE RESIDUE (HERE, "BA1" REPRESENTS THE "A"
C MATRIX SINCE "A(NN1,NN1)" IS ACTUALLY THE INVERSE OF "A")
      DO 930 I=1,NN1
      RESIDU(I)=0.0D00
      DO 931 J=1,NN1
      RESIDU(I)=RESIDU(I)+BA1(I,J)*Q(J)
  931 CONTINUE
      RESIDU(I)=RESIDU(I)-RHS1(I)
  930 CONTINUE
```

```
C ---------------------------------------------------------------
C COMPARE THE SQUARE OF THE RESIDUES WITH THE "TEST" VALUE. GO TO
C OUTPUT IF THE SUM IS UNDER "TEST"
      SUM3=0.0D00
      DO 939 I=1,NN1
      SUM3=SUM3+RESIDU(I)*RESIDU(I)
  939 CONTINUE
      WRITE(6,*)'ITERATION # ',KOUNT,' -->  RESIDU=',SUM3
      IF(SUM3.LT.TEST)GO TO 805

C ---------------------------------------------------------------
C CALCULATE NEW WING NODES POTENTIALS WITH NEW "RHS1" (S/R LUBS)
C ASSOCIATE RESULTING VALUES IN RHS1 WITH "Q" VECTOR
      CALL LUBS

      DO 57 I=1,NN1
      Q(I)=RHS1(I)
   57  CONTINUE

C ---------------------------------------------------------------
C ASSOCIATE FIELD POINTS POTENTIAL WITH WING POTENTIALS FOR MEETING
C POINTS (NGL(I).NE.0.0). CALCULATE NEW FIELD POINT POTENTIALS (S/R
C PHYGLOB)
      DO 950 I=1,NGLOB
      IF(NGL(I).NE.0)PHYG(I)=Q(NGL(I))
  950 CONTINUE

      CALL PHYGLOB

 9001 CONTINUE

  805 IF (KOUNT.GT.ITMAX) WRITE(6,*)'INSUFFICIENT NUMBER OF ITERATIONS
     + FOR SPECIFIED CONVERGEANCE VALUE'

C ---------------------------------------------------------------
C                    O U T P U T
C ---------------------------------------------------------------
C OUTPUT WRITING. FIRST, PRINT WING NODES COORDINATES & POTENTIALS
C AROUND WING SECTIONS STARTING FROM TIP

      WRITE(6,*)'********* PRINT FINAL RESULTS *********'
      WRITE(6,*)' '
C     WRITE(6,*)' WING NODE #,  X,Y,Z COORDINATES,  POTENTIAL'
      WRITE(6,*)' WING NODE # AND POTENTIAL'
      DO 222 I=1,NH
C     WRITE(6,*)' '
      DO 322 J=1,NV
      II=I+(J-1)*NH
      IF(J.EQ.NV)II=I+(J-1)*NH-1
      IF(J.EQ.NV.AND.I.EQ.1)II=1
C     WRITE(6,55)II,X(II),Y(II),Z(II),Q(II)
```

```
      WRITE(6,*)II,Q(II)
  322 CONTINUE
  222 CONTINUE
   55  FORMAT(I6,4F16.8)

C -------------------------------------------------------------
C CALCULATE WING PANEL CP AND THEN WING NODES CP. PRINT CP RESULTS
C (S/R PHYBXYZ)
      CALL PHYBXYZ

C -------------------------------------------------------------
C PRINT SIGMA VALUES OF UPPER & LOWER WING PLANE ELEMENTS
C     WRITE(6,*)' WING PLANE ELEMENT #   , SIGMA VALUE'
      DO 6110 I=NO,NO+1
      DO 6112 K=NO,MY-1
C     WRITE(6,*)' '
      DO 6111 J=1,MX-1
      IE=K+(J-1)*(MY-1)+(I-1)*(MY-1)*(MX-1)
C     WRITE(6,6779)IE,SIGMA(IE)
 6111 CONTINUE
 6112 CONTINUE
 6110 CONTINUE

C -------------------------------------------------------------
C PRINT POTENTIAL OF UPPER & LOWER WING PLANE FIELD NODES

C     WRITE(6,*)'WING PLANE FIELD NODES#,   COORDS.,   POTENTIAL'
      DO 6310 I=NO+1,NO+2
      DO 6312 K=1,MY
C     WRITE(6,*)' '
      DO 6311 J=1,MX
      II=K+(J-1)*MY+(I-1)*MY*MX
      IF(NGL(II).NE.0) GOTO 6311
C     WRITE(6,6779)II,PHYG(II)
 6311 CONTINUE
 6312 CONTINUE
 6310 CONTINUE
 6779 FORMAT(I5,F16.8)

C -------------------------------------------------------
C PRINT ELEMENTS WITH SHOCK SURFACES & THEIR MACH NUMBERS

      WRITE(6,*) ' LISTING OF ELEMENTS WITH SHOCK SURFACES'
      WRITE(6,*)' '
      WRITE(6,*)'                UPPER SHOCKS'
      WRITE(6,*)'I,KSUE(I),DMG(KSUE(I)),KSUE(I)+MY-1,DMG(KSUE(I)+MY-1)'
      DO 8150 I=1,KSU
      WRITE(6,8151)I,KSUE(I),DMG(KSUE(I)),KSUE(I)+MY-1,DMG(KSUE(I)+MY-1)
 8150 CONTINUE
      WRITE(6,*)' '
      WRITE(6,*)'                LOWER SHOCKS'
```

```
      WRITE(6,*)'I,KSLE(I),DMG(KSLE(I)),KSLE(I)+MY-1,DMG(KSLE(I)+MY-1)'
      DO 8152 I=1,KSL
      WRITE(6,8151)I,KSLE(I),DMG(KSLE(I)),KSLE(I)+MY-1,DMG(KSLE(I)+MY-1)
 8152 CONTINUE
 8151 FORMAT(2I7,F12.6,I7,F12.6)

C -------------------------------------------------------
C PRINT INFORMATION OF FIELD ELEMENTS

C     WRITE(6,*)'LISTING OF FIELD ELEMENTS INFORMATION'
C     WRITE(6,*)'I,KE(I),SIGMA(I),DMG(I),PSXE(I),PSYE(I),PSZE(I)'
      DO 8119 I=1,NELEM
C      WRITE(6,7881)I,KE(I),SIGMA(I),DMG(I),PSXE(I),PSYE(I),PSZE(I)
 8119 CONTINUE

C -------------------------------------------------------
C PRINT FIELD NODES INFORMATION

C     WRITE(6,*)'LISTING OF FIELD NODES INFORMATION'
C     WRITE(6,*)'I,KN(I),PHYG(I),  PSX(I,1),  U1(I), PSY,V1,PSZ,W1'
      DO 8129 I=1,NGLOB
C      WRITE(6,7882)I,KN(I),PHYG(I),PSX(I,1),U1(I),PSY(I,1),V1(I)
C     1,PSZ(I,1),W1(I)
 8129 CONTINUE
 7881 FORMAT(2I6,5F12.6)
 7882 FORMAT(2I6,7F12.6)

C -------------------------------------------------------
C END, MAIN PROGRAM

 9999 CONTINUE
      STOP
      END
C=============================================================
      SUBROUTINE XYZL
C THIS S/R READS THE WING POINTS, EVALUATES THE CORRESPONDING PANELS
C AND DETERMINES THE PANELS ASSOCIATED TO THE NODES FOR CP CALCULATION
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)
      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/PI11/PI
      COMMON/XYZR1/XR(NN,NPNO),YR(NN,NPNO),ZR(NN,NPNO)
      COMMON/COLMN1/COLX(NN),COMX(NN),CONX(NN)
      COMMON/COLMN2/COLY(NN),COMY(NN),CONY(NN)
      COMMON/COLMN3/COLZ(NN),COMZ(NN),CONZ(NN)
      COMMON/XYZT1/XT(3),YT(3),ZT(3)
```

```
      COMMON/XYZ21R/XP,YP,ZP
      COMMON/F123/F1,F2,F3
      COMMON/ST11/STIF(4)
      COMMON/ABC11/ALPHA,BETA,GAMA
      COMMON/BKARHS/A(NN1,NN1),RHS1(NN1),BA1(NN1,NN1),BAR(NN1)
      COMMON/QQ1/Q(NN1)
      COMMON/SAML/SMALL
      COMMON/XYZ21W/XPW,YPW,ZPW
      COMMON/F12W/F3W
      COMMON/NNNW/NODEW(NL,NWNO)
      COMMON/XYZIE/XIE(NN),YIE(NN),ZIE(NN)
      COMMON/QXYZ1/QX(NN),QY(NN),QZ(NN)
      COMMON/IBN/IB(NL),IBU(NL),ISBU(NN1),IBB(NN1)
      COMMON/XYZRW/XW(NL,NWUNO),YW(NL,NWUNO),ZW(NL,NWUNO)
      COMMON/CWLMN1/CWLX(NL),CWMX(NL),CWNX(NL)
      COMMON/CWLMN2/CWLY(NL),CWMY(NL),CWNY(NL)
      COMMON/CWLMN3/CWLZ(NL),CWMZ(NL),CWNZ(NL)
      COMMON/YMAX11/YMAX,YHSPAN,FRAC,SURFW
      COMMON/NE1B/NEB(NN1,NBQ)
C -------------------------------------------------------------
C READ WING NODES FROM INPUT FILE FOLLOWING SPECIFIED FORMAT
      DO 1246 I=1,NNK
      READ(5,*)X(I),Y(I),Z(I)
 1246 CONTINUE

C -------------------------------------------------------------
C NORMALIZE WING COORDINATES WITH HALF-SPAN (YHSPAN)
      YHSPAN=DABS(Y(1))
      IMAX=1
      DO 3246 I=1,NNK
      IF(DABS(Y(I)).GT.YHSPAN)THEN
      IMAX=I
      YHSPAN=DABS(Y(I))
      END IF
 3246 CONTINUE
      DO 560 I=1,NNK
      X(I)=X(I)/YHSPAN
      Y(I)=Y(I)/YHSPAN
      Z(I)=Z(I)/YHSPAN
  560 CONTINUE
      YMAX=DABS(Y(IMAX))
      WRITE(6,*)'WING NORMALIZED: HALFSPAN CHANGED FROM ',YHSPAN,' TO',
     +YMAX
C -------------------------------------------------------------
C CALCULATE COORDINATES OF NEW TIP NODES (AVERAGE OF UPPER & LOWER
NODES
      DO 568 I=1,(NV-1)/2-1
      I1=NNK+1+(I-1)
      IH=NH*(NV-1)/2+1-NH*I
      IL=NH*(NV-1)/2+1+NH*I
```

```
      X(IH)=(X(IH)+X(IL))/2.0D00
      X(IL)=X(IH)
      DX=(X(IH)-X(IL))/2.0D00
      DY=(Y(IH)-Y(IL))/2.0D00
      DZ=(Z(IH)-Z(IL))/2.0D00
      X(I1)=X(IL)+DX
      Y(I1)=Y(IL)+DY
      Z(I1)=Z(IL)+DZ
  568 CONTINUE
C -------------------------------------------------------------
C CALCULATE PANEL-NODES ASSOCIATIONS, WING SURFACE
C LOOP 5246 FOR UPPER, LOOP 246 FOR LOWER (SPECIAL CASES FOR T.E.)
      DO 247 I=1,NL
      DO 5246 J=1,(NV-1)/2
      IE=2*I-1+(J-1)*NL*2
      IE1=2*I+(J-1)*NL*2
      NODEB(IE,1)=IE+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE,2)=IE+1+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE,3)=IE+1+NH+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE1,1)=IE1-1+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE1,2)=IE1+NH+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE1,3)=IE1-1+NH+(J-1)*NH-(I-1)-(J-1)*2*NL
 5246 CONTINUE
      DO 246 J=(NV-1)/2+1,NV-1
      IE=2*I-1+(J-1)*NL*2
      IE1=2*I+(J-1)*NL*2
      NODEB(IE,1)=IE+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE,2)=IE+1+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE,3)=IE+NH+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE1,1)=IE1+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE1,2)=IE1+NH+(J-1)*NH-(I-1)-(J-1)*2*NL
      NODEB(IE1,3)=IE1-1+NH+(J-1)*NH-(I-1)-(J-1)*2*NL
      IF(J.EQ.NV-1)THEN
      NODEB(IE,3)=NODEB(IE,3)-1
      NODEB(IE1,2)=NODEB(IE1,2)-1
      NODEB(IE1,3)=NODEB(IE1,3)-1
      IF(I.EQ.1)THEN
      NODEB(IE,3)=1
      NODEB(IE1,3)=1
      END IF
      END IF
  246 CONTINUE
  247 CONTINUE

C -------------------------------------------------------------
C PANEL-NODE ASSOCIATIONS FOR TIP GRID. FIRST, 2 PANELS NEAR L.E.
      I=1
      IE=IE1+I
      NODEB(IE,1)=NH*(NV-1)/2+1
      NODEB(IE,2)=NODEB(IE,1)+NH
      NODEB(IE,3)=NNK+1
```

```
      I=I+1
      IE=IE1+I
      NODEB(IE,1)=NH*(NV-1)/2+1
      NODEB(IE,2)=NNK+1
      NODEB(IE,3)=NODEB(IE,1)-NH
C -------------------------------------------------------------
C SECOND, PANELS IN THE MIDDLE
      DO 651 J=1,(NV-1)/2-2
      I=I+1
      IE=IE1+I
      NODEB(IE,1)=NH*(NV-1)/2+1+J*NH
      NODEB(IE,2)=NODEB(IE,1)+NH
      NODEB(IE,3)=NNK+1+J
      I=I+1
      IE=IE1+I
      NODEB(IE,1)=NH*(NV-1)/2+1+J*NH
      NODEB(IE,2)=NNK+1+J
      NODEB(IE,3)=NODEB(IE,2)-1
      I=I+1
      IE=IE1+I
      NODEB(IE,1)=NNK+J
      NODEB(IE,2)=NODEB(IE,1)+1
      NODEB(IE,3)=NH*(NV-1)/2+1-J*NH
      I=I+1
      IE=IE1+I
      NODEB(IE,1)=NNK+J+1
      NODEB(IE,2)=NH*(NV-1)/2+1-(J+1)*NH
      NODEB(IE,3)=NODEB(IE,2)+NH
  651 CONTINUE
C -------------------------------------------------------------
C FINALLY, 2 PANELS NEAR T.E.
      I=I+1
      IE=IE1+I
      NODEB(IE,1)=NH*(NV-1)/2+1+NH*((NV-1)/2-1)
      NODEB(IE,2)=1
      NODEB(IE,3)=NN1
      I=I+1
      IE=IE1+I
      NODEB(IE,1)=NN1
      NODEB(IE,2)=1
      NODEB(IE,3)=1+NH

C -------------------------------------------------------------
C CALCULATE CENTER COORDINATES OF WING PANELS (XIE,YIE,ZIE VECTORS)
      DO 200 IE=1,NN
        XIE(IE)=( X(NODEB(IE,1))+X(NODEB(IE,2))+X(NODEB(IE,3)) )/3.0D00
        YIE(IE)=( Y(NODEB(IE,1))+Y(NODEB(IE,2))+Y(NODEB(IE,3)) )/3.0D00
        ZIE(IE)=( Z(NODEB(IE,1))+Z(NODEB(IE,2))+Z(NODEB(IE,3)) )/3.0D00
  200 CONTINUE
C -------------------------------------------------------------
C CALCULATE HALF-WING REFERENCE SURFACE. THIS IS DONE BY A POLYGONAL
```

```
C DISCRETIZATION AROUND THE WING PERIMETER
      SURFW=0.0D00
      J=1
      DO 100 I=1,NH-1
        II=I
        SURFW= SURFW + 0.5D00*( X(II)*Y(II+1) - X(II+1)*Y(II) )
  100 CONTINUE
      I=NH
      DO 101 J=1,(NV-1)/2
        II=I+(J-1)*NH
        SURFW= SURFW + 0.5D00*( X(II)*Y(II+NH) - X(II+NH)*Y(II) )
  101 CONTINUE
      J=(NV-1)/2+1
      DO 102 I=NH,2,-1
        II=I+(J-1)*NH
        SURFW= SURFW + 0.5D00*( X(II)*Y(II-1) - X(II-1)*Y(II) )
  102 CONTINUE
      I=1
      DO 103 J=(NV-1)/2+1,3,-1
        II=I+(J-1)*NH
        SURFW= SURFW + 0.5D00*( X(II)*Y(II-NH) - X(II-NH)*Y(II) )
  103 CONTINUE
      I=1
      J=2
      II=I+(J-1)*NH
      SURFW=SURFW + X(II)*Y(1) - X(1)*Y(II)
      SURFW=DABS(SURFW)
      WRITE(6,*)'HALF WING REFERENCE AREA IS ',SURFW

C -------------------------------------------------------
C ASSOCIATE WING PANELS WITH WING NODES FOR CP CALCULATION (NEB ARRAY)
C LOOP 751 INITIALIZES VALUES.LOOP 711 IS FOR "INSIDE" UPPER NODES. LOOP
C 611 IS FOR "INSIDE" LOWER NODES. FINISHING LOOP 710 ARE L.E. NODES.
      DO 751 J=1,NBQ
      DO 751 I=1,NN1
      NEB(I,J)=0
  751 CONTINUE
      DO 710 I=2,NH-1
      DO 711 J=2,(NV-1)/2
      II=I+(J-1)*NH
      NEB(II,1)=1+(I-2)*2+(J-2)*2*NL+1
      NEB(II,2)=1+(I-2)*2+(J-2)*2*NL
      NEB(II,3)=1+(I-2)*2+(J-2)*2*NL+1+2
      NEB(II,4)=1+(I-2)*2+(J-2)*2*NL+2+2*NL
      NEB(II,5)=1+(I-2)*2+(J-2)*2*NL+2+2*NL+1
      NEB(II,6)=1+(I-2)*2+(J-2)*2*NL+2*NL
  711 CONTINUE
      DO 611 J=(NV-1)/2+2,NV-1
      II=I+(J-1)*NH
      NEB(II,1)=1+(I-2)*2+(J-2)*2*NL+1
      NEB(II,2)=1+(I-2)*2+(J-2)*2*NL+1+1
```

```
      NEB(II,3)=1+(I-2)*2+(J-2)*2*NL+1+2
      NEB(II,4)=1+(I-2)*2+(J-2)*2*NL+1+1+2*NL
      NEB(II,5)=1+(I-2)*2+(J-2)*2*NL+1+2*NL
      NEB(II,6)=1+(I-2)*2+(J-2)*2*NL+2*NL
  611 CONTINUE
      J=(NV-1)/2+1
      II=I+(J-1)*NH
      NEB(II,1)=1+(I-2)*2+(J-2)*2*NL+1
      NEB(II,2)=1+(I-2)*2+(J-2)*2*NL
      NEB(II,3)=1+(I-2)*2+(J-2)*2*NL+1+2
      NEB(II,4)=1+(I-2)*2+(J-2)*2*NL+2+2*NL
      NEB(II,5)=1+(I-2)*2+(J-2)*2*NL+1+2*NL
      NEB(II,6)=1+(I-2)*2+(J-2)*2*NL+2*NL
  710 CONTINUE

