

NADC-77163-30

LEVEL #

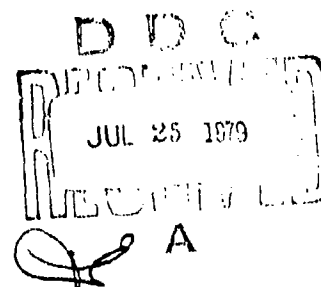
A COMPUTATIONAL MODEL FOR THREE-DIMENSIONAL
INCOMPRESSIBLE WALL JETS WITH LARGE CROSS FLOW

by

W.D. MURPHY, V. SHANKAR, and N. MALMUTH

SCIENCE CENTER
ROCKWELL INTERNATIONAL
THOUSAND OAKS, CA 91360

SEPTEMBER 1978



APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

PREPARED FOR

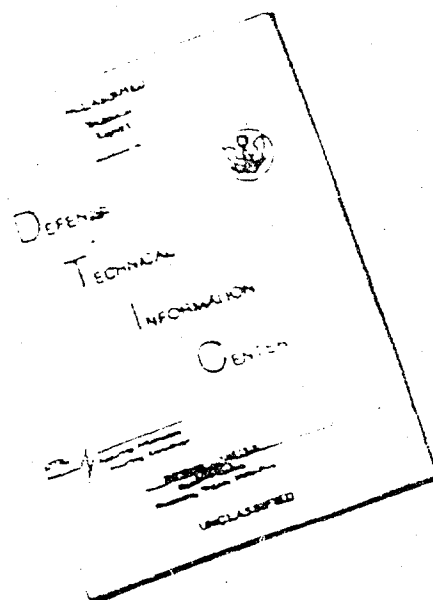
NAVAL AIR DEVELOPMENT CENTER
WARMINSTER, PENNSYLVANIA 18974

DDC FILE COPY

ADA 071 689

79 07 24 075

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST
QUALITY AVAILABLE. THE COPY
FURNISHED TO DTIC CONTAINED
A SIGNIFICANT NUMBER OF
PAGES WHICH DO NOT
REPRODUCE LEGIBLY.

REPRODUCED FROM
BEST AVAILABLE COPY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NADC-77163-30	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A COMPUTATIONAL MODEL FOR THREE-DIMENSIONAL INCOMPRESSIBLE WALL JETS WITH LARGE CROSS FLOW.		5. TYPE OF REPORT & PERIOD COVERED Final Report, 19 Sep 77 - through 18 Sep 78
7. AUTHOR(s) W.D. Murphy, V. Shankar and N. Malmuth		6. PERFORMING ORG. REPORT NUMBER SC5129.5FR
9. PERFORMING ORGANIZATION NAME AND ADDRESS Science Center Rockwell International Thousand Oaks, CA 91360		8. CONTRACT OR GRANT NUMBER(s) N62269-77-C-0412
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Air Development Center Code 3014 Warminster, PA 18974		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS P.E. - 61153N PROJ. - 602302 T.A. - 60230221 W.U. - 6070188
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) (12) 227p		12. REPORT DATE September 1978
		13. NUMBER OF PAGES 146
		14. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Incompressible Flow Partial Differential Equations Viscous Flow Numerical Analysis Jets Ejectors		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The flow field of three dimensional incompressible wall jets prototypic of thrust augmenting ejectors with large cross flow is solved using a very efficient centered-Euler scheme in an orthogonal curvilinear coordinate system. The computational model treats initial conditions with arbitrary velocity profiles at or downstream of the jet exit. An averaging approach is employed for the first few marching steps to overcome spurious numerical oscillations associated with arbitrary initial profiles. Laminar as well as turbulent wall jets are		

DD FORM 1 JAN 73 1473

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

389 949

LB

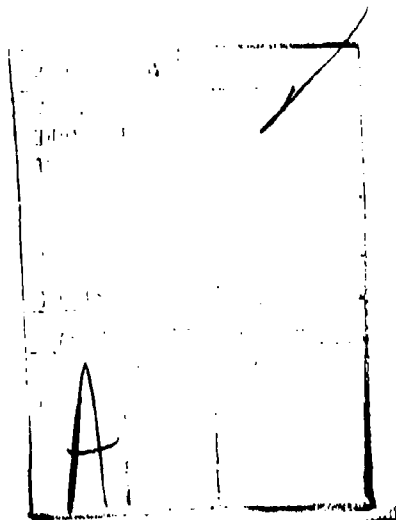
UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Block 20 Continued:

simulated. Turbulence is introduced using a two layer mixing length model appropriate to curved three-dimensional wall jets. Typical results quantifying jet spreading, jet growth, nominal separation and jet shrink effects due to cross flow are presented.

A class of cases was investigated roughly possessing initial flow angularity and adverse pressure gradients prototypic of those on the blown surfaces of typical propulsive lift systems such as the Navy/Rockwell XfV-12A thrust augmented wing. Results obtained from the computational model indicate that if the initial total velocity is kept fixed, then the introduction of the cross flow enhances the freestream decay rate of the peak of the velocity component in the freestream direction. In addition, the entrainment quantity and its rate decrease with increased cross flow. The three-dimensional phenomena not only influence the effect of taper on the boundary layer control characteristics of a Coanda flap, but also indicate a "jet shrink" which could be a mechanism promoting end-wall separation. To our knowledge, our model is the first to quantify such trends. Both should be considered in the design of any propulsive lift system. Finally, the effect on the prescribed external adverse pressure gradient in the presence and absence of cross flow has also been examined. From the limited results, the spanwise separation line moves progressively further upstream with increasing cross flow.



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

A COMPUTATIONAL MODEL FOR THREE-DIMENSIONAL
INCOMPRESSIBLE WALL JETS WITH LARGE CROSS FLOW

by

W.D. MURPHY, V. SHANKAR, and N. MALMUTH

SCIENCE CENTER
ROCKWELL INTERNATIONAL
THOUSAND OAKS, CA 91360

SEPTEMBER 1978

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

PREPARED FOR

NAVAL AIR DEVELOPMENT CENTER
WARMINSTER, PENNSYLVANIA 18974

FOREWORD

This document describes computational studies of three-dimensional incompressible turbulent wall jets with large cross flow. The effort was performed during the period September 19, 1977 to September 18, 1978 and was sponsored by the Naval Air Development Center under Contract N62269-77-C-0412.

The technical monitor for this study was Dr. K.A. Green.

ABSTRACT

The flow field of three dimensional incompressible wall jets prototypic of thrust augmenting ejectors with large cross flow is solved using a very efficient centered-Euler scheme in an orthogonal curvilinear coordinate system. The computational model treats initial conditions with arbitrary velocity profiles at or downstream of the jet exit. An averaging approach is employed for the first few marching steps to overcome spurious numerical oscillations associated with arbitrary initial profiles. Laminar as well as turbulent wall jets are simulated. Turbulence is introduced using a two layer mixing length model appropriate to curved three-dimensional wall jets. Typical results quantifying jet spreading, jet growth, nominal separation and jet shrink effects due to cross flow are presented.

A class of cases was investigated roughly possessing initial flow angularity and adverse pressure gradients prototypic of those on the blown surfaces of typical propulsive lift systems such as the Navy/Rockwell XTV-12A thrust augmented wing. Results obtained from the computational model indicate that if the initial total velocity is kept fixed, then the introduction of the cross flow enhances the freestream decay rate of the peak of the velocity component in the freestream direction. In addition, the entrainment quantity and its rate decrease with increased cross flow. The three-dimensional phenomena not only influence the effect of taper on the boundary layer control characteristics of a Coanda flap, but also indicate a "jet shrink" which could be a mechanism promoting end-wall separation. To our knowledge, our model is the first to quantify such trends. Both should be considered in the design of any propulsive lift system. Finally, the effect on the prescribed external adverse pressure gradient in the presence and absence of cross flow has also been examined. From the limited results, the spanwise separation line moves progressively further upstream with increasing cross flow.