C -------------------------------------------------------------
C LOOP 720 IS FOR UPPER T.E. NODES. LOOP 730 IS FOR LOWER T.E. NODES
C LOOP 721 IS FOR UPPER NODES AT TIP, LOOP 421 FOR LOWER NODES AT TIP
C NEXT IS FOR NODE AT TIP & L.E. (II=I+(J-1)*NH)
C NOT AS MANY PANELS ARE USED FOR THESE NODES. TIP PANELS NOT USED
      DO 720 I=2,NH-1
      J=1
      II=I
      NEB(II,4)=1+(I-2)*2+2
      NEB(II,5)=1+(I-2)*2+2+1
      NEB(II,6)=1+(I-2)*2
  720 CONTINUE
      DO 730 I=2,NH-1
      J=NV
      II=I+(J-1)*NH-1
      NEB(II,1)=1+(I-2)*2+(J-2)*2*NL+1
      NEB(II,2)=1+(I-2)*2+(J-2)*2*NL+1+1
      NEB(II,3)=1+(I-2)*2+(J-2)*2*NL+1+2
  730 CONTINUE
      I=1
      DO 721 J=2,(NV-1)/2
      II=I+(J-1)*NH
      NEB(II,3)=1+(I-2)*2+(J-2)*2*NL+1+2
      NEB(II,4)=1+(I-2)*2+(J-2)*2*NL+1+1+2*NL
      NEB(II,5)=1+(I-2)*2+(J-2)*2*NL+2+2*NL+1
  721 CONTINUE
      DO 421 J=(NV-1)/2+2,NV-1
      II=I+(J-1)*NH
      NEB(II,2)=1+(I-2)*2+(J-2)*2*NL+2
      NEB(II,3)=1+(I-2)*2+(J-2)*2*NL+1+2
      NEB(II,4)=1+(I-2)*2+(J-2)*2*NL+2+2*NL
  421 CONTINUE
      J=(NV-1)/2+1
      II=I+(J-1)*NH
      NEB(II,3)=1+(I-2)*2+(J-2)*2*NL+1+2
      NEB(II,4)=1+(I-2)*2+(J-2)*2*NL+2+2*NL
```

```
C -------------------------------------------------------
C LOOP 731 IS FOR UPPER NODES AT WING ROOT, LOOP 431 FOR LOWER NODES
C AT WING ROOT. NEXT IS NODE AT ROOT & L.E. (II=I+(J-1)*NH).FINALLY,
C POINTS AT T.E. AT TIP & ROOT ARE TAKEN CARE OF (NODES #1, NH & NNK)
      I=NH
      DO 731 J=2,(NV-1)/2
      II=I+(J-1)*NH
      NEB(II,1)=1+(I-2)*2+(J-2)*2*NL+1
      NEB(II,2)=1+(I-2)*2+(J-2)*2*NL
      NEB(II,6)=1+(I-2)*2+(J-2)*2*NL+2*NL
  731 CONTINUE
      DO 431 J=(NV-1)/2+2,NV-1
      II=I+(J-1)*NH
      NEB(II,1)=1+(I-2)*2+(J-2)*2*NL+1
      NEB(II,5)=1+(I-2)*2+(J-2)*2*NL+1+2*NL
      NEB(II,6)=1+(I-2)*2+(J-2)*2*NL+2*NL
  431 CONTINUE
      J=(NV-1)/2+1
      II=I+(J-1)*NH
      NEB(II,1)=1+(I-2)*2+(J-2)*2*NL+1
      NEB(II,2)=1+(I-2)*2+(J-2)*2*NL
      NEB(II,5)=1+(I-2)*2+(J-2)*2*NL+2+2*NL+1
      NEB(II,6)=1+(I-2)*2+(J-2)*2*NL+2*NL
      NEB(1,3)=1
      NEB(1,4)=2
      NEB(1,5)=(NV-2)*2*NL+1
      NEB(1,6)=(NV-2)*2*NL+2
      NEB(NH,6)=2*NL-1
      NEB(NNK,1)=(NV-1)*2*NL
C -------------------------------------------------------
C CALCULATION OF IB,IBU,IBB & ISBU VECTORS THAT NOTE T.E. POINTS
C BY TAKING THE NUMBER OF THE NODE AS VALUE. IBB & IB ARE FOR LOWER
C T.E. NODES AND ISBU & IBU FOR UPPER T.E. NODES
      DO 99 I=1,NL
      IB(I)=I+NH*(NV-1)
      IBU(I)=I+1
   99 CONTINUE
      DO 95 I=1,NN1
      IBB(I)=0
      ISBU(I)=0
   95 CONTINUE
      DO 193 I=1,NL
      IBB(IB(I))=IB(I)
      ISBU(IBU(I))=IBU(I)
  193 CONTINUE
C -------------------------------------------------------
C IF THERE ARE NOT EQUAL ALREADY, GIVE SAME AVERAGE VALUE FOR UPPER &
C LOWER T.E. POINTS (TO INSURE AN INFINITESIMAL THICKNESS OF THE WAKE)
      DO 1002 K=1,NL
      I=IB(K)
      IU=IBU(K)
```

```
      XA=(X(I)+X(IU))/2.0D00
      YA=(Y(I)+Y(IU))/2.0D00
      ZA=(Z(I)+Z(IU))/2.0D00
      X(I)=XA
      Y(I)=YA
      Z(I)=ZA
      X(IU)=XA
      Y(IU)=YA
      Z(IU)=ZA
 1002 CONTINUE
      RETURN
      END
C=======================================================
      SUBROUTINE BNODE
C THIS S/R ASSOCIATES THE T.E. NODES WITH THE WAKE PANELS (NODEW)
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/PI11/PI
      COMMON/NNNW/NODEW(NL,NWNO)

      DO 6000 IE=1,NL
      NODEW(IE,1)=IE+1
      NODEW(IE,2)=IE
      NODEW(IE,3)=IE+NH*(NV-1)-1
      NODEW(IE,4)=IE+NH*(NV-1)
      IF(IE.EQ.1)NODEW(IE,3)=1
 6000 CONTINUE

      RETURN
      END
C=======================================================
      SUBROUTINE XYZTRANW
C THIS S/R CALCULATES THE RELATIVE COORDS AND COSINES FOR WAKE PANELS
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/XYZT1/XT(3),YT(3),ZT(3)
      COMMON/XYZRW/XW(NL,NWUNO),YW(NL,NWUNO),ZW(NL,NWUNO)
      COMMON/CWLMN1/CWLX(NL),CWMX(NL),CWNX(NL)
```

```
      COMMON/CWLMN2/CWLY(NL),CWMY(NL),CWNY(NL)
      COMMON/CWLMN3/CWLZ(NL),CWMZ(NL),CWNZ(NL)
      COMMON/NNNW/NODEW(NL,NWNO)
C -------------------------------------------------------
C THE ORIGIN IS FIXED AT NODE #1 OF THE WAKE PANEL. THE XR AXIS IS IN
C THE SAME DIRECTION AS THE GENERAL X AXIS. YR & ZR AXIS ARE OBTAINED
C BY ROTATING THE Y & Z AXIS UNTIL THE ZERO-THICKNESS PANEL IS IN THE
C (XR,YR) PLANE.
C DO 170 CALCULATES THE DISTANCE OF PANEL NODES WITH PANEL ORIGIN.
C THEN, COSINES OF GENERAL AXIS WITH RELATIVE AXIS ARE EVALUATED (CW..)
C FINALLY, DO 1210 CALCULATES RELATIVE COORDS OF PANEL NODES.
      DO 7000 IE=1,NL

      DO 170 I=1,2
      XT(I)=X(NODEW(IE,I))-X(NODEW(IE,1))
      YT(I)=Y(NODEW(IE,I))-Y(NODEW(IE,1))
      ZT(I)=Z(NODEW(IE,I))-Z(NODEW(IE,1))
  170 CONTINUE

      CWLX(IE)=1.0D00
      CWMX(IE)=0.0D00
      CWNX(IE)=0.0D00

      CWLY(IE)=0.0D00
      CWMY(IE)=-YT(2)/DSQRT(ZT(2)*ZT(2)+YT(2)*YT(2))
      CWNY(IE)=-ZT(2)/DSQRT(ZT(2)*ZT(2)+YT(2)*YT(2))

      CWLZ(IE)=0.0D00
      CWMZ(IE)=ZT(2)/DSQRT(ZT(2)*ZT(2)+YT(2)*YT(2))
      CWNZ(IE)=-YT(2)/DSQRT(ZT(2)*ZT(2)+YT(2)*YT(2))

      DO 1210 I=1,2
      XW(IE,I)=CWLX(IE)*XT(I)+CWMX(IE)*YT(I)+CWNX(IE)*ZT(I)
      YW(IE,I)=CWLY(IE)*XT(I)+CWMY(IE)*YT(I)+CWNY(IE)*ZT(I)
      ZW(IE,I)=CWLZ(IE)*XT(I)+CWMZ(IE)*YT(I)+CWNZ(IE)*ZT(I)
 1210 CONTINUE

 7000 CONTINUE

      RETURN
      END
C=======================================================
      SUBROUTINE GRIDB
      IMPLICIT REAL*8 (A-H,O-Z)
C THIS S/R CALCULATES A PARALLEL GRID FOR WING SURFACE NODES. THIS IS
C REQUIRED IN THE POTENTIAL CALCULATIONS TO AVOID ZERO RADIUS IN SOME
C CASES.
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
```

```
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/PI11/PI
C -------------------------------------------------------
C X2,Y2,Z2 COORDS ARE CALCULATED FOR EACH WING NODES
C LOOP 100 IS FOR "INSIDE" NODES, UPPER & LOWER SURFACES (INCL. ROOT)
C LOOP 101 IS FOR L.E. POINTS (EXCEPT TIP)
C LOOP 102 IS FOR T.E. POINTS (EXCEPT TIP)
C USUALLY, POINTS ARE MOVED 10% OF THE DISTANCE BETWEEN OPPOSITE UPPER
C & LOWER SURFACE NODES
      DO 100 I=2,NH
      DO 200 J=2,(NV-1)/2
      II=I+(J-1)*NH
      II1=II+NH*(NV-1)-NH*(J-1)*2
      X2(II)=X(II)+(X(II1)-X(II))*0.1D00
      X2(II1)=X(II)+(X(II1)-X(II))*0.9D00
      Y2(II)=Y(II)+(Y(II1)-Y(II))*0.1D00
      Y2(II1)=Y(II)+(Y(II1)-Y(II))*0.9D00
      Z2(II)=Z(II)+(Z(II1)-Z(II))*0.1D00
      Z2(II1)=Z(II)+(Z(II1)-Z(II))*0.9D00
  200 CONTINUE
  100 CONTINUE

      DO 101 I=2,NH
      J=(NV-1)/2+1
      II=I+(J-1)*NH
      Y2(II)=Y(II)
      XAA=(X(II+NH)+X(II-NL-1))/2.0D00
      X2(II)=X(II)+(XAA-X(II))*0.1D00
      ZAA=(Z(II+NH)+Z(II-NL-1))/2.0D00
      Z2(II)=Z(II)+(ZAA-Z(II))*0.1D00
  101 CONTINUE

      DO 102 I=2,NH
      J=1
      II=I
      II1=II+NH*(NV-1)-1
      XAA=(X(II+NH)+X(II1-NL))/2.0D00
      X2(II)=X(II)+(XAA-X(II))*0.05D00
      X2(II1)=X2(II)
      Y2(II)=Y(II)
      Y2(II1)=Y(II)
      ZAA=(Z(II+NH)+Z(II1-NL))/2.0D00
      Z2(II)=Z(II)+(ZAA-Z(II))*0.05D00
      Z2(II1)=Z2(II)
  102 CONTINUE

C -------------------------------------------------------
C DO 300 IS FOR TIP NODES IN THE MIDDLE (UPPER & LOWER SURFACES ONLY)
```

```
C THEN, NODES AT TIP & L.E. AND AT TIP & T.E. (NODE 1) ARE EVALUATED
      I=1
      DO 300 J=2,(NV-1)/2
      II=I+(J-1)*NH
      II1=II+NH*(NV-1)-NH*(J-1)*2
      X2(II)=X(II)+(X(II1)-X(II))*0.1D00
      X2(II1)=X(II)+(X(II1)-X(II))*0.9D00
      YU=Y(II)+(Y(II+1)-Y(II))*0.01D00
      YD=Y(II1)+(Y(II1+1)-Y(II1))*0.01D00
      Y2(II)=YU+(YD-YU)*0.1D00
      Y2(II1)=YU+(YD-YU)*0.9D00
      Z2(II)=Z(II)+(Z(II1)-Z(II))*0.1D00
      Z2(II1)=Z(II)+(Z(II1)-Z(II))*0.9D00
  300 CONTINUE

      I=1
      J=(NV-1)/2+1
      II=I+(J-1)*NH
      XAA=(X(II+NH)+X(II-NL-1))/2.0D00
      X2(II)=X(II)+(XAA-X(II))*0.1D00
      Y2(II)=Y(II)+(Y(II+1)-Y(II))*0.01D00
      ZAA=(Z(II+NH)+Z(II-NL-1))/2.0D00
      Z2(II)=Z(II)+(ZAA-Z(II))*0.1D00

      I=1
      J=1
      II=I
      XAA=(X(NH*(NV-2)+1)+X(II+NH))/2.0D00
      X2(II)=X(II)+(XAA-X(II))*0.05D00
      Y2(II)=Y(II)+(Y(II+1)-Y(II))*0.001D00
      ZAA=(Z(NH*(NV-2)+1)+Z(II+NH))/2.0D00
      Z2(II)=Z(II)+(ZAA-Z(II))*0.05D00

C -------------------------------------------------------
C FINALLY, THE EXTRA TIP NODES (INSIDE,TIP GRID) ARE EVALUATED
      DO 569 I=NNK+1,NN1
      X2(I)=X(I)
      Y2(I)=Y(I)+0.001D00
      Z2(I)=Z(I)
  569 CONTINUE

      RETURN
      END
C=======================================================
      SUBROUTINE XYZTRANS
C THIS S/R CALCULATES THE RELATIVE COSINES & NODES COORDS. OF EACH PANEL
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
```

```
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/PI11/PI
      COMMON/XYZR1/XR(NN,NPNO),YR(NN,NPNO),ZR(NN,NPNO)
      COMMON/COLMN1/COLX(NN),COMX(NN),CONX(NN)
      COMMON/COLMN2/COLY(NN),COMY(NN),CONY(NN)
      COMMON/COLMN3/COLZ(NN),COMZ(NN),CONZ(NN)
      COMMON/XYZT1/XT(3),YT(3),ZT(3)
      COMMON/SAML/SMALL
      COMMON/NMNM1/NMNM
C -------------------------------------------------------------
C THE ORIGIN OF RELATIVE COORDS IS AT NODE 1. THE XR AXIS IS ALONG THE
C LINE JOINING NODES 1 AND 2. THE YR AXIS IS IN THE PLANE OF THE PANEL.
C THE ZR AXIS IS PERPENDICULAR TO THE PANEL SURFACE.
C LOOP 9000 IS REPEATED FOR EACH PANEL.
C LOOP 100 CALCULATES DISTANCES XT,YT & ZT OF NODES WITH ORIGIN.
C COSINES OF GENERAL AXIS WITH RELATIVE AXIS ARE THEN EVALUATED (CO..)
C LOOP 1110 CALCULATES THE REL. COORDS. OF THE PANEL NODES
      DO 9000 IE=1,NN

      DO 100 I=1,3
      XT(I)=X(NODEB(IE,I))-X(NODEB(IE,1))
      YT(I)=Y(NODEB(IE,I))-Y(NODEB(IE,1))
      ZT(I)=Z(NODEB(IE,I))-Z(NODEB(IE,1))
  100 CONTINUE

      A=YT(1)*ZT(2)+YT(3)*ZT(1)+YT(2)*ZT(3)
     1-YT(3)*ZT(2)-YT(2)*ZT(1)-YT(1)*ZT(3)
      B=XT(3)*ZT(2)+XT(2)*ZT(1)+XT(1)*ZT(3)
     1-XT(1)*ZT(2)-XT(3)*ZT(1)-XT(2)*ZT(3)
      C=XT(1)*YT(2)+XT(3)*YT(1)+XT(2)*YT(3)
     1-XT(3)*YT(2)-XT(2)*YT(1)-XT(1)*YT(3)
      D=XT(3)*YT(2)*ZT(1)+XT(2)*YT(1)*ZT(3)+XT(1)*YT(3)*ZT(2)
     1-XT(1)*YT(2)*ZT(3)-XT(2)*YT(3)*ZT(1)-XT(3)*YT(1)*ZT(2)

      ABSV=0.0000000000001D00
      IF(DABS(A).LT.ABSV)THEN
      COLZ(IE)=0.0D00
      ELSE
      COLZ(IE)=A/DABS(A)*1.0D00/DSQRT(1.0D00+B*B/A/A+C*C/A/A)
      END IF

      IF(DABS(B).LT.ABSV)THEN
      COMZ(IE)=0.0D00
      ELSE
      COMZ(IE)=B/DABS(B)*1.0D00/DSQRT(1.0D00+A*A/B/B+C*C/B/B)
      END IF

      IF(DABS(C).LT.ABSV)THEN
```

```
      CONZ(IE)=0.0D00
      ELSE
      CONZ(IE)=C/DABS(C)*1.0D00/DSQRT(1.0D00+A*A/C/C+B*B/C/C)
      END IF

      R12=DSQRT(XT(2)*XT(2)+YT(2)*YT(2)+ZT(2)*ZT(2))
      COLX(IE)=XT(2)/R12
      COMX(IE)=YT(2)/R12
      CONX(IE)=ZT(2)/R12

      R23=DSQRT((XT(3)-XT(2))*(XT(3)-XT(2))
     1+(YT(3)-YT(2))*(YT(3)-YT(2))
     2+(ZT(3)-ZT(2))*(ZT(3)-ZT(2)))

      R31=DSQRT(XT(3)*XT(3)+YT(3)*YT(3)+ZT(3)*ZT(3))

      RH=(R12*R12+R31*R31-R23*R23)/2.0D00/R12
      XH=RH*COLX(IE)
      YH=RH*COMX(IE)
      ZH=RH*CONX(IE)

      X3H=XT(3)-XH
      Y3H=YT(3)-YH
      Z3H=ZT(3)-ZH
      R3H=DSQRT(X3H*X3H+Y3H*Y3H+Z3H*Z3H)
      COLY(IE)=X3H/R3H
      COMY(IE)=Y3H/R3H
      CONY(IE)=Z3H/R3H

      DO 1110 I=1,3
      XR(IE,I)=COLX(IE)*XT(I)+COMX(IE)*YT(I)+CONX(IE)*ZT(I)
      YR(IE,I)=COLY(IE)*XT(I)+COMY(IE)*YT(I)+CONY(IE)*ZT(I)
      ZR(IE,I)=COLZ(IE)*XT(I)+COMZ(IE)*YT(I)+CONZ(IE)*ZT(I)
 1110 CONTINUE
C -------------------------------------------------------
C TESTS TO VERIFY IF NODE #3 IS LOCATED BETWEEN NODES #1 & #2 (ALONG XR)
C THE FIRST "IF" IS TO VERIFY IF #3 IS NOT TOO MUCH TO THE LEFT.
C THE NEXT ONE CHECKS IF #3 IS TOO FAR RIGHT.
C IN BOTH CASES, THERE WILL BE A RE-NUMEROTATION OF THE NODES FOR THAT
C PANEL (THE RELATIVE ORIGIN WILL CHANGE), FLAG "NMNM" IS INCREMENTED
C THIS IS REQUIRED BECAUSE OF THE INTEGRAL DISCRETIZATION (S/R FI)
      IF(DABS(XR(IE,3)).LT.SMALL.OR.XR(IE,3).LT.0.0D00 )THEN
      N1=NODEB(IE,1)
      N2=NODEB(IE,2)
      N3=NODEB(IE,3)
      NODEB(IE,1)=N2
      NODEB(IE,2)=N3
      NODEB(IE,3)=N1
      NMNM=NMNM+1
      END IF
```