CONTENTS

	<u>Page</u>
FOREWORD	v
ABSTRACT	vii
CONTENTS	ix
ILLUSTRATIONS	xi
NOMENCLATURE	xiii

PART I - THEORETICAL ANALYSIS AND RESULTS

1.0 Introduction	I-1
2.0 Problem Description and Formulation	I-4
3.0 Eddy-Viscosity Model	I-8
4.0 Finite Difference Equations	I-13
5.0 Results	I-19
6.0 Conclusions	I-30
7.0 References	I-32

PART II - USERS MANUAL

1. Deck Setup	II-1
2. Estimate of Running Time	II-2
3. Type and Configuration of Computer Used in Program Development	II-2
4. Name and Level of Programming Language Used in Program	II-2
A. Input-Output Information	II-2
Glossary of Input Parameters	II-2
Input	II-5

CONTENTS (Continued)

	<u>Page</u>
Plots	II-6
Example	II-7
Output	II-7
Example	II-9
B. Subroutine Description	II-11
BC	II-11
BETASV	II-11
BLOCK1	II-13
BLOCK2	II-13
BOX	II-14
LUSOLV	II-14
MAIN	II-14
NETRUN	II-13
NEWTON	II-13
OUTPT	II-16
PREP	II-17
PREPB	II-17
PREPG	II-18
PREPG2	II-18
PREPP*	II-19
PRMESH*	II-20
PROFLE*	II-20
RHSF	II-21
RHSF2D	II-21
TRUN	II-22
USOLVE	II-22
VANDET	II-22
VELMAX	II-23
WALJET	II-23
XMESH*	II-24
YMESH*	II-24
ZMESH*	II-24
APPENDIX - PROGRAM LISTING	A-1

ILLUSTRATIONS

<u>Figure</u>	<u>Title</u>	<u>Page</u>
1	XFV-12A Prototypic Augmenter Configuration	I-5
2	Upper Surface Blowing Boundary Layer Control-Supercirculation Wing	I-5
3	Physical System and Flow Schematic	I-7
4	Two-Layer Eddy Viscosity Model	I-10
5	Initial and Boundary Conditions	I-18
6	Effect of Cross Flow on Jet Growth, $\frac{u_{\max}}{u_e} = 2 - .5-z $, $u_e = u_0 x^{-1/2} \cos \theta$, $\theta = \frac{\pi}{2} z(z_{\text{tip}} - z)$, Standard Initial Profile (See Fig. 5), $z = z_1 = z_{\text{tip}}/2$	I-21
7	Upper Surface Blown Thrust Augmented Wing (TAW) . . .	I-22
8	Effect of Cross Flow on Nominal Entrainment, Q with $u_e = u_0 x^{-1/2} \cos \theta$, $Q = \int_0^\infty \int_0^{z_{\text{tip}}} u dy dz$, $\theta = C \frac{\pi}{2} z(z_{\text{tip}} - z)$, $\frac{u_{\max}}{u_e} = 2 - .5-z $, (Standard Initial Profile) $z = z_1 = z_{\text{tip}}/2$	I-24
9	Effect of Cross Flow on Reduced Shear Stress, $u_e = u_0 x^{-1/2} \cos \theta$, $\theta = C \frac{\pi}{2} z(z_{\text{tip}} - z)$, Standard Initial Profile, $(z = z_1 = z_{\text{tip}}/2)$, $\frac{u_{\max}}{u_e} = 2 - .5-z $	I-25

ILLUSTRATIONS (Continued)