```
      IF(DABS(XR(IE,3)-XR(IE,2)).LT.SMALL.OR.XR(IE,3).GT.XR(IE,2))THEN
      N1=NODEB(IE,1)
      N2=NODEB(IE,2)
      N3=NODEB(IE,3)
      NODEB(IE,1)=N3
      NODEB(IE,2)=N1
      NODEB(IE,3)=N2
      NMNM=NMNM+1
      END IF

 9000 CONTINUE

      RETURN
      END
C=============================================================
      SUBROUTINE XYZGN
C THIS S/R CALCULATES THE FIELD GRID COORDS. AND ASSOCIATES THE NODES
C WITH THE FIELD ELEMENTS
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/NNNW/NODEW(NL,NWNO)
      COMMON/NNODE1/NODE(NELEM,NDPL)
      COMMON/XYZG/X3(NGLOB),Y3(NGLOB),Z3(NGLOB)
      COMMON/XYZGIE/X3G(NELEM),Y3G(NELEM),Z3G(NELEM),V3G(NELEM)
      COMMON/XYZA1/XAL(MZ2),YAL(MY),YAT(MY),ZAU(MZ2)
      COMMON/NGL1/NGL(NGLOB),NGLT(NN1),NWAKE(NGLOB)
      COMMON/DG10/DG0,DG1
      COMMON/YMAX11/YMAX,YHSPAN,FRAC,SURFW
      COMMON/DN11/DNLOC1(8,3,8)
      REAL RR(8),SS(8),TT(8)
C -------------------------------------------------------------
C BASED ON DG0 VALUE, DO 1789 CALCULATES THE "NO" DISTANCES IN X DIREC-
C TION (XAL), Z DIRECTION (ZAU) AND Y DIRECTION (YAL & YAT)
C 1ST FIELD POINT WILL BE AT DG0 DISTANCE FROM WING, SECOND AT
C DG0+1.5*DG0, THIRD AT DG0+1.5*DG0+2.0*DG0, ETC. I.E. THERE IS A 0.5*
C DG0 INCREMENT FOR EACH DISTANCE BETWEEN NODES (GRID GETS BIGGER)
      DO 1789 L=1,NO
      FN1L=DFLOAT(NO+1-L)
      FNL=DFLOAT(NO-L)
      YAL(L)=-YMAX-DG0*FN1L-DG1*FN1L*FNL/2.0D00
      YAT(L)=YAL(L)
      XAL(L)=DG0*FN1L+DG1*FN1L*FNL/2.0D00
      ZAU(L)=DG0*FN1L+DG1*FN1L*FNL/2.0D00
 1789 CONTINUE
```

```
      XAL(NO+1)=0.0D00

C -------------------------------------------------------
C DO 1011 CALCULATES Y COORDS. OF NODES AT L.E. (YAL) AND T.E (YAT)
C BASED ON CORRESPONDING WING NODES
      DO 1011 L=NO+1,MY
      YAT(L)=Y(L-NO)
      YAL(L)=Y(L-NO+NH*(NV-1)/2)
 1011 CONTINUE

C -------------------------------------------------------
C ASSOCIATION OF MEETING FIELD AND WING NODES
C DO 201 INITIALIZES "NGL", DO 271 "NGLT"  & DO 272 "NGLOB"
C NGL & NGLT VECTORS TAKE VALUE OF CORRESPONDING NODE
C LOOP 111 IS FOR UPPER NODES AND LOOP 511 FOR LOWER NODES (WITH SPECIAL
C CASES FOR THE T.E.)
      DO 201 I=1,NGLOB
      NGL(I)=0
  201 CONTINUE

      DO 271 I=1,NN1
      NGLT(I)=0
  271 CONTINUE

      DO 272 I=1,NGLOB
      NWAKE(I)=0
  272 CONTINUE

      DO 111 J=NO+1,NO+(NV-1)/2+1
      DO 112 K=NO+1,MY
      II=K+(J-1)*MY+(MZ2-1)*MY*MX
      JJ=NH*(NV-1)/2+1+(K-NO-1)-(J-NO-1)*NH
      NGL(II)=JJ
      NGLT(JJ)=II
  112 CONTINUE
  111 CONTINUE

      DO 511 J=NO+1,NO+(NV-1)/2+1
      DO 512 K=NO+1,MY
      II=K+(J-1)*MY+(MZ2)*MY*MX
      JJ=NH*(NV-1)/2+1+(K-NO-1)+(J-NO-1)*NH
      NGL(II)=JJ
      NGLT(JJ)=II
      IF(J.EQ.NO+(NV-1)/2+1)THEN
      NGL(II)=NGL(II)-1
      NGLT(JJ-1)=II
      IF(K.EQ.NO+1)NGL(II)=1
      IF(K.EQ.NO+1)NGLT(1)=II
      END IF
  512 CONTINUE
  511 CONTINUE
```

```
C -------------------------------------------------------
C LOOP 1312 CALCULATES COORDINATES OF FIELD NODES IN UPPER WING PLANE
C THE PLANE IS DIVIDED IN ZONES. ZONE 3B COVERS THE WING
C
C        _____ _____________________
C       |  1A |          1B         |
C       |_____|_____________________| L.E.          |-----> Y
C       |     |                     |               |
C       |  3A |TIP      3B          |WING ROOT      |
C       |_____|_____________________|               X
C       |  2A |          2B         |T.E.
C       |_____|_____________________|
C
      DO 1312 J=1,MX
      DO 1313 K=1,MY
      II=K+(J-1)*MY+(MZ2-1)*MY*MX
      IF(J.LT.NO+2)THEN
        Y3(II)=YAL(K)
         IF(K.LT.NO+2)THEN
C ZONE 1A
            Z3(II)=Z(NH*(NV-1)/2+1)
            X3(II)=X(NH*(NV-1)/2+1)-XAL(J)
          ELSE
C ZONE 1B
            Z3(II)=Z(NH*(NV-1)/2+1-(NO+1)+K)
            X3(II)=X(NH*(NV-1)/2+1-(NO+1)+K)-XAL(J)
          END IF
       ELSE
          IF(J.GT.NO+(NV-1)/2)THEN
            Y3(II)=YAT(K)
             IF(K.LT.NO+2)THEN
C ZONE 2A
               Z3(II)=Z(1)
               X3(II)=X(1)+XAL(MX+1-J)
             ELSE
C ZONE 2B
               Z3(II)=Z(K-NO)
               X3(II)=X(K-NO)+XAL(MX+1-J)
             END IF
          ELSE
            Y3(II)=YAL(K)
            JJ=NH*(NV-1)/2+1-NH-(J-(NO+2))*NH
            J1=NNK+1+(J-(NO+2))
            IF(K.LT.NO+1)THEN
C ZONE 3A
               Z3(II)=Z(J1)
               X3(II)=X(J1)
C           ELSE
C ZONE 3B
C              Z3(II)=Z(JJ+(K-(NO+1)))
C              X3(II)=X(JJ+(K-(NO+1)))
```

```
          END IF
        END IF
      END IF
 1313 CONTINUE
 1312 CONTINUE

C -------------------------------------------------------------
C CALCULATION OF FIELD NODE COORDINATES FOR LOWER WING PLANE
C IN ZONE 2B, WAKE NODES ARE NOTED IN VECTOR "WAKEN"
      DO 2312 J=1,MX
      DO 2313 K=1,MY
      II=K+(J-1)*MY+(MZ2)*MY*MX
      IF(J.LT.NO+2)THEN
        Y3(II)=YAL(K)
        IF(K.LT.NO+2)THEN
C ZONE 1A
          Z3(II)=Z(NH*(NV-1)/2+1)
          X3(II)=X(NH*(NV-1)/2+1)-XAL(J)
        ELSE
C ZONE 1B
          Z3(II)=Z(NH*(NV-1)/2+1-(NO+1)+K)
          X3(II)=X(NH*(NV-1)/2+1-(NO+1)+K)-XAL(J)
        END IF
      ELSE
        IF(J.GT.NO+(NV-1)/2)THEN
          Y3(II)=YAT(K)
          IF(K.LT.NO+2)THEN
C ZONE 2A
            Z3(II)=Z(1)
            X3(II)=X(1)+XAL(MX+1-J)
          ELSE
C ZONE 2B
            Z3(II)=Z(K-NO)
            X3(II)=X(K-NO)+XAL(MX+1-J)
            NWAKE(II)=II
          END IF
        ELSE
          Y3(II)=YAL(K)
          JJ=NH*(NV-1)/2+1+NH+(J-(NO+2))*NH
          J1=NNK+1+(J-(NO+2))
          IF(K.LT.NO+1)THEN
C ZONE 3A
            Z3(II)=Z(J1)
            X3(II)=X(J1)
C           ELSE
C ZONE 3B
C             Z3(II)=Z(JJ+(K-(NO+1)))
C             X3(II)=X(JJ+(K-(NO+1)))
          END IF
        END IF
      END IF
```

```
 2313 CONTINUE
 2312 CONTINUE
C ---------------------------------------------------------
C BODY FITTING OF SPECIFIED FIELD NODES OVER WING NODES
      DO 900 I=1,NGLOB
      IF(NGL(I).NE.0)THEN
      X3(I)=X(NGL(I))
      Y3(I)=Y(NGL(I))
      Z3(I)=Z(NGL(I))
      END IF
  900 CONTINUE
C ---------------------------------------------------------
C CALCULATION OF FIELD NODES COORDS. OVER UPPER WING PLANE
C ONLY THE Z3 COORDS CHANGE WITH PREVIOUS PLANES
      DO 4311 I=1,NO
      DO 4312 J=1,MX
      DO 4313 K=1,MY
      II=K+(J-1)*MY+(I-1)*MY*MX
      II1=K+(J-1)*MY+(MZ2-1)*MY*MX
      X3(II)=X3(II1)
      Y3(II)=Y3(II1)
      Z3(II)=Z3(II1)+ZAU(I)
 4313 CONTINUE
 4312 CONTINUE
 4311 CONTINUE
C ---------------------------------------------------------
C CALCULATION OF FIELD NODES COORDS. UNDER LOWER WING PLANE
C ONLY THE Z3 COORDS CHANGE WITH PREVIOUS PLANES
      DO 3311 I=MZ2+2,MZ
      DO 3312 J=1,MX
      DO 3313 K=1,MY
      II=K+(J-1)*MY+(I-1)*MY*MX
      II1=K+(J-1)*MY+(MZ2)*MY*MX
      X3(II)=X3(II1)
      Y3(II)=Y3(II1)
      Z3(II)=Z3(II1)-ZAU(MZ+1-I)
 3313 CONTINUE
 3312 CONTINUE
 3311 CONTINUE
C ---------------------------------------------------------
C ASSOCIATION OF FIELD NODES WITH FIELD ELEMENTS
C CONVENTION OF NODES NUMBERING IS:
C         8_________7
C        /|        /
C     5/_|_____6_/ |     Z   Y
C      |  |_______|__|     | /
C      | /4       |  /3    |/
C      |/_________|/       |------>X
C     1           2
C
C ELEMENT NUMBERING STARTS AT THE TOP CORNER OF FIELD GRID, WING TIP &
```

```
C L.E. SIDES, AND IS INCREMENTED ALONG Y AXIS THEN X AXIS.
C DO 310 FOR ELEMENTS OVER THE WING, DO 410 FOR ELEMENTS UNDER
      MXY=MX*MY
      DO 310 I=1,NO
      DO 311 J=1,MX-1
      DO 312 K=1,MY-1
      IE=K+(J-1)*(MY-1)+(I-1)*(MY-1)*(MX-1)
      II=K+(J-1)*MY+(I-1)*MY*MX
      NODE(IE,1)=II+MXY
      NODE(IE,2)=II+MXY+MY
      NODE(IE,3)=II+MXY+MY+1
      NODE(IE,4)=II+MXY+1
      NODE(IE,5)=II
      NODE(IE,6)=II+MY
      NODE(IE,7)=II+MY+1
      NODE(IE,8)=II+1
  312 CONTINUE
  311 CONTINUE
  310 CONTINUE
      DO 410 I=NO+1,2*NO
      DO 411 J=1,MX-1
      DO 412 K=1,MY-1
      IE=K+(J-1)*(MY-1)+(I-1)*(MY-1)*(MX-1)
      IK=I+1
      II=K+(J-1)*MY+(IK-1)*MY*MX
      NODE(IE,1)=II+MXY
      NODE(IE,2)=II+MXY+MY
      NODE(IE,3)=II+MXY+MY+1
      NODE(IE,4)=II+MXY+1
      NODE(IE,5)=II
      NODE(IE,6)=II+MY
      NODE(IE,7)=II+MY+1
      NODE(IE,8)=II+1
  412 CONTINUE
  411 CONTINUE
  410 CONTINUE
C -------------------------------------------------------------
C CALCULATION OF THE CENTERS X3G,Y3G & Z3G AND VOLUMES V3G OF ELEMENTS
      DO 370 IE=1,NELEM
      SUMX=0.0D00
      SUMY=0.0D00
      SUMZ=0.0D00
      DO 380 J=1,8
      SUMX=SUMX+X3(NODE(IE,J))
      SUMY=SUMY+Y3(NODE(IE,J))
      SUMZ=SUMZ+Z3(NODE(IE,J))
  380 CONTINUE
      X3G(IE)=SUMX/8.0D00
      Y3G(IE)=SUMY/8.0D00
      Z3G(IE)=SUMZ/8.0D00
      DZ=Z3(NODE(IE,5))-Z3(NODE(IE,1))
```

```
      DY=Y3(NODE(IE,4))-Y3(NODE(IE,1))
      AX=(X3(NODE(IE,4))+X3(NODE(IE,1)))/2.0D00
      BX=(X3(NODE(IE,3))+X3(NODE(IE,2)))/2.0D00
      DX=BX-AX
      V3G(IE)=DX*DY*DZ
  370 CONTINUE
C -------------------------------------------------------------
C CALCULATION OF ARRAY "DNLOC(8,3,8)"; THIS ARRAY IS USED IN S/R
C SHAP1. VECTORS RR(8), SS(8) & TT(8) MAKE COMBINAISONS OF 1 & -1
      RR(1)=-1.0D00
      RR(2)=1.0D00
      RR(3)=1.0D00
      RR(4)=-1.0D00
      RR(5)=-1.0D00
      RR(6)=1.0D00
      RR(7)=1.0D00
      RR(8)=-1.0D00

      SS(1)=-1.0D00
      SS(2)=-1.0D00
      SS(3)=1.0D00
      SS(4)=1.0D00
      SS(5)=-1.0D00
      SS(6)=-1.0D00
      SS(7)=1.0D00
      SS(8)=1.0D00

      TT(1)=-1.0D00
      TT(2)=-1.0D00
      TT(3)=-1.0D00
      TT(4)=-1.0D00
      TT(5)=1.0D00
      TT(6)=1.0D00
      TT(7)=1.0D00
      TT(8)=1.0D00

      DO 400 I=1,8
      DNLOC1(1,1,I)=-(1.0D00-SS(I))*(1.0D00-TT(I))/8.0D00
      DNLOC1(1,2,I)=-(1.0D00-RR(I))*(1.0D00-TT(I))/8.0D00
      DNLOC1(1,3,I)=-(1.0D00-RR(I))*(1.0D00-SS(I))/8.0D00
      DNLOC1(2,1,I)=(1.0D00-SS(I))*(1.0D00-TT(I))/8.0D00
      DNLOC1(2,2,I)=-(1.0D00+RR(I))*(1.0D00-TT(I))/8.0D00
      DNLOC1(2,3,I)=-(1.0D00+RR(I))*(1.0D00-SS(I))/8.0D00
      DNLOC1(3,1,I)=(1.0D00+SS(I))*(1.0D00-TT(I))/8.0D00
      DNLOC1(3,2,I)=(1.0D00+RR(I))*(1.0D00-TT(I))/8.0D00
      DNLOC1(3,3,I)=-(1.0D00+RR(I))*(1.0D00+SS(I))/8.0D00
      DNLOC1(4,1,I)=-(1.0D00+SS(I))*(1.0D00-TT(I))/8.0D00
      DNLOC1(4,2,I)=(1.0D00-RR(I))*(1.0D00-TT(I))/8.0D00
      DNLOC1(4,3,I)=-(1.0D00-RR(I))*(1.0D00+SS(I))/8.0D00
      DNLOC1(5,1,I)=-(1.0D00-SS(I))*(1.0D00+TT(I))/8.0D00
      DNLOC1(5,2,I)=-(1.0D00-RR(I))*(1.0D00+TT(I))/8.0D00
```

```
      DNLOC1(5,3,I)=(1.0D00-RR(I))*(1.0D00-SS(I))/8.0D00
      DNLOC1(6,1,I)=(1.0D00-SS(I))*(1.0D00+TT(I))/8.0D00
      DNLOC1(6,2,I)=-(1.0D00+RR(I))*(1.0D00+TT(I))/8.0D00
      DNLOC1(6,3,I)=(1.0D00+RR(I))*(1.0D00-SS(I))/8.0D00
      DNLOC1(7,1,I)=(1.0D00+SS(I))*(1.0D00+TT(I))/8.0D00
      DNLOC1(7,2,I)=(1.0D00+RR(I))*(1.0D00+TT(I))/8.0D00
      DNLOC1(7,3,I)=(1.0D00+RR(I))*(1.0D00+SS(I))/8.0D00
      DNLOC1(8,1,I)=-(1.0D00+SS(I))*(1.0D00+TT(I))/8.0D00
      DNLOC1(8,2,I)=(1.0D00-RR(I))*(1.0D00+TT(I))/8.0D00
      DNLOC1(8,3,I)=(1.0D00-RR(I))*(1.0D00+SS(I))/8.0D00
  400 CONTINUE

      RETURN
      END
C=======================================================
      SUBROUTINE BUILDA
C THIS S/R ASSEMBLES THE "A" MATRIX AND THE RIGHT SIDE TERM
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/PI11/PI
      COMMON/XYZR1/XR(NN,NPNO),YR(NN,NPNO),ZR(NN,NPNO)
      COMMON/COLMN1/COLX(NN),COMX(NN),CONX(NN)
      COMMON/COLMN2/COLY(NN),COMY(NN),CONY(NN)
      COMMON/COLMN3/COLZ(NN),COMZ(NN),CONZ(NN)
      COMMON/XYZT1/XT(3),YT(3),ZT(3)
      COMMON/XYZ21R/XP,YP,ZP
      COMMON/F123/F1,F2,F3
      COMMON/ST11/STIF(4)
      COMMON/ABC11/ALPHA,BETA,GAMA
      COMMON/BKARHS/A(NN1,NN1),RHS1(NN1),BA1(NN1,NN1),BAR(NN1)
      COMMON/XYZ21W/XPW,YPW,ZPW
      COMMON/F12W/F3W
      COMMON/NNNW/NODEW(NL,NWNO)
      COMMON/IBN/IB(NL),IBU(NL),ISBU(NN1),IBB(NN1)
      COMMON/SAML/SMALL
      COMMON/XYZRW/XW(NL,NWUNO),YW(NL,NWUNO),ZW(NL,NWUNO)
      COMMON/CWLMN1/CWLX(NL),CWMX(NL),CWNX(NL)
      COMMON/CWLMN2/CWLY(NL),CWMY(NL),CWNY(NL)
      COMMON/CWLMN3/CWLZ(NL),CWMZ(NL),CWNZ(NL)
C -------------------------------------------------------
C INITIALIZATION OF ARRAYS "A" & "RHS1"
      DO 40 J=1,NN1
      RHS1(J)=0.0D00
      DO 30 I=1,NN1
```