<u>Figure</u>	<u>Title</u>	<u>Page</u>
10	Effect of Cross Flow on Jet Spreading, $u_e = u_o x^{-1/2} \cos \theta$, $\theta = C \frac{\pi}{2} z(z_{tip} - z)$, Standard Initial Profile at Midspan ($z = z_{tip}/2$), $\frac{u_{max}}{u_e} = 2 - .5 - z $	I-27
11	Effect of Cross Flow on the Locus of Nominal Separation, $u_e = u_o x^{-1/2} \cos \theta$, $w_e = u_o x^{-1/2} \sin \theta$, $\theta = C \frac{\pi}{2} z(z_{tip} - z)$, $\frac{u_{max}}{u_e} = 2 - .5 - z $	I-28
12	Cross Flow Effect on Jet "Shrink" and "End Wall Pullaway," $u_e = u_o x^{-1/2} \cos \theta$, $w_e = u_o x^{-1/2} \sin \theta$, $\frac{u_{max}}{u_e} = 2 - .5 - z $	I-29
13	Cross Flow Effect on Jet "Shrink" with Streamwise Increase in Cross Flow, $u_e = u_o x^{-1/2} \cos \theta$, $w_e = u_o x^{1/2} \sin \theta$, $\frac{u_{max}}{u_e} = 2 - 0.5 - z $, $\theta = \frac{C\pi}{2} z(z_{tip} - z)$	I-31

NOMENCLATURE

b	Defined in Eq. (14a)
f, g	Reduced vector potentials (Eqs. (8a) and (8b))
h_1, h_2, h_3	Metric coefficients
K_1, K_2	Geodesic curvatures
$P_i, (i=1, \dots, 10)$	Constants defined on page I-12
M, N, Q, R	Constants defined on page I-12
Q	Entrainment quantity
u, v, w	x, y, z components of velocity (see Fig. 3)
$\overline{u'v'}, \overline{v'w'}$	Reynolds stress correlations
S_1	Arc length along x coordinate
S	Area of integration
(x, y, z)	Orthogonal curvilinear coordinates parallel and perpendicular to wall (see Fig. 3)
y^*	Height of first turbulent layer in two layer eddy viscosity model
y_1	Constant appearing in eddy viscosity model, page I-9
ϵ	Eddy viscosity (page I-9)
$\overline{\epsilon}$	Reduced eddy viscosity (page I-9)
ϵ^+	Reduced eddy viscosity defined in Eq. (14a)
κ_1, κ_2	Radius of curvature of $z=\text{constant}$ and $x=\text{constant}$ lines on wall jet surface
ϕ	Component of vector potential (page I-8), velocity potential (page I-20)
ψ	Component of vector potential (page I-8)

NOMENCLATURE (Continued)

σ	Source strength (page I-20)
ν	Kinematic viscosity
τ	Shear stress = $\rho(\epsilon + \nu) \sqrt{u_y^2 + w_y^2}$
θ	Initial jet flow angle parameter, Fig. 8
ξ, ζ	Dummy variables, page I-20

Subscripts

e	External flow
i, j	Nodal locations for difference operators (Note also that subscript notation is used in what follows to denote partial differentiation.)

Superscript

n	Time level and nodal location for difference operators
-----	--

PART I - THEORETICAL ANALYSIS AND RESULTS

1.0 Introduction

Modern naval aircraft can reduce strike force vulnerability through greater dispersal. A way of achieving this dispersal would be the development and deployment of vertical lift-off and landing aircraft operating from ships that are significantly smaller than the currently operating carriers. One propulsion concept designed to achieve a vertical and short take-off and landing (V/STOL) aircraft involves the use of thrust augmenting ejectors to amplify the thrust of the engines in the vertical mode. This technique is utilized in the current XFV-12A aircraft. To achieve adequate accelerations for typical payloads, a high augmentation ratio (ϕ) is required. Various augments designs have been proposed to achieve these high ϕ values as well as integrate well with the aircraft structure. In the XFV-12A, an ejector system composed of a centerbody and two Coanda wall jets is currently under development. A central feature of the flow fields produced by this device is three dimensionality. This has been particularly evident in subscale flow visualization on the Coanda surfaces. It is believed that these flow processes may be important toward ϕ maximization. The augmentation ratio depends on the momentum flux and the dynamic head losses downstream of the Coandas and the centerbody nozzle, and in particular, in the diffuser section of the augmenter. Maintenance of a high momentum and minimization of extraneous motions, while accelerating as quickly as possible the secondary flows using the primary jets, is of paramount importance. An example of an extraneous motion is associated with certain vortex structures which have been observed in augmenter wing configurations. It has been postulated that these motions and general flow turning are induced by the three-dimensional features of the configuration. If the wing flap on which the Coandas are mounted has a tapered diffuser, swept back trailing edge, and small aspect ratio planform, considerable cross flow in the secondary can arise. Under these circumstances, this turning is

further enhanced by spanwise pressure gradients arising along the tapered centerbody and Coanda slots. These factors can also lead to considerable flow nonuniformities both in the spanwise and chordwise directions. To compound the situation, flow surveys for three dimensional research configurations of this type have indicated regions of separation near the endwalls. The amount and nature of blowing in these regions to maintain attached flow and create the highest possible momentum flux across some downstream control surface needs quantification.

From the foregoing considerations, it is obvious that an analysis which assumes mean flow quantities such as that employed in the usual one-dimensional derivations can be considerably in error. In fact, if it is assumed in such a treatment that the flow is turned by an angle ω uniformly along the span, the reduction in ϕ is only $\cos \omega$, which is considerably less than rough comparisons between quasi-two-dimensional configurations and actual three-dimensional arrangements would indicate. One way of understanding the foregoing relationships is through theoretical modeling which can provide a means of reducing the high cost of powered lift testing. Unfortunately, existing methodology has been limited in the past to two-dimensional flows for the analysis of wall jets and complete ejector systems. Analytical methods and computational algorithms are therefore necessary to compute three-dimensional flows typical of reality.

To shed light on typical flow patterns encountered due to the effect of taper and sweep on augments wings as well as upper-surface-blown configurations, a study, "Three-Dimensional Flow of a Wall Jet," was initiated by the Naval Air Development Center to investigate wall jet flows which exemplify typical features of more complex propulsive lift applications. The purpose of this study has been to apply modern computational methods to the treatment of wall jet flows with three dimensionality. In a previous phase of the effort, small cross flow wall jets were considered.¹ This report relaxes that assumption and considers large cross flows that occur in practice.

The formulation employs boundary layer equations in an orthogonal curvilinear coordinate system. If the distance from the jet exit is sufficiently large to establish complete mixing, the jet exit height is small compared to a characteristic radius of curvature, and the Reynolds number based on the exit height is large, the order of magnitude analysis given in Ref. 1 extended to three dimensions indicates that the wall jet equations are substantially the same as the three-dimensional boundary layer equations. The basic idea in both the wall jet and boundary layer approximations is the same, namely, that diffusional gradients for the vorticity in the direction normal to the surface are dominant terms in the equations. In fact, half of the wall jet has a boundary layer character due to the no-slip condition on the surface. Furthermore, to dominant order, the pressure is independent of the coordinate normal to the surface. Equilibration of centrifugal force between the streamlines with pressure gradients across them is accomplished with the second-order pressure term and is expressed in the second-order momentum equation in the normal direction. The complete mixing condition is synonymous with merger of the boundary layer from the wall and the shear layer from the jet exit. Assuming that the characteristic distance for merger D is determined by a spreading angle λ of the order of 2.86° appropriate for submerged turbulent free shear layers measured by Reichardt,² then the merger distance is $d \cot \lambda$ where d is the slot height. For an aircraft similar to the XFV-12A, assuming that $d \sim 3$ cm and the wing chord $L \sim 300$ cm, this estimate shows that $D/L \sim 0.2$. Note, however, that this can be reduced to practically zero if the wall boundary layer of the flow emanating upstream of the nozzle exit completely fills the exit location.