```
      A(J,I)=0.0D00
   30 CONTINUE
   40 CONTINUE

C ------------------------------------------------------------
C LOOP 7001 IS REPEATED FOR EACH WING NODE
C LOWER T.E. NODES ARE REMOVED FROM THE LOOP (J=IBB(J)) FOR S/R BC
      DO 7001 J=1,NN1

      IF(J.EQ.IBB(J))GO TO 7001

C ------------------------------------------------------------
C EFFECT OF WING PANELS ON WING NODES
C DISTANCES OF NODE WITH PANEL ORIGINS ARE FIRST EVALUATED (X2P,Y2P,Z2P)
C RELATIVE COORDS OF NODE XP,YP,ZP FOLLOW. PARALLEL GRID USED.
      DO 7000 IE=1,NN

      X2P=X2(J)-X(NODEB(IE,1))
      Y2P=Y2(J)-Y(NODEB(IE,1))
      Z2P=Z2(J)-Z(NODEB(IE,1))

      XP=COLX(IE)*X2P+COMX(IE)*Y2P+CONX(IE)*Z2P
      YP=COLY(IE)*X2P+COMY(IE)*Y2P+CONY(IE)*Z2P
      ZP=COLZ(IE)*X2P+COMZ(IE)*Y2P+CONZ(IE)*Z2P

C ------------------------------------------------------------
C CALCULATION OF VARIABLES F1,F2 & F3 WITH S/R FI
C CALCULATION OF "STIFNESS" TERMS STIF(I)

      CALL FI(IE,J)
      STIF(1)=(F3-(F1+XP*F3)/XR(IE,2)
     1+(XR(IE,3)-XR(IE,2))*(F2+YP*F3)/XR(IE,2)/YR(IE,3))

      STIF(2)=((F1+XP*F3)/XR(IE,2)
     1-(F2+YP*F3)*XR(IE,3)/XR(IE,2)/YR(IE,3))

      STIF(3)=(F2+YP*F3)/YR(IE,3)

C ------------------------------------------------------------
C REPEAT CALCULATIONS FOR OTHER SIDE OF WING PANELS
C ONLY THE Y2 COORDINATE CHANGES
      Y2P=-Y2(J)-Y(NODEB(IE,1))

      XP=COLX(IE)*X2P+COMX(IE)*Y2P+CONX(IE)*Z2P
      YP=COLY(IE)*X2P+COMY(IE)*Y2P+CONY(IE)*Z2P
      ZP=COLZ(IE)*X2P+COMZ(IE)*Y2P+CONZ(IE)*Z2P

      CALL FI(IE,J)
      STIF(1)=STIF(1)+(F3-(F1+XP*F3)/XR(IE,2)
     1+(XR(IE,3)-XR(IE,2))*(F2+YP*F3)/XR(IE,2)/YR(IE,3))
```

```
      STIF(2)=STIF(2)+((F1+XP*F3)/XR(IE,2)
     1-(F2+YP*F3)*XR(IE,3)/XR(IE,2)/YR(IE,3))

      STIF(3)=STIF(3)+(F2+YP*F3)/YR(IE,3)

C -------------------------------------------------------
C MODIFICATION OF A(J,NCOL) TERMS RELATED TO PANEL NODES EFFECT
C ON NODE J
      DO 20 K=1,3
      NCOL=NODEB(IE,K)
      A(J,NCOL)=A(J,NCOL)-STIF(K)
   20 CONTINUE

 7000 CONTINUE

C -------------------------------------------------------
C EFFECT OF WAKE PANELS ON WING NODES
C X2P,Y2P & Z2P ARE DISTANCES OF NODE WITH WAKE PANEL ORIGIN
C XPW,YPW & ZPW ARE RELATIVE COORDS. OF NODE FROM WAKE PANEL
      DO 8000 IEW=1,NL

      X2P=X2(J)-X(NODEW(IEW,1))
      Y2P=Y2(J)-Y(NODEW(IEW,1))
      Z2P=Z2(J)-Z(NODEW(IEW,1))

      XPW=CWLX(IEW)*X2P+CWMX(IEW)*Y2P+CWNX(IEW)*Z2P
      YPW=CWLY(IEW)*X2P+CWMY(IEW)*Y2P+CWNY(IEW)*Z2P
      ZPW=CWLZ(IEW)*X2P+CWMZ(IEW)*Y2P+CWNZ(IEW)*Z2P
C -------------------------------------------------------------
C FI3W S/R CALCULATES F3W TERM. STIF TERMS ARE THE EFFECT OF THE 4 NODES
      CALL FI3W(IEW,J)

      STIF(1)=F3W/2.0D00
      STIF(2)=STIF(1)
      STIF(3)=-STIF(1)
      STIF(4)=-STIF(1)
C -------------------------------------------------------------
C CALCULATIONS ARE REPEATED FOR OTHER SIDE WAKE PANELS. ONLY Y2
CHANGES
      Y2P=-Y2(J)-Y(NODEW(IEW,1))

      XPW=CWLX(IEW)*X2P+CWMX(IEW)*Y2P+CWNX(IEW)*Z2P
      YPW=CWLY(IEW)*X2P+CWMY(IEW)*Y2P+CWNY(IEW)*Z2P
      ZPW=CWLZ(IEW)*X2P+CWMZ(IEW)*Y2P+CWNZ(IEW)*Z2P

      CALL FI3W(IEW,J)

      STIF(1)=STIF(1)+F3W/2.0D00
      STIF(2)=STIF(2)+F3W/2.0D00
      STIF(3)=STIF(3)-F3W/2.0D00
      STIF(4)=STIF(4)-F3W/2.0D00
```

```
C -------------------------------------------------------
C MODIFICATION OF "A" MATRIX FOR THE 4 T.E. NODES OF WAKE PANEL
      DO 21 K=1,4
      NCOL=NODEW(IEW,K)
      A(J,NCOL)=A(J,NCOL)-STIF(K)
   21 CONTINUE

 8000 CONTINUE

C -------------------------------------------------------
C CALCULATION OF INCOMPRESSIBLE RIGHT HAND SIDE VECTOR
C 4*PI COMES FROM THE "A" MATRIX COEFFICIENTS
      RHS1(J)=(X2(J)*DCOS(ALPHA)+Y2(J)*DCOS(BETA)+Z2(J)*DCOS(GAMA))
     1*4.0D00*PI
 7001 CONTINUE

      RETURN
      END
C=======================================================
      SUBROUTINE BC
C THIS S/R ASSURES THE KUTTA CONDITION AT THE WING T.E.
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/BKARHS/A(NN1,NN1),RHS1(NN1),BA1(NN1,NN1),BAR(NN1)
      COMMON/IBN/IB(NL),IBU(NL),ISBU(NN1),IBB(NN1)
C -------------------------------------------------------
C APPLICATION OF KUTTA CONDITION TO LOWER T.E. NODES
C KUTTA CONDITION IS DONE BY EQUALLING UPPER & LOWER SPEEDS.
C UPPER SPEED IS CALCULATED BY DIVIDING THE DIFFERENCE IN POTENTIAL
C OF THE LAST 2 NODES BY THEIR DISTANCE "DS1". CORRESPONDING LOWER
C PANEL DISTANCE IS "DS". "A" MATRIX  TERMS ARE THEN MODIFIED
C ACCORDINGLY TO REPRESENT THE SPEEDS EQUALITY. ALL OTHER TERMS OF
C "A" AND "RHS1" TERM TAKE VALUE OF 0.0 FOR THAT NODE ROW "I"
C FOR EXAMPLE: (Q(2)-Q(9))/DS1=(Q(43)-Q(37))/DS
      DO 811 K=1,NL
      I=IB(K)
      IU=IBU(K)
      DO 100 J=1,NN1
      A(I,J)=0.0D00
  100 CONTINUE
      DX=X(I)-X(I-NL)
      DY=Y(I)-Y(I-NL)
      DZ=Z(I)-Z(I-NL)
      DS=DSQRT(DX*DX+DY*DY+DZ*DZ)
      DX1=X(IU)-X(IU+NH)
```

```
      DY1=Y(IU)-Y(IU+NH)
      DZ1=Z(IU)-Z(IU+NH)
      DS1=DSQRT(DX1*DX1+DY1*DY1+DZ1*DZ1)
      A(I,I)=1.0D00
      A(I,I-NL)=-1.0D00
      A(I,IU)=-1.0D00/DS1*DS
      A(I,IU+NH)=1.0D00/DS1*DS
      RHS1(I)=0.0D00
  811 CONTINUE

      RETURN
      END
C=============================================================
      SUBROUTINE LUDC
C MATRIX DECOMPOSER WITH "LU" METHOD
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))

      COMMON/BKARHS/A(NN1,NN1),RHS1(NN1),BA1(NN1,NN1),BAR(NN1)
C -------------------------------------------------------
C THE INVERSE OF "A" IS STORED IN "A" ARRAY.
C LOOP 200 CALCULATES THE "L" SIDE.
C LOOP 400 CALCULATES THE "U" SIDE.
      DO 100 M=1,NN1
      DO 200 I=M,NN1
      SUMM=0.0D00
      DO 300 K=1,M-1
      SUMM=SUMM+A(I,K)*A(K,M)
  300 CONTINUE
      A(I,M)=A(I,M)-SUMM
  200 CONTINUE
      DO 400 J=M+1,NN1
      SUMM=0.0D00
      DO 500 K=1,M-1
      SUMM=SUMM+A(M,K)*A(K,J)
  500 CONTINUE
      A(M,J)=(A(M,J)-SUMM)/A(M,M)
  400 CONTINUE
  100 CONTINUE
      RETURN
      END
C=============================================================
      SUBROUTINE LUBS
C THIS S/R CALCULATES THE SOLUTION OF INVERSE MATRIX "A" AND RIGHT SIDE
C TERM "RHS1". SOLUTION IS STORED ALSO IN "RHS1"
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
```

```
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))

      COMMON/BKARHS/A(NN1,NN1),RHS1(NN1),BA1(NN1,NN1),BAR(NN1)
C -------------------------------------------------------------
C BACK SUBSTITUTION IS USED TO EVALUATE SOLUTIONS
      DO 600 I=1,NN1
      SUMM=0.0D00
      DO 700 K=1,I-1
      SUMM=SUMM+A(I,K)*RHS1(K)
  700 CONTINUE
      RHS1(I)=(RHS1(I)-SUMM)/A(I,I)
  600 CONTINUE
      DO 900 I=NN1,1,-1
      SUMM=0.0D00
      DO 800 K=I+1,NN1
      SUMM=SUMM+A(I,K)*RHS1(K)
  800 CONTINUE
      RHS1(I)=RHS1(I)-SUMM
  900 CONTINUE
      RETURN
      END
C=============================================================
      SUBROUTINE PHYGLOB
C THIS S/R CALCULATES POTENTIALS OF FIELD NODES
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/PI11/PI
      COMMON/XYZR1/XR(NN,NPNO),YR(NN,NPNO),ZR(NN,NPNO)
      COMMON/COLMN1/COLX(NN),COMX(NN),CONX(NN)
      COMMON/COLMN2/COLY(NN),COMY(NN),CONY(NN)
      COMMON/COLMN3/COLZ(NN),COMZ(NN),CONZ(NN)
      COMMON/XYZT1/XT(3),YT(3),ZT(3)
      COMMON/XYZ21R/XP,YP,ZP
      COMMON/XYZIE/XIE(NN),YIE(NN),ZIE(NN)
      COMMON/YMAX11/YMAX,YHSPAN,FRAC,SURFW
      COMMON/F123/F1,F2,F3
      COMMON/ST11/STIF(4)
      COMMON/ABC11/ALPHA,BETA,GAMA
      COMMON/BKARHS/A(NN1,NN1),RHS1(NN1),BA1(NN1,NN1),BAR(NN1)
      COMMON/XYZ21W/XPW,YPW,ZPW
      COMMON/F12W/F3W
      COMMON/NNNW/NODEW(NL,NWNO)
      COMMON/IBN/IB(NL),IBU(NL),ISBU(NN1),IBB(NN1)
```

```
      COMMON/SAML/SMALL
      COMMON/XYZRW/XW(NL,NWUNO),YW(NL,NWUNO),ZW(NL,NWUNO)
      COMMON/CWLMN1/CWLX(NL),CWMX(NL),CWNX(NL)
      COMMON/CWLMN2/CWLY(NL),CWMY(NL),CWNY(NL)
      COMMON/CWLMN3/CWLZ(NL),CWMZ(NL),CWNZ(NL)
      COMMON/QQ1/Q(NN1)
      COMMON/PG1/PHYG(NGLOB)
      COMMON/XYZG/X3(NGLOB),Y3(NGLOB),Z3(NGLOB)
      COMMON/NGL1/NGL(NGLOB),NGLT(NN1),NWAKE(NGLOB)
      COMMON/STGB/STIFB(NPNO)
      COMMON/STGW/STIFW(NWNO)
      COMMON/XYZGIE/X3G(NELEM),Y3G(NELEM),Z3G(NELEM),V3G(NELEM)
      COMMON/PS111/SIGMA(NELEM)
      COMMON/NNODE1/NODE(NELEM,NDPL)
      COMMON/KE123/KSU,KSL,KSUE(NSHO),KSLE(NSHO),KE(NELEM),KN(NGLOB)
      COMMON/CPSIUV/PSXE(NELEM),PSYE(NELEM),PSZE(NELEM)
      COMMON/SHOPUP/XCU(NSHO),YCU(NSHO),ZCU(NSHO),VSAU(NSHO)
      COMMON/SHOPLO/XCL(NSHO),YCL(NSHO),ZCL(NSHO),VSAL(NSHO)
C -------------------------------------------------------------
C DO 7001 IS REPEATED FOR EACH FIELD NODES
C NODES MEETING WING NODES (NGL(J).NE.0) ARE REMOVED FROM LOOP
      DO 7001 J=1,NGLOB

      IF(NGL(J).NE.0) GO TO 7001

C -------------------------------------------------------------
C NODES OF LOWER WING PLANE TAKE THE SAME POTENTIALS AS THEIR
C COUNTERPARTS OF THE UPPER WING PLANE EXCEPT FOR THE WAKE NODES
C (NWAKE.NE.0) WHICH HAVE DIFFERENT POTENTIALS (WAKE DISCONTINUITY)
      IF ( ( (J.GT.(MZ2*MY*MX)) .AND. (J.LT.((MZ2+1)*MY*MX+1)) ).AND.
     +(NWAKE(J).EQ.0) ) THEN
        PHYG(J)=PHYG(J-MX*MY)
      ELSE

C -------------------------------------------------------------
C CALCULATION OF THE EFFECT OF WING PANELS ON FIELD NODES (LOOP 7000)
C THE EFFECTS ARE NOTED IN "SUM".
C EVALUATION OF RADIUS "R" BETWEEN NODE AND PANEL CENTER
      SUM=0.0D00
      DO 7000 IE=1,NN
      R=DSQRT( (X3(J)-XIE(IE))*(X3(J)-XIE(IE))
     +       +(Y3(J)-YIE(IE))*(Y3(J)-YIE(IE))
     +       +(Z3(J)-ZIE(IE))*(Z3(J)-ZIE(IE)) )
      IF (R.GT.(FRAC*YMAX)) THEN

C -------------------------------------------------------------
C IF "R" IS GREATER THAN FRAC*YMAX, USE SIMPLIFIED VERSION
C THIS VERSION USES AN AVERAGE OF PANEL NODES POTENTIALS (QC)
        X2P=X3(J)-X(NODEB(IE,1))
        Y2P=Y3(J)-Y(NODEB(IE,1))
        Z2P=Z3(J)-Z(NODEB(IE,1))
```

```
      ZP=COLZ(IE)*X2P+COMZ(IE)*Y2P+CONZ(IE)*Z2P
      QC=(Q(NODEB(IE,1))+Q(NODEB(IE,2))+Q(NODEB(IE,3)))/3.0D00
      SUM=SUM - QC*ZP*XR(IE,2)*YR(IE,3)/2.0D00/(R**3.0D00)

C -------------------------------------------------------------
C REPEAT CALCULATIONS FOR OTHER SIDE OF WING PANEL (NEG. YIE & Y3)
      R=DSQRT( (X3(J)-XIE(IE))*(X3(J)-XIE(IE))
     +        +(Y3(J)+YIE(IE))*(Y3(J)+YIE(IE))
     +        +(Z3(J)-ZIE(IE))*(Z3(J)-ZIE(IE)) )

      Y2P=-Y3(J)-Y(NODEB(IE,1))
      ZP=COLZ(IE)*X2P+COMZ(IE)*Y2P+CONZ(IE)*Z2P
      SUM=SUM - QC*ZP*XR(IE,2)*YR(IE,3)/2.0D00/(R**3.0D00)
     ELSE
C -------------------------------------------------------------
C FOR SHORT RADII, USE COMPLETE VERSION (SAME AS IN S/R BUILDA)
      X2P=X3(J)-X(NODEB(IE,1))
      Y2P=Y3(J)-Y(NODEB(IE,1))
      Z2P=Z3(J)-Z(NODEB(IE,1))
      XP=COLX(IE)*X2P+COMX(IE)*Y2P+CONX(IE)*Z2P
      YP=COLY(IE)*X2P+COMY(IE)*Y2P+CONY(IE)*Z2P
      ZP=COLZ(IE)*X2P+COMZ(IE)*Y2P+CONZ(IE)*Z2P
      CALL FI(IE,J)
      STIFB(1)=(F3-(F1+XP*F3)/XR(IE,2)
    1 +(XR(IE,3)-XR(IE,2))*(F2+YP*F3)/XR(IE,2)/YR(IE,3))
      STIFB(2)=((F1+XP*F3)/XR(IE,2)
    1 -(F2+YP*F3)*XR(IE,3)/XR(IE,2)/YR(IE,3))
      STIFB(3)=(F2+YP*F3)/YR(IE,3)
C -------------------------------------------------------------
C ADD EFFECT OF 3 PANEL NODES TO "SUM"
      DO 19 K=1,3
      SUM=SUM+STIFB(K)*Q(NODEB(IE,K))
   19  CONTINUE

C -------------------------------------------------------------
C EFFECT OF OTHER SIDE: RE-EVALUATE "R" TO KNOW WHICH VERSION TO USE
      R=DSQRT( (X3(J)-XIE(IE))*(X3(J)-XIE(IE))
     +        +(Y3(J)+YIE(IE))*(Y3(J)+YIE(IE))
     +        +(Z3(J)-ZIE(IE))*(Z3(J)-ZIE(IE)) )
      IF(R.GT.(FRAC*YMAX)) THEN

C -------------------------------------------------------------
C "R" GREATER THAN FRAC*YMAX: USE SIMPLIFED VERSION
       QC=(Q(NODEB(IE,1))+Q(NODEB(IE,2))+Q(NODEB(IE,3)))/3.0D00
       Y2P=-Y3(J)-Y(NODEB(IE,1))
       ZP=COLZ(IE)*X2P+COMZ(IE)*Y2P+CONZ(IE)*Z2P
       SUM=SUM - QC*ZP*XR(IE,2)*YR(IE,3)/2.0D00/(R**3.0D00)
      ELSE
C -------------------------------------------------------------
C R SHORTER: USE LONGER VERSION (WITH NEG. Y3)
       Y2P=-Y3(J)-Y(NODEB(IE,1))
```