For the computational model, a transformation is incorporated to stretch the coordinate normal to the flow. At streamwise planes, the

resulting nonlinear partial differential equations are treated as ordinary differential equations, incorporating source terms involving partial derivatives representing the upstream history of the flow field. These are solved using a very efficient two-point boundary value finite-difference method devised by Keller and Cebeci³⁻⁵ known as the "box method." In the code, the turbulence is introduced using a two layer mixing length model appropriate to three-dimensional wall jets.

2.0 Problem Description and Formulation

To fix the ideas regarding three-dimensional wall jet flows relevant to propulsive lift applications, consider two prototypic configurations as indicated in Figures 1 and 2. The generic arrangement shown in Figure 1 is relevant to an augmenter ejector wing of the type used on the XFV-12A without the centerbody nozzle. In Figure 2, the shape indicated corresponds to an application involving boundary layer control and supercirculation development. For both cases, the development of the wall jet over the curved surfaces S_1 is of interest. In the augmenter of Figure 1, Coanda jets flow over the surfaces S_1 and S_2 emanating from slots T_1 and T_2 and provide a vertical lift force. In the analysis of the flow field over S_1 , we suppress the influence of the surfaces S_2 , A, and C. In addition, we assume that the secondary flow produced by entrainment resulting from the primary jets emanating from T_1 and T_2 is known *a priori*. In actuality, these must be computed as an integral part of the problem. For tractability, we restrict our consideration to the indicated (parabolic) formulation since it is a building block to a later analysis of the primary secondary interaction. For the configuration of Figure 2, the orientation is similar, and we neglect the primary secondary interaction features. Thus, both configurations lead to the problem of the development of a 3-D wall jet over a curved surface S_1 in which boundary conditions are specified on some interfacial layer with the external or entrained flow. It should be noted that the mixing downstream of the slots T_1 is with a flow above the slot which obeys the no slip condition at the slot trailing edge.