```
          XP=COLX(IE)*X2P+COMX(IE)*Y2P+CONX(IE)*Z2P
          YP=COLY(IE)*X2P+COMY(IE)*Y2P+CONY(IE)*Z2P
          ZP=COLZ(IE)*X2P+COMZ(IE)*Y2P+CONZ(IE)*Z2P
          CALL FI(IE,J)
          STIFB(1)=(F3-(F1+XP*F3)/XR(IE,2)
     1     +(XR(IE,3)-XR(IE,2))*(F2+YP*F3)/XR(IE,2)/YR(IE,3))
          STIFB(2)=((F1+XP*F3)/XR(IE,2)
     1     -(F2+YP*F3)*XR(IE,3)/XR(IE,2)/YR(IE,3))
          STIFB(3)=(F2+YP*F3)/YR(IE,3)
          DO 20 K=1,3
            SUM=SUM+STIFB(K)*Q(NODEB(IE,K))
   20     CONTINUE
        END IF
      END IF
 7000 CONTINUE

C -------------------------------------------------------------
C LOOP  8000: EFFECT OF WAKE PANELS ON FIELD NODE
C THIS LOOP IS IDENTICAL TO THE LOOP 8000 IN S/R BUILDA
C
      DO 8000 IEW=1,NL
      X2P=X3(J)-X(NODEW(IEW,1))
      Y2P=Y3(J)-Y(NODEW(IEW,1))
      Z2P=Z3(J)-Z(NODEW(IEW,1))
      XPW=CWLX(IEW)*X2P+CWMX(IEW)*Y2P+CWNX(IEW)*Z2P
      YPW=CWLY(IEW)*X2P+CWMY(IEW)*Y2P+CWNY(IEW)*Z2P
      ZPW=CWLZ(IEW)*X2P+CWMZ(IEW)*Y2P+CWNZ(IEW)*Z2P
      CALL FI3W(IEW,J)
      STIFW(1)=F3W/2.0D00
      STIFW(2)=STIFW(1)
      STIFW(3)=-STIFW(1)
      STIFW(4)=-STIFW(1)
C -------------------------------------------------------------
C EFFECT OF OTHER SIDE OF WING WAKE PANELS (NEG. Y3)
      Y2P=-Y3(J)-Y(NODEW(IEW,1))
      XPW=CWLX(IEW)*X2P+CWMX(IEW)*Y2P+CWNX(IEW)*Z2P
      YPW=CWLY(IEW)*X2P+CWMY(IEW)*Y2P+CWNY(IEW)*Z2P
      ZPW=CWLZ(IEW)*X2P+CWMZ(IEW)*Y2P+CWNZ(IEW)*Z2P
      CALL FI3W(IEW,J)
      STIFW(1)=STIFW(1)+F3W/2.0D00
      STIFW(2)=STIFW(2)+F3W/2.0D00
      STIFW(3)=STIFW(3)-F3W/2.0D00
      STIFW(4)=STIFW(4)-F3W/2.0D00
C -------------------------------------------------------------
C ADD 4 EFFECTS TO "SUM"
      DO 21 K=1,4
      SUM=SUM+STIFW(K)*Q(NODEW(IEW,K))
   21 CONTINUE
 8000 CONTINUE
C -------------------------------------------------------------
C ADD EFFECTS OF SUM & INCOMPRESSIBLE FREE STREAM FLOW TO NODE
```

```
 POTENTIAL
      PHYG(J)=(X3(J)*DCOS(ALPHA)+Y3(J)*DCOS(BETA)+Z3(J)*DCOS(GAMA))
     1+SUM/4.0D00/PI

C -------------------------------------------------------------
C CALCULATE EFFECT OF FIELD ELEMENTS:"SUMP1" VALUE
C EFFECT OF OTHER WING SIDE ELEMENTS (NEG. Y3G) INCLUDED
      SUMP1=0.0D00
      DO 689 IE=1,NELEM
      XRS=X3(J)-X3G(IE)
      YRS=Y3(J)-Y3G(IE)
      ZRS=Z3(J)-Z3G(IE)
      SUMP1=SUMP1-SIGMA(IE)*V3G(IE)/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
      YRS=Y3(J)+Y3G(IE)
      SUMP1=SUMP1-SIGMA(IE)*V3G(IE)/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
  689 CONTINUE

C -------------------------------------------------------------
C CALCULATE EFFECT OF SHOCK SURFACES
C LOOP 681 EVALUATES THE EFFECT OF UPPER SHOCKS (VARIABLE SUMU1)
C EFFECT OF OTHER WING SIDE SHOCKS (NEG. YCU) INCLUDED
      SUMU1=0.0D00
      DO 681 K=1,KSU
      IE=KSUE(K)
      XRS=X3(J)-XCU(K)
      YRS=Y3(J)-YCU(K)
      ZRS=Z3(J)-ZCU(K)
      SUMU1=SUMU1+(PSXE(IE)-PSXE(IE+MY-1))*VSAU(K)
     1/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
      YRS=Y3(J)+YCU(K)
      SUMU1=SUMU1+(PSXE(IE)-PSXE(IE+MY-1))*VSAU(K)
     1/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
  681 CONTINUE

C -------------------------------------------------------------
C LOOP 682 EVALUATES THE EFFECT OF LOWER SHOCKS (VARIABLE SUML1)
C EFFECT OF OTHER WING SIDE SHOCKS (NEG. YCL) INCLUDED
      SUML1=0.0D00
      DO 682 K=1,KSL
      IE=KSLE(K)
      XRS=X3(J)-XCL(K)
      YRS=Y3(J)-YCL(K)
      ZRS=Z3(J)-ZCL(K)
      SUML1=SUML1+(PSXE(IE)-PSXE(IE+MY-1))*VSAL(K)
     1/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
      YRS=Y3(J)+YCL(K)
      SUML1=SUML1+(PSXE(IE)-PSXE(IE+MY-1))*VSAL(K)
     1/DSQRT(XRS*XRS+YRS*YRS+ZRS*ZRS)
  682 CONTINUE

C -------------------------------------------------------------
```

```
C MODIFY NODE POTENTIAL "PHYG" TO INCLUDE EFFECT OF ELEMENTS & SHOCKS
      PHYG(J)=PHYG(J)+(SUMP1+SUMU1+SUML1)/4.0D00/PI

      END IF

 7001 CONTINUE

      RETURN
      END
C=============================================================
      SUBROUTINE UV
C THIS S/R CALCULATES THE VALUES OF SIGMA FOR EACH FIELD ELEMENT
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)

      COMMON/DNG1/DNGLO1(8,3)
      COMMON/PG1/PHYG(NGLOB)
      COMMON/CPSIUV/PSXE(NELEM),PSYE(NELEM),PSZE(NELEM)
      COMMON/PXYSS/PSY(NGLOB,NDPL),PSX(NGLOB,NDPL),PSZ(NGLOB,NDPL)
      COMMON/PS111/SIGMA(NELEM)
      COMMON/NNODE1/NODE(NELEM,NDPL)
      COMMON/XYZG/X3(NGLOB),Y3(NGLOB),Z3(NGLOB)
      COMMON/XYZSS4/PSXXE(NELEM),PSXYE(NELEM),PSXZE(NELEM)
      COMMON/XYZSS5/PSYXE(NELEM),PSYYE(NELEM),PSYZE(NELEM)
      COMMON/XYZSS6/PSZXE(NELEM),PSZYE(NELEM),PSZZE(NELEM)
      COMMON/DM010/DM0,DMM
      COMMON/DMGVV/DMG(NELEM),VV2(NELEM)
      COMMON/KE123/KSU,KSL,KSUE(NSHO),KSLE(NSHO),KE(NELEM),KN(NGLOB)
      COMMON/UVW11/U1(NGLOB),V1(NGLOB),W1(NGLOB)
      COMMON/SHOPUP/XCU(NSHO),YCU(NSHO),ZCU(NSHO),VSAU(NSHO)
      COMMON/SHOPLO/XCL(NSHO),YCL(NSHO),ZCL(NSHO),VSAL(NSHO)
      REAL PSXX(8),PSYY(8),PSZZ(8),PSXY(8),PSYX(8),PSXZ(8),PSZX(8),
     +     PSYZ(8),PSZY(8)

C ------------------------------------------------------------
C INITIALIZE VALUES OF PSX,PSY & PSZ AND ALSO KN (NODES PART OF SHOCKS)
      DO 654 I=1,NGLOB
      KN(I)=0
      DO 599 J=1,8
      PSX(I,J)=0.0D00
      PSY(I,J)=0.0D00
      PSZ(I,J)=0.0D00
  599 CONTINUE
  654 CONTINUE

C ------------------------------------------------------------
C LOOP 581 CALCULATES THE FIRST DERIVATIVES PSXE,PSYE & PSZE OF ELEMENTS
```

```
C SIGMA VALUES ARE INITIALIZED, AS ARE VALUES OF "KE"(SHOCK ELEMENTS).
C IN LOOP 301, FOR EACH NODE (8) OF THE ELEMENT, THE DERIVATIVES PSX,PSY
C & PSZ ARE EVALUATED. THESE VALUES ARE BASED ON THE POTENTIALS OF THE
C 8 NODES OF THE ELEMENT ONLY. A FINITE ELEMENT GRID IS USED (DNGLO1
C TERMS ARE OBTAINED BY S/R SHAP1).

      DO 581 IE=1,NELEM
      SIGMA(IE)=0.0D00
      KE(IE)=0
      DO 301 I=1,8
      CALL SHAP1(I,IE)
      PSX(NODE(IE,I),I)=
     1 DNGLO1(1,1)*PHYG(NODE(IE,1))+DNGLO1(2,1)*PHYG(NODE(IE,2))
     2+DNGLO1(3,1)*PHYG(NODE(IE,3))+DNGLO1(4,1)*PHYG(NODE(IE,4))
     3+DNGLO1(5,1)*PHYG(NODE(IE,5))+DNGLO1(6,1)*PHYG(NODE(IE,6))
     4+DNGLO1(7,1)*PHYG(NODE(IE,7))+DNGLO1(8,1)*PHYG(NODE(IE,8))
      PSY(NODE(IE,I),I)=
     1 DNGLO1(1,2)*PHYG(NODE(IE,1))+DNGLO1(2,2)*PHYG(NODE(IE,2))
     2+DNGLO1(3,2)*PHYG(NODE(IE,3))+DNGLO1(4,2)*PHYG(NODE(IE,4))
     3+DNGLO1(5,2)*PHYG(NODE(IE,5))+DNGLO1(6,2)*PHYG(NODE(IE,6))
     4+DNGLO1(7,2)*PHYG(NODE(IE,7))+DNGLO1(8,2)*PHYG(NODE(IE,8))
      PSZ(NODE(IE,I),I)=
     1 DNGLO1(1,3)*PHYG(NODE(IE,1))+DNGLO1(2,3)*PHYG(NODE(IE,2))
     2+DNGLO1(3,3)*PHYG(NODE(IE,3))+DNGLO1(4,3)*PHYG(NODE(IE,4))
     3+DNGLO1(5,3)*PHYG(NODE(IE,5))+DNGLO1(6,3)*PHYG(NODE(IE,6))
     4+DNGLO1(7,3)*PHYG(NODE(IE,7))+DNGLO1(8,3)*PHYG(NODE(IE,8))
  301 CONTINUE

C -------------------------------------------------------------
C THEN, THE VALUES OF PSXE, PSYE & PSZE FOR THE ELEMENT ARE CALCULATED
C BY MAKING THE AVERAGE OF THE 8 NODES DERIVATIVES PSX, PSY & PSZ
      PSXE(IE)=(
     1 PSX(NODE(IE,1),1)+PSX(NODE(IE,2),2)
     2+PSX(NODE(IE,3),3)+PSX(NODE(IE,4),4)
     3+PSX(NODE(IE,5),5)+PSX(NODE(IE,6),6)
     4+PSX(NODE(IE,7),7)+PSX(NODE(IE,8),8)
     5)/8.0D00
      PSYE(IE)=(
     1 PSY(NODE(IE,1),1)+PSY(NODE(IE,2),2)
     2+PSY(NODE(IE,3),3)+PSY(NODE(IE,4),4)
     3+PSY(NODE(IE,5),5)+PSY(NODE(IE,6),6)
     4+PSY(NODE(IE,7),7)+PSY(NODE(IE,8),8)
     5)/8.0D00
      PSZE(IE)=(
     1 PSZ(NODE(IE,1),1)+PSZ(NODE(IE,2),2)
     2+PSZ(NODE(IE,3),3)+PSZ(NODE(IE,4),4)
     3+PSZ(NODE(IE,5),5)+PSZ(NODE(IE,6),6)
     4+PSZ(NODE(IE,7),7)+PSZ(NODE(IE,8),8)
     5)/8.0D00
C -------------------------------------------------------------
C CALCULATE THE SQUARE OF THE SPEED (VV2) AND THE ELEMENT MACH # (DMG).
```

```
C VV2 IS THE SQUARE OF THE SUM OF POTENTIALS, DMG USES A COMPRESSIBLE
C EQUATION.
C WHEN THERE IS A DISCREPANCY IN THE RESULTS, VV2 WILL BE TOO HIGH AND
C THIS WILL CAUSE A SQRT OF A NEGATIVE NUMBER IN THE DMG EQUATION.
C THIS CAN HAPPEN WHEN "FRAC" IS TOO SMALL OR (AT THIS STAGE) WHEN
C ALPHA IS TOO HIGH. A MESSAGE IS PRINTED MENTIONING THE ELEMENT #.
C PROGRAM CAN BE STOPPED AT THIS STAGE.
      VV2(IE)=PSXE(IE)*PSXE(IE)+PSYE(IE)*PSYE(IE)
     1+PSZE(IE)*PSZE(IE)
      IF (VV2(IE).GE.(1.0D00/(0.2D00*DM0))) THEN
        WRITE(6,*)'************************************************'
        WRITE(6,*)'VV2=',VV2(IE),' FOR ELEMENT ',IE,' IS TOO HIGH'
        WRITE(6,*)'A DISCREPANCY IN THE RESULTS CAUSES AN ERROR'
C       WRITE(6,*)'THE PROGRAM IS STOPPED'
C       STOP
      END IF
      DM2=DM0*VV2(IE)
      DMG(IE)=DSQRT(DM2/(1.0D00-0.2D00*DM2))
  581 CONTINUE

C -------------------------------------------------------
C SHOCK CAPTURING LOOPS
C LOOP 123 INITIALIZES THE SHOCK SURFACES VECTORS KSUE & KSLE
C LOOP 121 FINDS THE "UPPER" SHOCKS: THERE IS A SHOCK WHEN A ELEMENT
C MACH NUMBER IS .GE.1.0 AND THAT THE FOLLOWING ELEMENT (ALONG X AXIS)
C MACH NUMBER IS .LT.1.0. KSU IS INCREMENTED WITH EACH NEW SHOCK AND IS
C LIMITED BY INPUT PARAMETER NSHO (MESSAGE). THE SHOCK SURFACE IS THE
C MEETING SIDE OF THE TWO VOLUME ELEMENTS.
      DO 123 I=1,NSHO
      KSUE(I)=0
      KSLE(I)=0
  123 CONTINUE

      KSU=0
      DO 121 IE=1,NELEM/2
      IF(DMG(IE).GT.1.0D00 )THEN
      IF(DMG(IE+MY-1).LT.1.0D00 )THEN
      KSU=KSU+1
      IF(KSU.GT.NSHO) WRITE(6,*)'BIGGER NUMBER OF UPPER SHOCKS
     +    THAN "NSHO"=',NSHO
C -------------------------------------------------------------
C MODIFY VECTORS "KSUE","KE","KN" (4 NODES PART OF SHOCK SURFACE), XCU,
C YCU & ZCU (CENTER OF SHOCK) AND VSAU (SHOCK SURFACE)
      KSUE(KSU)=IE
      KE(IE+MY-1)=IE+MY-1
      KN(NODE(IE,2))=NODE(IE,2)
      KN(NODE(IE,3))=NODE(IE,3)
      KN(NODE(IE,6))=NODE(IE,6)
      KN(NODE(IE,7))=NODE(IE,7)
      XCU(KSU)=(X3(NODE(IE,2))+X3(NODE(IE,3))+X3(NODE(IE,6))
     1+X3(NODE(IE,7)))/4.0D00
```

```
      YCU(KSU)=(Y3(NODE(IE,2))+Y3(NODE(IE,3))+Y3(NODE(IE,6))
     1+Y3(NODE(IE,7)))/4.0D00
      ZCU(KSU)=(Z3(NODE(IE,2))+Z3(NODE(IE,3))+Z3(NODE(IE,6))
     1+Z3(NODE(IE,7)))/4.0D00
      XSA=X3(NODE(IE,3))-X3(NODE(IE,2))
      YSA=Y3(NODE(IE,3))-Y3(NODE(IE,2))
      ZSA=Z3(NODE(IE,6))-Z3(NODE(IE,2))
      VSAU(KSU)=DSQRT(XSA*XSA+YSA*YSA)*ZSA
      END IF
      END IF
  121 CONTINUE

C -------------------------------------------------------------
C LOOP 122 FINDS THE "LOWER" SHOCKS, AS IN LOOP 121
      KSL=0
      DO 122 IE=NELEM/2+1,NELEM
      IF(DMG(IE).GT.1.0D00 )THEN
      IF(DMG(IE+MY-1).LT.1.0D00 )THEN
      KSL=KSL+1
      IF(KSL.GT.NSHO) WRITE(6,*)'BIGGER NUMBER OF LOWER SHOCKS
     +    THAN "NSHO"=',NSHO
C -------------------------------------------------------------
C MODIFY VECTORS "KSLE","KE", "KN" (4 NODES PART OF SHOCK SURFACE), XCL,
C YCL & ZCL (CENTER OF SHOCK) AND VSAL (SHOCK SURFACE)
      KSLE(KSL)=IE
      KE(IE+MY-1)=IE+MY-1
      KN(NODE(IE,2))=NODE(IE,2)
      KN(NODE(IE,3))=NODE(IE,3)
      KN(NODE(IE,6))=NODE(IE,6)
      KN(NODE(IE,7))=NODE(IE,7)
      XCL(KSL)=(X3(NODE(IE,2))+X3(NODE(IE,3))+X3(NODE(IE,6))
     1+X3(NODE(IE,7)))/4.0D00
      YCL(KSL)=(Y3(NODE(IE,2))+Y3(NODE(IE,3))+Y3(NODE(IE,6))
     1+Y3(NODE(IE,7)))/4.0D00
      ZCL(KSL)=(Z3(NODE(IE,2))+Z3(NODE(IE,3))+Z3(NODE(IE,6))
     1+Z3(NODE(IE,7)))/4.0D00
      XSA=X3(NODE(IE,3))-X3(NODE(IE,2))
      YSA=Y3(NODE(IE,3))-Y3(NODE(IE,2))
      ZSA=Z3(NODE(IE,6))-Z3(NODE(IE,2))
      VSAL(KSL)=DSQRT(XSA*XSA+YSA*YSA)*ZSA
      END IF
      END IF
  122 CONTINUE

C -------------------------------------------------------------
C LOOP 600 CALCULATES THE DERIVATIVES PSX,PSY & PSZ FOR THE FIELD NODES
C THESE DERIVATIVES WERE CALCULATED IN LOOP 301 BUT THEY WERE THE CON-
C TRIBUTION ONLY OF THE NODE TO THE ELEMENT DERIVATIVE. THIS LOOP MAKES
C THE AVERAGE OF THE DERIVATIVES FOR THE GENERAL CASE (8 VALUES).
C THE FIRST "IF" IS FOR NODES PART OF SHOCK SURFACES (KN.EQ.I):
C DERIVATIVES ARE CALCULATED BEFORE (PSX,PSY,PSZ) AND AFTER (U1,V1,W1)
```