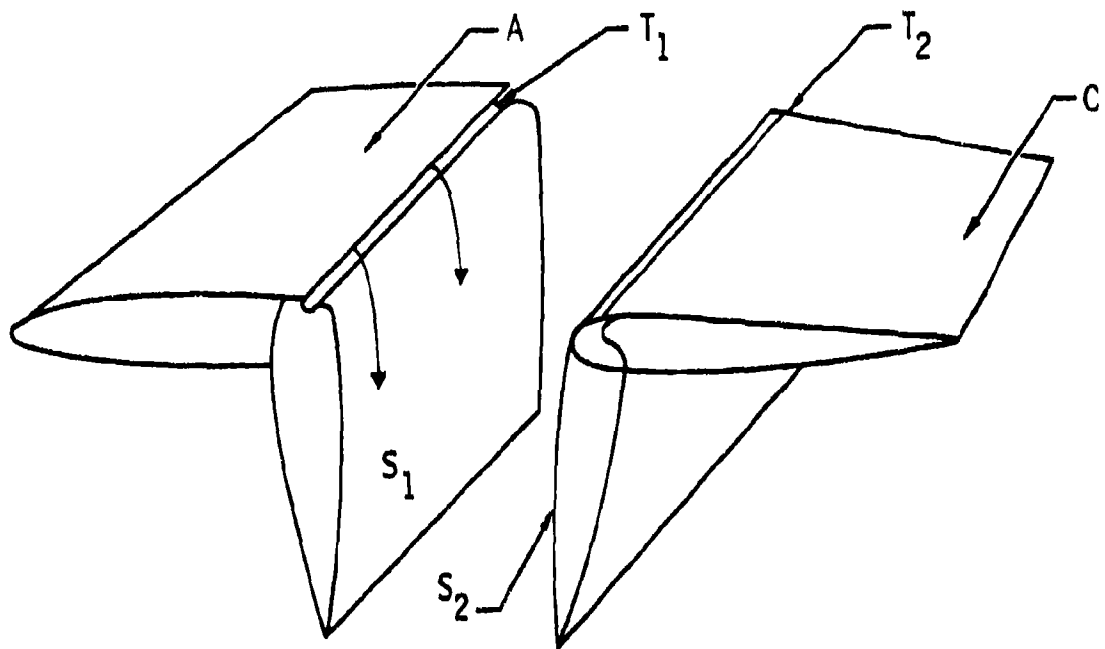


Figure 1. XFV-12A Prototypic Augmenter Configuration

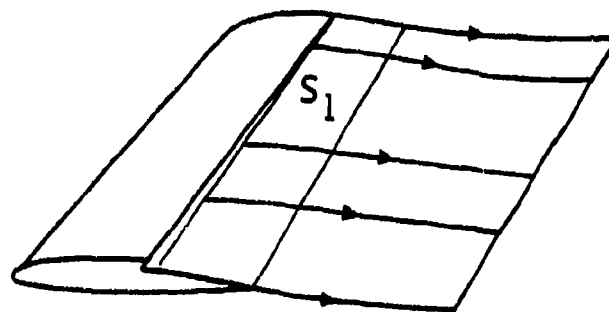


Figure 2. Upper Surface Blowing Boundary Layer Control-Supercirculation Wing

In Figure 3, the prototypic geometry for the flow over the surfaces S_1 is shown with the prescribed initial and boundary velocity distributions schematically indicated. A curvilinear coordinate system is used which consists of conveniently oriented lines in the surface and normals to them as shown in the figure. The initial distribution of velocities is specified along some line $x=x_0$ =constant say, which may be the jet exit. In the wall jet/boundary layer approximation the appropriate equations describing the wall jet flow are:

Continuity

$$(\partial/\partial x)(h_2 u) + (\partial/\partial z)(h_1 w) + (\partial/\partial y)(h_1 h_2 v) = 0. \quad (1)$$

x-Momentum

$$\frac{u}{h_1} \frac{\partial u}{\partial x} + \frac{w}{h_2} \frac{\partial u}{\partial z} + v \frac{\partial u}{\partial y} - uwK_1 + w^2K_2 = -\frac{1}{\rho h_1} \frac{\partial p}{\partial x} + \frac{\partial}{\partial y} \left(\nu \frac{\partial u}{\partial y} - \overline{u'v'} \right). \quad (2)$$

z-Momentum

$$\frac{u}{h_1} \frac{\partial w}{\partial x} + \frac{w}{h_2} \frac{\partial w}{\partial z} + v \frac{\partial w}{\partial y} - uwK_2 + u^2K_1 = -\frac{1}{\rho h_2} \frac{\partial p}{\partial z} + \frac{\partial}{\partial y} \left(\nu \frac{\partial w}{\partial y} - \overline{w'v'} \right). \quad (3)$$

Here, h_1 and h_2 are metric coefficients and are functions of x and z , and the parameters K_1 and K_2 are known as the geodesic curvatures of the curves $z = \text{constant}$ and $x = \text{constant}$, respectively.

The boundary conditions for Eqs. (1) through (3) for zero mass transfer at the wall and compatibility with the external flow are