```
C THE SHOCK; THEN, THERE IS AN AVERAGE OF ONLY 4 VALUES. THE AVERAGE
C IS STORED IN THE 1ST POSITION OF THE ARRAY PSX(I,1),....
      DO 600 I=1,NGLOB
      IF(KN(I).EQ.I)THEN
         U1(I)=(PSX(I,1)+PSX(I,4)+PSX(I,5)+PSX(I,8))/4.0D00
         V1(I)=(PSY(I,1)+PSY(I,4)+PSY(I,5)+PSY(I,8))/4.0D00
         W1(I)=(PSZ(I,1)+PSZ(I,4)+PSZ(I,5)+PSZ(I,8))/4.0D00
         PSX(I,1)=(PSX(I,2)+PSX(I,3)+PSX(I,6)+PSX(I,7))/4.0D00
         PSY(I,1)=(PSY(I,2)+PSY(I,3)+PSY(I,6)+PSY(I,7))/4.0D00
         PSZ(I,1)=(PSZ(I,2)+PSZ(I,3)+PSZ(I,6)+PSZ(I,7))/4.0D00
      ELSE
         SUMX=0.0D00
         SUMY=0.0D00
         SUMZ=0.0D00
         DO 601 J=1,8
         SUMX=SUMX+PSX(I,J)
         SUMY=SUMY+PSY(I,J)
         SUMZ=SUMZ+PSZ(I,J)
  601    CONTINUE
         PSX(I,1)=SUMX/8.0D00
         PSY(I,1)=SUMY/8.0D00
         PSZ(I,1)=SUMZ/8.0D00
      END IF
  600 CONTINUE

C -------------------------------------------------------------
C THE NEXT LOOPS ARE FOR NODES WHICH DID NOT APPLY TO THE PREVIOUS
C CASE. THEY ARE NODES ON THE 6 SIDES OF THE FIELD GRID AND ON THE
C MEETING LOWER & UPPER WING PLANES. THEY ARE JONCTION POINTS FOR
C LESS THAN 8 VOLUME ELEMENTS. UPPER AND LOWER WING PLANE VALUES ARE
C DIVIDED BY TWO BECAUSE OF THE DOUBLE VALUE OF POTENTIAL AT THE SAME
C LOCATION (THIS WOULD CAUSE DISCREPANCY IN POTENTIAL DERIVATIVES)
C -------------------------------------------------------------
C LOOP 23 IS FOR THE 8 CORNERS OF THE UPPER FIELD GRID
C LOOP 33 IS FOR THE 8 CORNERS OF THE LOWER FIELD GRID
      DO 23 I=1,MZ2,MZ2-1
      DO 24 J=1,MX,MX-1
      DO 25 K=1,MY,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*8.0D00
      PSY(II,1)=PSY(II,1)*8.0D00
      PSZ(II,1)=PSZ(II,1)*8.0D00
   25 CONTINUE
   24 CONTINUE
   23 CONTINUE

      DO 33 I=MZ2+1,MZ,MZ2-1
      DO 34 J=1,MX,MX-1
      DO 35 K=1,MY,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*8.0D00
```

```
      PSY(II,1)=PSY(II,1)*8.0D00
      PSZ(II,1)=PSZ(II,1)*8.0D00
   35 CONTINUE
   34 CONTINUE
   33 CONTINUE

C ------------------------------------------------------------
C LOOP 43 IS FOR THE EDGES ALONG Y AXIS OF UPPER FIELD GRID
C LOOP 53 IS FOR THE EDGES ALONG Y AXIS OF LOWER FIELD GRID

      DO 43 I=1,MZ2,MZ2-1
      DO 44 J=1,MX,MX-1
      DO 45 K=2,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*4.0D00
      PSY(II,1)=PSY(II,1)*4.0D00
      PSZ(II,1)=PSZ(II,1)*4.0D00
   45 CONTINUE
   44 CONTINUE
   43 CONTINUE

      DO 53 I=MZ2+1,MZ,MZ2-1
      DO 54 J=1,MX,MX-1
      DO 55 K=2,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*4.0D00
      PSY(II,1)=PSY(II,1)*4.0D00
      PSZ(II,1)=PSZ(II,1)*4.0D00
   55 CONTINUE
   54 CONTINUE
   53 CONTINUE

C ------------------------------------------------------------
C LOOP 63 IS FOR THE EDGES ALONG X AXIS FOR UPPER FIELD GRID
C LOOP 73 IS FOR THE EDGES ALONG X AXIS FOR LOWER FIELD GRID
C THERE IS A "IF" FOR NODES FOLLOWING A SHOCK SURFACE (USE U1,V1,W1)
      DO 63 I=1,MZ2,MZ2-1
      DO 64 J=2,MX-1
      DO 65 K=1,MY,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*4.0D00
      PSY(II,1)=PSY(II,1)*4.0D00
      PSZ(II,1)=PSZ(II,1)*4.0D00
      IF(KN(II).EQ.II)THEN
      U1(II)=U1(II)*4.0D00
      V1(II)=V1(II)*4.0D00
      W1(II)=W1(II)*4.0D00
      END IF
   65 CONTINUE
   64 CONTINUE
   63 CONTINUE
```

```
      DO 73 I=MZ2+1,MZ,MZ2-1
      DO 74 J=2,MX-1
      DO 75 K=1,MY,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*4.0D00
      PSY(II,1)=PSY(II,1)*4.0D00
      PSZ(II,1)=PSZ(II,1)*4.0D00
      IF(KN(II).EQ.II)THEN
      U1(II)=U1(II)*4.0D00
      V1(II)=V1(II)*4.0D00
      W1(II)=W1(II)*4.0D00
      END IF
   75 CONTINUE
   74 CONTINUE
   73 CONTINUE

C ---------------------------------------------------------------
C LOOP 263 IS FOR EDGES ALONG Z AXIS FOR UPPER FIELD GRID
C LOOP 363 IS FOR EDGES ALONG Z AXIS FOR LOWER FIELD GRID

      DO 263 I=2,MZ2-1
      DO 264 J=1,MX,MX-1
      DO 265 K=1,MY,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*4.0D00
      PSY(II,1)=PSY(II,1)*4.0D00
      PSZ(II,1)=PSZ(II,1)*4.0D00
   265 CONTINUE
   264 CONTINUE
   263 CONTINUE

      DO 363 I=MZ2+2,MZ-1
      DO 364 J=1,MX,MX-1
      DO 365 K=1,MY,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*4.0D00
      PSY(II,1)=PSY(II,1)*4.0D00
      PSZ(II,1)=PSZ(II,1)*4.0D00
   365 CONTINUE
   364 CONTINUE
   363 CONTINUE

C ---------------------------------------------------------------
C LOOP 83 IS FOR POINTS OF XY UPPER AND LOWER SIDES,UPPER FIELD GRID
C LOOP 93 IS FOR POINTS OF XY UPPER AND LOWER SIDES,LOWER FIELD GRID
C THUS, NODES PART OF MEETING WING PLANES ARE INCLUDED
C SPECIAL CASES FOR NODES PART OF SHOCK SURFACES

      DO 83 I=1,MZ2,MZ2-1
      DO 84 J=2,MX-1
      DO 85 K=2,MY-1
```

```
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*2.0D00
      PSY(II,1)=PSY(II,1)*2.0D00
      PSZ(II,1)=PSZ(II,1)*2.0D00
      IF(KN(II).EQ.II)THEN
      U1(II)=U1(II)*2.0D00
      V1(II)=V1(II)*2.0D00
      W1(II)=W1(II)*2.0D00
      END IF
   85 CONTINUE
   84 CONTINUE
   83 CONTINUE

      DO 93 I=MZ2+1,MZ,MZ2-1
      DO 94 J=2,MX-1
      DO 95 K=2,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*2.0D00
      PSY(II,1)=PSY(II,1)*2.0D00
      PSZ(II,1)=PSZ(II,1)*2.0D00
      IF(KN(II).EQ.II)THEN
      U1(II)=U1(II)*2.0D00
      V1(II)=V1(II)*2.0D00
      W1(II)=W1(II)*2.0D00
      END IF
   95 CONTINUE
   94 CONTINUE
   93 CONTINUE

C -----------------------------------------------------------
C LOOP 113 IS FOR POINTS OF 2 YZ SIDES, UPPER FIELD GRID
C LOOP 213 IS FOR POINTS OF 2 YZ SIDES, LOWER FIELD GRID

      DO 113 I=2,MZ2-1
      DO 114 J=1,MX,MX-1
      DO 115 K=2,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*2.0D00
      PSY(II,1)=PSY(II,1)*2.0D00
      PSZ(II,1)=PSZ(II,1)*2.0D00
  115 CONTINUE
  114 CONTINUE
  113 CONTINUE

      DO 213 I=MZ2+2,MZ-1
      DO 214 J=1,MX,MX-1
      DO 215 K=2,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*2.0D00
      PSY(II,1)=PSY(II,1)*2.0D00
      PSZ(II,1)=PSZ(II,1)*2.0D00
```

```
  215 CONTINUE
  214 CONTINUE
  213 CONTINUE

C ------------------------------------------------------------
C LOOP 313 IS FOR NODES OF 2 XZ SIDES, UPPER FIELD GRID
C LOOP 413 IS FOR NODES OF 2 XZ SIDES, LOWER FIELD GRID
C SPECIAL CASES FOR NODES PART OF SHOCKS

      DO 313 I=2,MZ2-1
      DO 314 J=2,MX-1
      DO 315 K=1,MY,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*2.0D00
      PSY(II,1)=PSY(II,1)*2.0D00
      PSZ(II,1)=PSZ(II,1)*2.0D00
      IF(KN(II).EQ.II)THEN
      U1(II)=U1(II)*2.0D00
      V1(II)=V1(II)*2.0D00
      W1(II)=W1(II)*2.0D00
      END IF
  315 CONTINUE
  314 CONTINUE
  313 CONTINUE

      DO 413 I=MZ2+2,MZ-1
      DO 414 J=2,MX-1
      DO 415 K=1,MY,MY-1
      II=K+(J-1)*MY+(I-1)*MY*MX
      PSX(II,1)=PSX(II,1)*2.0D00
      PSY(II,1)=PSY(II,1)*2.0D00
      PSZ(II,1)=PSZ(II,1)*2.0D00
      IF(KN(II).EQ.II)THEN
      U1(II)=U1(II)*2.0D00
      V1(II)=V1(II)*2.0D00
      W1(II)=W1(II)*2.0D00
      END IF
  415 CONTINUE
  414 CONTINUE
  413 CONTINUE

C ------------------------------------------------------------
C FOR ALL NODES NOT PART OF SHOCK SURFACES, EQUAL U1 WITH PSX, ETC..
      DO 145 I=1,NGLOB
      IF(KN(I).NE.I)THEN
      U1(I)=PSX(I,1)
      V1(I)=PSY(I,1)
      W1(I)=PSZ(I,1)
      END IF
  145 CONTINUE
```

```
C -------------------------------------------------------------
C CORRECTIONS FOR NODES ON SIDES OF FIELD GRID TERMINATED
C -------------------------------------------------------------
C INITIALIZE SECOND DERIVATIVES VECTORS FOR NODES

      DO 1654 I=1,8
      PSXX(I)=0.0D00
      PSXY(I)=0.0D00
      PSXZ(I)=0.0D00
      PSYX(I)=0.0D00
      PSYY(I)=0.0D00
      PSYZ(I)=0.0D00
      PSZX(I)=0.0D00
      PSZY(I)=0.0D00
      PSZZ(I)=0.0D00
 1654 CONTINUE

C -------------------------------------------------------------
C CALCULATION OF ELEMENT SECOND DERIVATIVES
C THIS WORKS AS IN LOOP 581 FOR THE FIRST ELEMENT DERIVATIVES.
C THE 9 SECOND DERIVATIVES ARE CALCULATED AT EACH NODE OF THE ELEMENT
C USING THE FIRST DERIVATIVES OF THE 8 ELEMENT NODES WITH A FINITE
C ELEMENT GRID (DNGLO1 TERMS, S/R SHAP1). AN AVERAGE OF THESE NODE
C VALUES IS THEN MADE RESULTING IN THE ELEMENT VALUES.
C THERE ARE TWO CASES TO ACCOUNT FOR SHOCK SURFACES:
      DO 1581 IE=1,NELEM
      IF(KE(IE).EQ.IE)THEN
C -------------------------------------------------------------
C FIRST CASE: ELEMENT FOLLOWING A SHOCK SURFACE:USE U1,V1,W1 DERIVATIVES
      DO 1371 I=1,8
      CALL SHAP1(I,IE)
      PSXX(I)=
     1 DNGLO1(1,1)*U1(NODE(IE,1))+DNGLO1(2,1)*U1(NODE(IE,2))
     2+DNGLO1(3,1)*U1(NODE(IE,3))+DNGLO1(4,1)*U1(NODE(IE,4))
     3+DNGLO1(5,1)*U1(NODE(IE,5))+DNGLO1(6,1)*U1(NODE(IE,6))
     4+DNGLO1(7,1)*U1(NODE(IE,7))+DNGLO1(8,1)*U1(NODE(IE,8))
      PSXY(I)=
     1 DNGLO1(1,2)*U1(NODE(IE,1))+DNGLO1(2,2)*U1(NODE(IE,2))
     2+DNGLO1(3,2)*U1(NODE(IE,3))+DNGLO1(4,2)*U1(NODE(IE,4))
     3+DNGLO1(5,2)*U1(NODE(IE,5))+DNGLO1(6,2)*U1(NODE(IE,6))
     4+DNGLO1(7,2)*U1(NODE(IE,7))+DNGLO1(8,2)*U1(NODE(IE,8))
      PSXZ(I)=
     1 DNGLO1(1,3)*U1(NODE(IE,1))+DNGLO1(2,3)*U1(NODE(IE,2))
     2+DNGLO1(3,3)*U1(NODE(IE,3))+DNGLO1(4,3)*U1(NODE(IE,4))
     3+DNGLO1(5,3)*U1(NODE(IE,5))+DNGLO1(6,3)*U1(NODE(IE,6))
     4+DNGLO1(7,3)*U1(NODE(IE,7))+DNGLO1(8,3)*U1(NODE(IE,8))
      PSYX(I)=
     1 DNGLO1(1,1)*V1(NODE(IE,1))+DNGLO1(2,1)*V1(NODE(IE,2))
     2+DNGLO1(3,1)*V1(NODE(IE,3))+DNGLO1(4,1)*V1(NODE(IE,4))
     3+DNGLO1(5,1)*V1(NODE(IE,5))+DNGLO1(6,1)*V1(NODE(IE,6))
     4+DNGLO1(7,1)*V1(NODE(IE,7))+DNGLO1(8,1)*V1(NODE(IE,8))
```

```
      PSYY(I)=
     1 DNGLO1(1,2)*V1(NODE(IE,1))+DNGLO1(2,2)*V1(NODE(IE,2))
     2+DNGLO1(3,2)*V1(NODE(IE,3))+DNGLO1(4,2)*V1(NODE(IE,4))
     3+DNGLO1(5,2)*V1(NODE(IE,5))+DNGLO1(6,2)*V1(NODE(IE,6))
     4+DNGLO1(7,2)*V1(NODE(IE,7))+DNGLO1(8,2)*V1(NODE(IE,8))
      PSYZ(I)=
     1 DNGLO1(1,3)*V1(NODE(IE,1))+DNGLO1(2,3)*V1(NODE(IE,2))
     2+DNGLO1(3,3)*V1(NODE(IE,3))+DNGLO1(4,3)*V1(NODE(IE,4))
     3+DNGLO1(5,3)*V1(NODE(IE,5))+DNGLO1(6,3)*V1(NODE(IE,6))
     4+DNGLO1(7,3)*V1(NODE(IE,7))+DNGLO1(8,3)*V1(NODE(IE,8))
      PSZX(I)=
     1 DNGLO1(1,1)*W1(NODE(IE,1))+DNGLO1(2,1)*W1(NODE(IE,2))
     2+DNGLO1(3,1)*W1(NODE(IE,3))+DNGLO1(4,1)*W1(NODE(IE,4))
     3+DNGLO1(5,1)*W1(NODE(IE,5))+DNGLO1(6,1)*W1(NODE(IE,6))
     4+DNGLO1(7,1)*W1(NODE(IE,7))+DNGLO1(8,1)*W1(NODE(IE,8))
      PSZY(I)=
     1 DNGLO1(1,2)*W1(NODE(IE,1))+DNGLO1(2,2)*W1(NODE(IE,2))
     2+DNGLO1(3,2)*W1(NODE(IE,3))+DNGLO1(4,2)*W1(NODE(IE,4))
     3+DNGLO1(5,2)*W1(NODE(IE,5))+DNGLO1(6,2)*W1(NODE(IE,6))
     4+DNGLO1(7,2)*W1(NODE(IE,7))+DNGLO1(8,2)*W1(NODE(IE,8))
      PSZZ(I)=
     1 DNGLO1(1,3)*W1(NODE(IE,1))+DNGLO1(2,3)*W1(NODE(IE,2))
     2+DNGLO1(3,3)*W1(NODE(IE,3))+DNGLO1(4,3)*W1(NODE(IE,4))
     3+DNGLO1(5,3)*W1(NODE(IE,5))+DNGLO1(6,3)*W1(NODE(IE,6))
     4+DNGLO1(7,3)*W1(NODE(IE,7))+DNGLO1(8,3)*W1(NODE(IE,8))
 1371 CONTINUE
      ELSE
C -------------------------------------------------------------
C 2ND CASE: ELEMENTS THAT PRECEDE A SHOCK SURFACE AND THAT ARE
C NOT PART OF A SHOCK SURFACE. PSX,PSY & PSZ VALUES HAVE BEEN
C CALCULATED ACCORDINGLY
      DO 1301 I=1,8
      CALL SHAP1(I,IE)
      PSXX(I)=
     1 DNGLO1(1,1)*PSX(NODE(IE,1),1)+DNGLO1(2,1)*PSX(NODE(IE,2),1)
     2+DNGLO1(3,1)*PSX(NODE(IE,3),1)+DNGLO1(4,1)*PSX(NODE(IE,4),1)
     3+DNGLO1(5,1)*PSX(NODE(IE,5),1)+DNGLO1(6,1)*PSX(NODE(IE,6),1)
     4+DNGLO1(7,1)*PSX(NODE(IE,7),1)+DNGLO1(8,1)*PSX(NODE(IE,8),1)
      PSXY(I)=
     1 DNGLO1(1,2)*PSX(NODE(IE,1),1)+DNGLO1(2,2)*PSX(NODE(IE,2),1)
     2+DNGLO1(3,2)*PSX(NODE(IE,3),1)+DNGLO1(4,2)*PSX(NODE(IE,4),1)
     3+DNGLO1(5,2)*PSX(NODE(IE,5),1)+DNGLO1(6,2)*PSX(NODE(IE,6),1)
     4+DNGLO1(7,2)*PSX(NODE(IE,7),1)+DNGLO1(8,2)*PSX(NODE(IE,8),1)
      PSXZ(I)=
     1 DNGLO1(1,3)*PSX(NODE(IE,1),1)+DNGLO1(2,3)*PSX(NODE(IE,2),1)
     2+DNGLO1(3,3)*PSX(NODE(IE,3),1)+DNGLO1(4,3)*PSX(NODE(IE,4),1)
     3+DNGLO1(5,3)*PSX(NODE(IE,5),1)+DNGLO1(6,3)*PSX(NODE(IE,6),1)
     4+DNGLO1(7,3)*PSX(NODE(IE,7),1)+DNGLO1(8,3)*PSX(NODE(IE,8),1)
      PSYX(I)=
     1 DNGLO1(1,1)*PSY(NODE(IE,1),1)+DNGLO1(2,1)*PSY(NODE(IE,2),1)
     2+DNGLO1(3,1)*PSY(NODE(IE,3),1)+DNGLO1(4,1)*PSY(NODE(IE,4),1)
```

```
     3+DNGLO1(5,1)*PSY(NODE(IE,5),1)+DNGLO1(6,1)*PSY(NODE(IE,6),1)
     4+DNGLO1(7,1)*PSY(NODE(IE,7),1)+DNGLO1(8,1)*PSY(NODE(IE,8),1)
      PSYY(I)=
     1 DNGLO1(1,2)*PSY(NODE(IE,1),1)+DNGLO1(2,2)*PSY(NODE(IE,2),1)
     2+DNGLO1(3,2)*PSY(NODE(IE,3),1)+DNGLO1(4,2)*PSY(NODE(IE,4),1)
     3+DNGLO1(5,2)*PSY(NODE(IE,5),1)+DNGLO1(6,2)*PSY(NODE(IE,6),1)
     4+DNGLO1(7,2)*PSY(NODE(IE,7),1)+DNGLO1(8,2)*PSY(NODE(IE,8),1)
      PSYZ(I)=
     1 DNGLO1(1,3)*PSY(NODE(IE,1),1)+DNGLO1(2,3)*PSY(NODE(IE,2),1)
     2+DNGLO1(3,3)*PSY(NODE(IE,3),1)+DNGLO1(4,3)*PSY(NODE(IE,4),1)
     3+DNGLO1(5,3)*PSY(NODE(IE,5),1)+DNGLO1(6,3)*PSY(NODE(IE,6),1)
     4+DNGLO1(7,3)*PSY(NODE(IE,7),1)+DNGLO1(8,3)*PSY(NODE(IE,8),1)
      PSZX(I)=
     1 DNGLO1(1,1)*PSZ(NODE(IE,1),1)+DNGLO1(2,1)*PSZ(NODE(IE,2),1)
     2+DNGLO1(3,1)*PSZ(NODE(IE,3),1)+DNGLO1(4,1)*PSZ(NODE(IE,4),1)
     3+DNGLO1(5,1)*PSZ(NODE(IE,5),1)+DNGLO1(6,1)*PSZ(NODE(IE,6),1)
     4+DNGLO1(7,1)*PSZ(NODE(IE,7),1)+DNGLO1(8,1)*PSZ(NODE(IE,8),1)
      PSZY(I)=
     1 DNGLO1(1,2)*PSZ(NODE(IE,1),1)+DNGLO1(2,2)*PSZ(NODE(IE,2),1)
     2+DNGLO1(3,2)*PSZ(NODE(IE,3),1)+DNGLO1(4,2)*PSZ(NODE(IE,4),1)
     3+DNGLO1(5,2)*PSZ(NODE(IE,5),1)+DNGLO1(6,2)*PSZ(NODE(IE,6),1)
     4+DNGLO1(7,2)*PSZ(NODE(IE,7),1)+DNGLO1(8,2)*PSZ(NODE(IE,8),1)
      PSZZ(I)=
     1 DNGLO1(1,3)*PSZ(NODE(IE,1),1)+DNGLO1(2,3)*PSZ(NODE(IE,2),1)
     2+DNGLO1(3,3)*PSZ(NODE(IE,3),1)+DNGLO1(4,3)*PSZ(NODE(IE,4),1)
     3+DNGLO1(5,3)*PSZ(NODE(IE,5),1)+DNGLO1(6,3)*PSZ(NODE(IE,6),1)
     4+DNGLO1(7,3)*PSZ(NODE(IE,7),1)+DNGLO1(8,3)*PSZ(NODE(IE,8),1)
 1301 CONTINUE
      END IF
C -------------------------------------------------------------
C CALCULATE ELEMENT AVERAGES
      PSXXE(IE)=(
     1 PSXX(1)+PSXX(2)
     2+PSXX(3)+PSXX(4)
     3+PSXX(5)+PSXX(6)
     4+PSXX(7)+PSXX(8)
     5)/8.0D00
      PSXYE(IE)=(
     1 PSXY(1)+PSXY(2)
     2+PSXY(3)+PSXY(4)
     3+PSXY(5)+PSXY(6)
     4+PSXY(7)+PSXY(8)
     5)/8.0D00
      PSXZE(IE)=(
     1 PSXZ(1)+PSXZ(2)
     2+PSXZ(3)+PSXZ(4)
     3+PSXZ(5)+PSXZ(6)
     4+PSXZ(7)+PSXZ(8)
     5)/8.0D00
      PSYXE(IE)=(
     1 PSYX(1)+PSYX(2)
```

```
     2+PSYX(3)+PSYX(4)
     3+PSYX(5)+PSYX(6)
     4+PSYX(7)+PSYX(8)
     5)/8.0D00
      PSYYE(IE)=(
     1 PSYY(1)+PSYY(2)
     2+PSYY(3)+PSYY(4)
     3+PSYY(5)+PSYY(6)
     4+PSYY(7)+PSYY(8)
     5)/8.0D00
      PSYZE(IE)=(
     1 PSYZ(1)+PSYZ(2)
     2+PSYZ(3)+PSYZ(4)
     3+PSYZ(5)+PSYZ(6)
     4+PSYZ(7)+PSYZ(8)
     5)/8.0D00
      PSZXE(IE)=(
     1 PSZX(1)+PSZX(2)
     2+PSZX(3)+PSZX(4)
     3+PSZX(5)+PSZX(6)
     4+PSZX(7)+PSZX(8)
     5)/8.0D00
      PSZYE(IE)=(
     1 PSZY(1)+PSZY(2)
     2+PSZY(3)+PSZY(4)
     3+PSZY(5)+PSZY(6)
     4+PSZY(7)+PSZY(8)
     5)/8.0D00
      PSZZE(IE)=(
     1 PSZZ(1)+PSZZ(2)
     2+PSZZ(3)+PSZZ(4)
     3+PSZZ(5)+PSZZ(6)
     4+PSZZ(7)+PSZZ(8)
     5)/8.0D00
 1581 CONTINUE

C ---------------------------------------------------------
C CALCULATION OF SIGMA VALUE FOR EACH FIELD ELEMENT
C FOR SUPERSONIC ELEMENTS, ARTIFICIAL VISCOSITY IS INTRODUCED BY
C USING INFORMATION (VALUES OF MACH NUMBER AND SIGMA) FROM
C DOWNSTREAM (THE PRECEDING ELEMENT ALONG X AXIS)

      DO 879 I=1,NELEM
      IF(DMG(I).GT.1.0D00 )THEN
C ---------------------------------------------------------
C SUPERSONIC CASE; AVSI IS THE (PHY)SS DERIVATIVE OF THE ELEMENT AND
C AVSIU THE (PHY)SS DERIVATIVE OF THE PREVIOUS ELEMENT."AH" IS A
C RELAXATION FACTOR. FINAL EQUATION IS A FUNCTION OF THESE VALUES AND
C THE CORRESPONDING  ELEMENT MACH NUMBERS

      AVSI=
```

```
     1(PSXE(I)*(PSXE(I)*PSXXE(I)+PSYE(I)*(PSXYE(I)+PSYXE(I)))
     2+PSYE(I)*(PSYE(I)*PSYYE(I)+PSZE(I)*(PSYZE(I)+PSZYE(I)))
     3+PSZE(I)*(PSZE(I)*PSZZE(I)+PSXE(I)*(PSXZE(I)+PSZXE(I)))
     4)/VV2(I)

      AVSIU=
     1(PSXE(I-MY+1)*(PSXE(I-MY+1)*PSXXE(I-MY+1)+PSYE(I-MY+1)
     2*(PSXYE(I-MY+1)+PSYXE(I-MY+1)))
     3+PSYE(I-MY+1)*(PSYE(I-MY+1)*PSYYE(I-MY+1)+PSZE(I-MY+1)
     4*(PSYZE(I-MY+1)+PSZYE(I-MY+1)))
     5+PSZE(I-MY+1)*(PSZE(I-MY+1)*PSZZE(I-MY+1)+PSXE(I-MY+1)
     6*(PSXZE(I-MY+1)+PSZXE(I-MY+1)))
     7)/VV2(I-MY+1)

      AH=1.0D00

      D2MG=DMG(I)*DMG(I)
      D2MG1=D2MG-1.0D00
      D2MGU=DMG(I-MY+1)*DMG(I-MY+1)-1.0D00

      D2MG1=D2MG1+0.5D00*D2MG1*D2MG1
      D2MGU=D2MGU+0.5D00*D2MGU*D2MGU

      AF1=D2MG*AVSI
      AF2=D2MG1*AVSI
      AF3=D2MGU*AVSIU

      SIGMA(I)=AF1-(AF2-AF3)*AH

      ELSE
C -------------------------------------------------------------
C SUBSONIC CASE: NORMAL SIGMA EQUATION USED

      SIGMA(I)=DMG(I)*DMG(I)/VV2(I)
     1*(PSXE(I)*PSXE(I)*PSXXE(I)+PSYE(I)*PSYE(I)*PSYYE(I)
     2+PSZE(I)*PSZE(I)*PSZZE(I)
     3+PSXE(I)*PSYE(I)*(PSXYE(I)+PSYXE(I))
     4+PSYE(I)*PSZE(I)*(PSYZE(I)+PSZYE(I))
     5+PSZE(I)*PSXE(I)*(PSZXE(I)+PSXZE(I)))

      END IF
  879 CONTINUE

      RETURN
      END
C=======================================================
      SUBROUTINE FI(IE,J)
C THIS S/R CALCULATES THE F1,F2 & F3 TERMS PART OF THE POTENTIAL
C EFFECT OF WING PANELS ON NODES DISCRETIZATION
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
```

```
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/XYZR1/XR(NN,NPNO),YR(NN,NPNO),ZR(NN,NPNO)
      COMMON/XYZ21R/XP,YP,ZP
      COMMON/F123/F1,F2,F3
      COMMON/SAML/SMALL
      COMMON/PI11/PI
C -------------------------------------------------------------
C CALCULATION OF X3P,Y3P & X2P USED IN ALL 3 CALCULATIONS
      X3P=XR(IE,3)-XP
      Y3P=YR(IE,3)-YP
      X2P=XR(IE,2)-XP

C -------------------------------------------------------------
C CALCULATION OF F1 TERM

      IF(DABS(ZP).LT.SMALL)THEN
      F1=0.0D00
      ELSE
      XY3=XR(IE,3)/YR(IE,3)
      XY3C=DSQRT(XY3*XY3+1.0D00)
      XY32=(XR(IE,2)-XR(IE,3))/YR(IE,3)
      XY32C=DSQRT(XY32*XY32+1.0D00)
      F1=ZP/XY3C*DLOG(2.0D00*(XY3C*XY3C*YR(IE,3)-(YP+XP*XY3)+
     1XY3C*DSQRT(X3P*X3P+Y3P*Y3P+ZP*ZP)))
     2-ZP/XY3C*DLOG(2.0D00*(-(YP+XP*XY3)+
     3XY3C*DSQRT(XP*XP+YP*YP+ZP*ZP)))
     4-ZP/XY32C*DLOG(2.0D00*(XY32C*XY32C*YR(IE,3)-
     5(YP+XY32*X2P)+XY32C*DSQRT(X3P*X3P+Y3P*Y3P+ZP*ZP)))
     6+ZP/XY32C*DLOG(2.0D00*(-(YP+XY32*X2P)+
     7XY32C*DSQRT(X2P*X2P+YP*YP+ZP*ZP)))
      END IF

C -------------------------------------------------------------
C CALCULATION OF F2 TERM

      IF(DABS(ZP).LT.SMALL)THEN
      F2=0.0D00
      ELSE
      XY3=YR(IE,3)/XR(IE,3)
      XY3C=DSQRT(XY3*XY3+1.0D00)
      XY32=YR(IE,3)/(XR(IE,3)-XR(IE,2))
      XY32C=DSQRT(XY32*XY32+1.0D00)
      F2=ZP*DLOG(2.0D00*(X2P+DSQRT(X2P*X2P+YP*YP+ZP*ZP)))
     1-ZP*DLOG(2.0D00*(-XP+DSQRT(XP*XP+YP*YP+ZP*ZP)))
     2+ZP/XY3C*DLOG(2.0D00*(-(XP+YP*XY3)+
     3XY3C*DSQRT(XP*XP+YP*YP+ZP*ZP)))
     4-ZP/XY3C*DLOG(2.0D00*(XY3C*XY3C*XR(IE,3)-(XP+YP*XY3)+
```

```
     5XY3C*DSQRT(X3P*X3P+Y3P*Y3P+ZP*ZP)))
     6+ZP/XY32C*DLOG(2.0D00*(XY32C*XY32C*XR(IE,3)-
     7(XP-XY32*(Y3P-XR(IE,3)*XY32))+XY32C*DSQRT(X3P*X3P+Y3P*Y3P+ZP*ZP)))
     8-ZP/XY32C*DLOG(2.0D00*(XY32C*XY32C*XR(IE,2)-
     9(XP-XY32*(Y3P-XR(IE,3)*XY32))+XY32C*DSQRT(X2P*X2P+YP*YP+ZP*ZP)))
      END IF

C -------------------------------------------------------------
C CALCULATION OF F3 TERM

      IF(DABS(ZP).LT.SMALL)THEN
      XY3=XR(IE,3)/YR(IE,3)
      XY32=(XR(IE,3)-XR(IE,2))/YR(IE,3)
      Y2P=YR(IE,2)-YP
      IF(DABS(X3P).LT.SMALL.AND.DABS(Y3P).LT.SMALL)THEN
      F3=-DATAN2(XR(IE,3)-XR(IE,2),YR(IE,3))
     2+DATAN2(XY32*ZP,DSQRT(YP*YP+X2P*X2P+ZP*ZP))
     4+DATAN2(XR(IE,3),YR(IE,3))
     6-DATAN2(XY3*ZP,DSQRT(YP*YP+XP*XP+ZP*ZP))
      ELSE
      IF(DABS(X2P).LT.SMALL.AND.DABS(Y2P).LT.SMALL)THEN
      F3=-DATAN2(XY32*ZP,DSQRT(Y3P*Y3P+X3P*X3P+ZP*ZP))
     2+DATAN2(XR(IE,3)-XR(IE,2),YR(IE,3))
     4+DATAN2(XY3*ZP*ZP-(XY3*YP-XP)*Y3P,
     5ZP*DSQRT(Y3P*Y3P+X3P*X3P+ZP*ZP))
     6-DATAN2(XY3*ZP,DSQRT(YP*YP+XP*XP+ZP*ZP))
      ELSE
      IF(DABS(XP).LT.SMALL.AND.DABS(YP).LT.SMALL)THEN
      F3=-DATAN2(XY32*ZP*ZP-(-XY32*Y3P+X3P)*Y3P,
     1ZP*DSQRT(Y3P*Y3P+X3P*X3P+ZP*ZP))
     2+DATAN2(XY32*ZP,DSQRT(YP*YP+X2P*X2P+ZP*ZP))
     4+DATAN2(XY3*ZP,DSQRT(Y3P*Y3P+X3P*X3P+ZP*ZP))
     6-DATAN2(XR(IE,3),YR(IE,3))
      ELSE
      F3=0.0D00
      END IF
      END IF
      END IF
      ELSE
      XY3=XR(IE,3)/YR(IE,3)
      XY32=(XR(IE,3)-XR(IE,2))/YR(IE,3)
      IF(DABS(XY32).LT.SMALL)THEN
      F3=-DATAN2((XY32*ZP*ZP-(-XY32*Y3P+X3P)*Y3P),
     1DABS(ZP)*DSQRT(Y3P*Y3P+X3P*X3P+ZP*ZP))
     2+DATAN2((XY32*ZP*ZP-(-XY32*Y3P+X3P)*(-YP)),
     3DABS(ZP)*DSQRT(YP*YP+X2P*X2P+ZP*ZP))
     4+XY3/DABS(XY3)*DATAN2(XY3*XY3*ZP*ZP-XY3*(XY3*YP-XP)*Y3P,
     5DABS(XY3*ZP)*DSQRT(Y3P*Y3P+X3P*X3P+ZP*ZP))
     6-XY3/DABS(XY3)*DATAN2(XY3*XY3*ZP*ZP-XY3*(XY3*YP-XP)*(-YP),
     7DABS(XY3*ZP)*DSQRT(YP*YP+XP*XP+ZP*ZP))
```

```
      ELSE
      F3=-XY32/DABS(XY32)*DATAN2(XY32*(XY32*ZP*ZP-(-XY32*Y3P+X3P)*Y3P),
     1DABS(XY32*ZP)*DSQRT(Y3P*Y3P+X3P*X3P+ZP*ZP))
     2+XY32/DABS(XY32)*DATAN2(XY32*(XY32*ZP*ZP-(-XY32*Y3P+X3P)*(-YP)),
     3DABS(XY32*ZP)*DSQRT(YP*YP+X2P*X2P+ZP*ZP))
     4+XY3/DABS(XY3)*DATAN2(XY3*XY3*ZP*ZP-XY3*(XY3*YP-XP)*Y3P,
     5DABS(XY3*ZP)*DSQRT(Y3P*Y3P+X3P*X3P+ZP*ZP))
     6-XY3/DABS(XY3)*DATAN2(XY3*XY3*ZP*ZP-XY3*(XY3*YP-XP)*(-YP),
     7DABS(XY3*ZP)*DSQRT(YP*YP+XP*XP+ZP*ZP))
      END IF
      IF(ZP.LT.0.0D00)F3=-F3
      END IF

      RETURN
      END
C=======================================================
      SUBROUTINE FI3W(IEW,J)
C THIS S/R CALCULATES THE EFFECT OF THE WAKE ON NODES POTENTIAL.
C TERM F3W IS PART OF THE DISCRETIZATION
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)
      COMMON/SAML/SMALL
      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/XYZ21W/XPW,YPW,ZPW
      COMMON/F12W/F3W
      COMMON/NNNW/NODEW(NL,NWNO)
      COMMON/XYZRW/XW(NL,NWUNO),YW(NL,NWUNO),ZW(NL,NWUNO)
      COMMON/PI11/PI

      IF(DABS(ZPW).LT.SMALL)THEN
         IF((XPW.GT.XW(IEW,1).OR.XPW.GT.XW(IEW,2)).AND.
     1    (DABS(YPW-YW(IEW,1)).LT.SMALL.OR.DABS(YPW-YW(IEW,2)).LT.
     2    SMALL))THEN
            F3W=PI
            IF(J.GT.NGLOB/2)F3W=-PI
            IF(IEW.EQ.1.AND.DABS(YPW-YW(IEW,2)).LT.SMALL)F3W=0.0D00
         ELSE
            F3W=0.0D00
         END IF
      ELSE
      XY12=XW(IEW,2)/YW(IEW,2)
      XPA=XPW-XW(IEW,1)
      YPA=YPW-YW(IEW,1)
      XPB=XPW-XW(IEW,2)
      YPB=YPW-YW(IEW,2)
      IF(DABS(XY12).LT.SMALL)THEN
      F3W=DATAN2(ZPW*YPB-YPA*ZPW,YPB*YPA+ZPW*ZPW)
```

```
     2-ZPW/DABS(ZPW)*
     3DATAN2(-XPA*YPB,
     4DABS(ZPW)*DSQRT(XPA*XPA+YPB*YPB+ZPW*ZPW))
     5+ZPW/DABS(ZPW)*
     6DATAN2(-XPA*YPA,
     7DABS(ZPW)*DSQRT(XPA*XPA+YPA*YPA+ZPW*ZPW))
      ELSE
      F3W=DATAN2(ZPW*YPB-ZPW*YPA,YPB*YPA+ZPW*ZPW)
     2-ZPW/DABS(ZPW)*XY12/DABS(XY12)*
     3DATAN2(XY12*(XY12*ZPW*ZPW+(-XPW+XY12*YPW)*YPB),
     4DABS(XY12*ZPW)*DSQRT(XPB*XPB+YPB*YPB+ZPW*ZPW))
     5+ZPW/DABS(ZPW)*XY12/DABS(XY12)*
     6DATAN2(XY12*(XY12*ZPW*ZPW+(-XPW+XY12*YPW)*YPW),
     7DABS(XY12*ZPW)*DSQRT(XPA*XPA+YPA*YPA+ZPW*ZPW))
      END IF
      END IF
      RETURN
      END
C=======================================================
      SUBROUTINE PHYBXYZ
C THIS S/R EVALUATES THE PRESSURE COEFFICIENTS OF WING PANELS AND
C THEN THE CP OF THE WING NODES (AVERAGE OF CLOSE PANELS CP).
C CP OUTPUTS ARE PRINTED IN THIS S/R
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)
      COMMON/XYZA/X(NN1),Y(NN1),Z(NN1),X2(NN1),Y2(NN1),Z2(NN1)
      COMMON/NNNN/NODEB(NN,NPNO)
      COMMON/PI11/PI
      COMMON/XYZR1/XR(NN,NPNO),YR(NN,NPNO),ZR(NN,NPNO)
      COMMON/COLMN1/COLX(NN),COMX(NN),CONX(NN)
      COMMON/COLMN2/COLY(NN),COMY(NN),CONY(NN)
      COMMON/COLMN3/COLZ(NN),COMZ(NN),CONZ(NN)
      COMMON/NNNW/NODEW(NL,NWNO)
      COMMON/QQ1/Q(NN1)
      COMMON/QXYZ1/QX(NN),QY(NN),QZ(NN)
      COMMON/XYZIE/XIE(NN),YIE(NN),ZIE(NN)
      COMMON/VCP1/CP(NN),CPN(NN1),VC(NN),DML(NN)
      COMMON/DM010/DM0,DMM
      COMMON/NE1B/NEB(NN1,NBQ)
      COMMON/KE123/KSU,KSL,KSUE(NSHO),KSLE(NSHO),KE(NELEM),KN(NGLOB)
      COMMON/ABC11/ALPHA,BETA,GAMA
      COMMON/NGL1/NGL(NGLOB),NGLT(NN1),NWAKE(NGLOB)
      COMMON/YMAX11/YMAX,YHSPAN,FRAC,SURFW
C -------------------------------------------------------
C LOOP 7000 CALCULATES WING PANEL CP
C BE & CE ARE THE SPEEDS ALONG XR & YR RELATIVE COORDS. RESPECTIVELY
C QX,QY AND QZ ARE THE POTENTIAL DERIVATIVES (SPEEDS) IN GENERAL COORDS
```

```
C VC IS THE ABSOLUTE SPEED, DML THE PANEL MACH NUMBER
C THE CP IS THEN CALCULATED: THERE ARE EQUATIONS FOR THE COMPRESSIBLE
C AND INCOMPRESSIBLE CASES
      DO 7000 IE=1,NN
      BE=(Q(NODEB(IE,2))-Q(NODEB(IE,1)))/XR(IE,2)
      CE=(Q(NODEB(IE,3))*XR(IE,2)-Q(NODEB(IE,2))*XR(IE,3)
     1+Q(NODEB(IE,1))*(XR(IE,3)-XR(IE,2)))/XR(IE,2)/YR(IE,3)
      QX(IE)=BE*COLX(IE)+CE*COLY(IE)
      QY(IE)=BE*COMX(IE)+CE*COMY(IE)
      QZ(IE)=BE*CONX(IE)+CE*CONY(IE)
      VC(IE)=DSQRT(QX(IE)*QX(IE)+QY(IE)*QY(IE)+QZ(IE)*QZ(IE))
      DM2=DM0*VC(IE)*VC(IE)
      DML(IE)=DSQRT(DM2/(1.0D00-0.2D00*DM2))
      PRESE=( 0.2D00*DMM*DMM*(1.0D00-VC(IE)*VC(IE)) + 1.0D00)**3.5D00
      IF(DMM.GT.0.001D00) THEN
        CP(IE)=(PRESE-1.0D00)/0.7D00/DMM/DMM
      ELSE
        CP(IE)=1.0D00-VC(IE)*VC(IE)
      END IF
 7000 CONTINUE
C -------------------------------------------------------
C CALCULATION OF WING NODE CP
C AN AVERAGE OF MEETING PANELS CP IS MADE
C -------------------------------------------------------
C LOOP 710 IS FOR "INSIDE" UPPER & LOWER NODES (6 MEETING PANELS)
C SPECIAL CASE FOR WING NODES OF WHICH THE CORRESPONDING FIELD NODE
C (NGLT(II)) IS PART OF A SHOCK SURFACE (KN.EQ.NGLT(II)): AN AVERAGE
C OF CP BEFORE THE SHOCK IS MADE FOR UPPER & LOWER SURFACE (2ND IF)
      DO 710 I=2,NH-1
      DO 711 J=2,NV-1
      II=I+(J-1)*NH
      SUMV=0.0D00
      DO 822 K=1,NBQ
      SUMV=SUMV+CP(NEB(II,K))
  822 CONTINUE
      CPN(II)=SUMV/DFLOAT(NBQ)
      IF(KN(NGLT(II)).EQ.NGLT(II))THEN
      IF(II.LT.NH*(NV-1)/2+1)THEN
      CPN(II)=(CP(NEB(II,4))+CP(NEB(II,5))+CP(NEB(II,6)))/3.0D00
      ELSE
      CPN(II)=(CP(NEB(II,1))+CP(NEB(II,2))+CP(NEB(II,3)))/3.0D00
      END IF
      END IF
  711 CONTINUE
  710 CONTINUE
C -------------------------------------------------------
C LOOP 720 IS FOR UPPER T.E. NODES (3 PANELS AVERAGE)
C LOOP 730 IS FOR LOWER T.E. NODES (3 PANELS AVERAGE)
      DO 720 I=2,NH-1
      J=1
```

```
      II=I
      CPN(II)=(CP(NEB(II,4))+CP(NEB(II,5))+CP(NEB(II,6)))/3.0D00
  720 CONTINUE
      DO 730 I=2,NH-1
      J=NV
      II=I+(J-1)*NH-1
      CPN(II)=(CP(NEB(II,1))+CP(NEB(II,2))+CP(NEB(II,3)))/3.0D00
  730 CONTINUE
C -------------------------------------------------------------
C LOOP 721 IS FOR UPPER TIP NODES (3 PANELS AVERAGE)
C LOOP 2721 IS FOR LOWER TIP NODES (3 PANELS AVERAGE)
C WHEN THERE ARE SHOCKS, AVERAGE OF 2 PANELS BEFORE SHOCK IS MADE
      I=1
      DO 721 J=2,(NV-1)/2
      II=I+(J-1)*NH
      CPN(II)=(CP(NEB(II,3))+CP(NEB(II,4))+CP(NEB(II,5)))/3.0D00
      IF(KN(NGLT(II)).EQ.NGLT(II))THEN
      CPN(II)=(CP(NEB(II,4))+CP(NEB(II,5)))/2.0D00
      END IF
  721 CONTINUE
      DO 2721 J=(NV-1)/2+2,NV-1
      II=I+(J-1)*NH
      CPN(II)=(CP(NEB(II,2))+CP(NEB(II,3))+CP(NEB(II,4)))/3.0D00
      IF(KN(NGLT(II)).EQ.NGLT(II))THEN
      CPN(II)=(CP(NEB(II,2))+CP(NEB(II,3)))/2.0D00
      END IF
 2721 CONTINUE
C -------------------------------------------------------------
C NODE AT TIP & L.E. : 2 PANEL AVERAGE
      J=(NV-1)/2+1
      II=I+(J-1)*NH
      CPN(II)=(CP(NEB(II,3))+CP(NEB(II,4)))/2.0D00
C -------------------------------------------------------------
C LOOP 731 IS FOR WING ROOT UPPER NODES (3 PANELS AVERAGE)
C LOOP 2731 IS FOR WING ROOT LOWER NODES (3 PANELS AVERAGE)
C SPECIAL CASES FOR SHOCKS AS BEFORE
      I=NH
      DO 731 J=2,(NV-1)/2
      II=I+(J-1)*NH
      CPN(II)=(CP(NEB(II,1))+CP(NEB(II,2))+CP(NEB(II,6)))/3.0D00
      IF(KN(NGLT(II)).EQ.NGLT(II))THEN
      CPN(II)=CP(NEB(II,6))
      END IF
 731 CONTINUE
      DO 2731 J=(NV-1)/2+2,NV-1
      II=I+(J-1)*NH
      CPN(II)=(CP(NEB(II,1))+CP(NEB(II,5))+CP(NEB(II,6)))/3.0D00
      IF(KN(NGLT(II)).EQ.NGLT(II))THEN
      CPN(II)=CP(NEB(II,1))
      END IF
2731 CONTINUE
```

```
C ---------------------------------------------------------
C NODE AT L.E. AND WING ROOT: 2 PANEL AVERAGE
      J=(NV-1)/2+1
      II=I+(J-1)*NH
      CPN(II)=(CP(NEB(II,1))+CP(NEB(II,2))
     1+CP(NEB(II,5))+CP(NEB(II,6)))/4.0D00
C ---------------------------------------------------------
C NODE 1,NH & NNK (WING CORNERS):#1 USES 4 PANELS, THE OTHER 2 ONLY 1
      CPN(1)=(CP(NEB(1,3))+CP(NEB(1,4))
     1+CP(NEB(1,5))+CP(NEB(1,6)))/4.0D00
      CPN(NH)=CP(NEB(NH,6))
      CPN(NNK)=CP(NEB(NNK,1))
C ---------------------------------------------------------
C CALCULATION OF WING LIFT COEFFICIENT
C THIS IS DONE BY INTEGRATING THE PANEL CP WITH NECESSARY COSINES
C IN THIS VERSION, T.E. PANELS ARE REMOVED FROM THE SUMMATION BECAUSE
C OF THEIR HIGH CP VALUES. REFERENCE SURFACE CALCULATED IN S/R XYZL IS
C USED (SURFW). "FNOR" REPRESENTS THE NORMAL FORCE ON PANEL SURFACE.
C LATERAL FORCES ARE ASSUMED TO CANCEL THEMSELVES (SYMETRICAL LOADING)
      SUMX=0.0D00
      SUMZ=0.0D00
      IESURF=NL*(NV-1)*2
      DO 75 IE=1,NN
        IF( (IE.LE.2*NL) .OR. (IE.GE.(IESURF-2*NL)) ) GOTO 75
        FNOR=CP(IE)*XR(IE,2)*YR(IE,3)/2.0D00
        IF(DABS(COLZ(IE)).GT.0.001D00) SUMX=SUMX-FNOR/COLZ(IE)
        IF(DABS(CONZ(IE)).GT.0.001D00) SUMZ=SUMZ-FNOR/CONZ(IE)
   75 CONTINUE
      CFX=SUMX/SURFW
      CFZ=SUMZ/SURFW
      CL=CFZ*DCOS(ALPHA) - CFX*DSIN(ALPHA)
      CD=CFX*DCOS(ALPHA) + CFZ*DSIN(ALPHA)
      WRITE(6,*)'*****************************************'
      WRITE(6,*)'WING LIFT COEFFICIENT (CL)= ',CL
C     WRITE(6,*)'WING DRAG COEFFICIENT (CD)= ',CD
      WRITE(6,*)'*****************************************'

C ---------------------------------------------------------
C        C P    O U T P U T     W R I T I N G
C ---------------------------------------------------------
C FIRST, CP OF WING PANELS AND RELEVANT INFORMATION IS PRINTED BY
C WING SECTIONS.
C TWO LOOPS ARE NECESSARY (322 & 323) FOR ODD & EVEN NUMBERED
C TRIANGULAR PANELS PER SECTION.
      WRITE(6,*)' '
C     WRITE(6,*)' WING PANELS INFORMATION'
C     WRITE(6,*)' IE  X   Y   Z   CP   M   VX   VY   VZ   VC'
C     WRITE(6,*)' '
   80 FORMAT(I5,9F11.6)
      DO 222 I=1,NL*2-1,2
C     WRITE(6,*)' '
```

```
      DO 322 J=1,NV-1
      IE=I+(J-1)*NL*2
C     WRITE(6,80)IE,XIE(IE),YIE(IE),ZIE(IE),CP(IE),DML(IE),
C    1QX(IE),QY(IE),QZ(IE),VC(IE)
  322 CONTINUE
C     WRITE(6,*)' '
      DO 323 J=1,NV-1
      IE1=I+(J-1)*NL*2+1
C     WRITE(6,80)IE1,XIE(IE1),YIE(IE1),ZIE(IE1),CP(IE1),DML(IE1),
C    1QX(IE1),QY(IE1),QZ(IE1),VC(IE1)
C     WRITE(6,*)IE1,COLZ(IE1),COMZ(IE1),CONZ(IE1),CP(IE1)
  323 CONTINUE
  222 CONTINUE

C -------------------------------------------------------------
C PRINT SIMILAR INFORMATION FOR WINGTIP PANELS
C     WRITE(6,*)' '
C     WRITE(6,*)' WING TIP PANELS INFORMATION'
C     WRITE(6,*)' IE  X   Y   Z   CP   M   VX   VY   VZ   VC'
      DO 344 IE=NL*(NV-1)*2+1,NN
C     WRITE(6,80)IE,XIE(IE),YIE(IE),ZIE(IE),CP(IE),DML(IE),
C    1QX(IE),QY(IE),QZ(IE),VC(IE)
  344 CONTINUE

C -------------------------------------------------------------
C PRINT WING NODES CP AND COORDS AROUND WING SECTIONS STARTING FROM
TIP
C     WRITE(6,*) ' '
      WRITE(6,*) 'WING NODE #,   X,Y,Z COORDINATES,   CP VALUES'
      DO 1710 I=1,NH
      DO 1711 J=1,NV
      II=I+(J-1)*NH
      IF(J.EQ.NV)THEN
      II=II-1
      IF(I.EQ.1)II=1
      END IF
C     WRITE(6,8899)II,X(II),Y(II),Z(II),CPN(II)
      WRITE(6,*) II,CPN(II)
 1711 CONTINUE
 1710 CONTINUE
 8899 FORMAT(I6,4F15.7)
      RETURN
      END
C=============================================================
      SUBROUTINE SHAP1(I,IE)
C THIS S/R CALCULATES THE FINITE ELEMENT DISCRETIZATION TERMS "DNGLO1"
C USED IN S/R "UV"
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(NH=6,NV=33,NO=4,NSHO=100,NL=NH-1,
     1NNK=NH*NV-1,NN1=NNK+(NV-1)/2-1,NN=NL*(NV-1)*2+((NV-1)/2-1)*4,
     2MX=(NV-1)/2+1+2*NO,MY=NH+NO,MZ=(NO+1)*2,MZ2=NO+1,
```

```
     3NGLOB=MX*MY*MZ,NELEM=(MX-1)*(MY-1)*(MZ-2))
      PARAMETER(NDPL=8,NPNO=3,NWNO=4,NWUNO=2,NBQ=6)
      COMMON/DJ1/DJACO1(3,3)
      COMMON/DE1/DETER1
      COMMON/NNODE1/NODE(NELEM,NDPL)
      COMMON/XYZG/X3(NGLOB),Y3(NGLOB),Z3(NGLOB)
      COMMON/DN11/DNLOC1(8,3,8)
      COMMON/DNG1/DNGLO1(8,3)
C -------------------------------------------------------------
C LOOP 101 INITIALIZES DJACO1 ARRAY
C LOOP 100 CALCULATES DJACO1 VALUES USING DNLOC1 TERMS (S/R XYZGN)
C AND FIELD NODES COORDINATES

      DO 101 JJ=1,3
      DO 101 II=1,3
 101  DJACO1(II,JJ)=0.0D00

      DO 100 JJ=1,3
      DO 100 II=1,8
      DJACO1(JJ,1)=DJACO1(JJ,1)+DNLOC1(II,JJ,I)*
     1X3(NODE(IE,II))
      DJACO1(JJ,2)=DJACO1(JJ,2)+DNLOC1(II,JJ,I)*
     1Y3(NODE(IE,II))
      DJACO1(JJ,3)=DJACO1(JJ,3)+DNLOC1(II,JJ,I)*
     1Z3(NODE(IE,II))
 100  CONTINUE

C -------------------------------------------------------------
C CALCULATE "DETER1" TERM

      DETER1=
     1 DJACO1(1,1)*DJACO1(2,2)*DJACO1(3,3)
     2-DJACO1(1,1)*DJACO1(3,2)*DJACO1(2,3)
     3+DJACO1(2,1)*DJACO1(3,2)*DJACO1(1,3)
     4-DJACO1(2,1)*DJACO1(1,2)*DJACO1(3,3)
     5+DJACO1(3,1)*DJACO1(1,2)*DJACO1(2,3)
     6-DJACO1(3,1)*DJACO1(2,2)*DJACO1(1,3)

C -------------------------------------------------------------
C CALCULATE DNGLO1 TERMS

      DO 200 M=1,8
      DNGLO1(M,1)=(
     1 DNLOC1(M,1,I)*DJACO1(2,2)*DJACO1(3,3)
     2-DNLOC1(M,1,I)*DJACO1(3,2)*DJACO1(2,3)
     3+DNLOC1(M,2,I)*DJACO1(3,2)*DJACO1(1,3)
     4-DNLOC1(M,2,I)*DJACO1(1,2)*DJACO1(3,3)
     5+DNLOC1(M,3,I)*DJACO1(1,2)*DJACO1(2,3)
     6-DNLOC1(M,3,I)*DJACO1(2,2)*DJACO1(1,3)
     7)/DETER1
      DNGLO1(M,2)=(
```

```
     1 DJACO1(1,1)*DNLOC1(M,2,I)*DJACO1(3,3)
     2-DJACO1(1,1)*DNLOC1(M,3,I)*DJACO1(2,3)
     3+DJACO1(2,1)*DNLOC1(M,3,I)*DJACO1(1,3)
     4-DJACO1(2,1)*DNLOC1(M,1,I)*DJACO1(3,3)
     5+DJACO1(3,1)*DNLOC1(M,1,I)*DJACO1(2,3)
     6-DJACO1(3,1)*DNLOC1(M,2,I)*DJACO1(1,3)
     7)/DETER1
      DNGLO1(M,3)=(
     1 DJACO1(1,1)*DJACO1(2,2)*DNLOC1(M,3,I)
     2-DJACO1(1,1)*DJACO1(3,2)*DNLOC1(M,2,I)
     3+DJACO1(2,1)*DJACO1(3,2)*DNLOC1(M,1,I)
     4-DJACO1(2,1)*DJACO1(1,2)*DNLOC1(M,3,I)
     5+DJACO1(3,1)*DJACO1(1,2)*DNLOC1(M,2,I)
     6-DJACO1(3,1)*DJACO1(2,2)*DNLOC1(M,1,I)
     7)/DETER1
  200 CONTINUE
      RETURN
      END
```