$$y = 0$$

$$u, w, v = 0$$

$$y \rightarrow \infty$$

$$u \rightarrow u_e(x, z)$$

$$w \rightarrow w_e(x, z). \quad (4)$$

SC79-3525-A

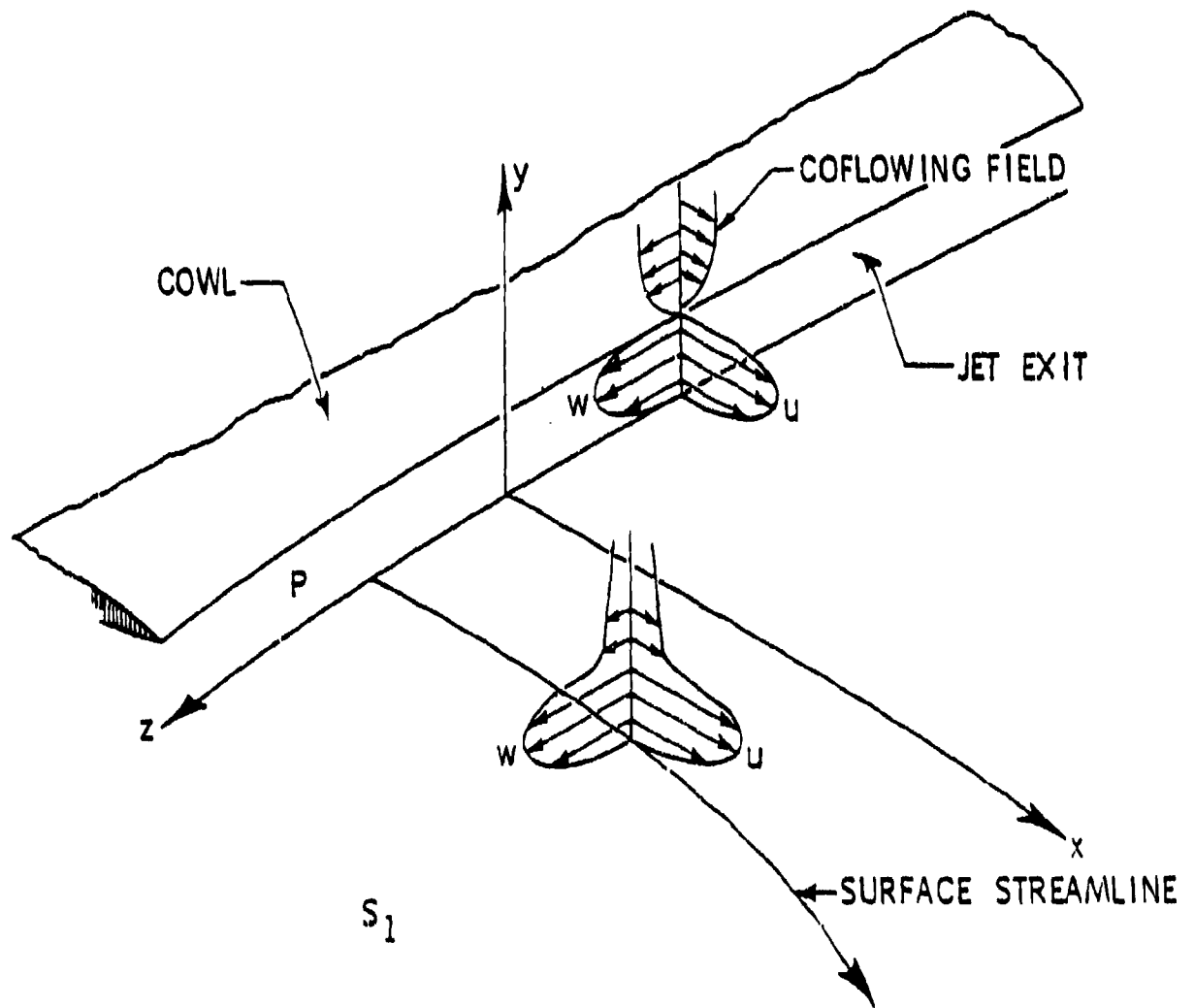


Figure 3. Physical System and Flow Schematic

As indicated earlier, the previous equations are transformed by defining

$$x = x \quad z = z \quad \eta = (u_e / \nu s_1)^{1/2} y \quad (5)$$

and introducing a two-component vector potential given by

$$h_2 u = \frac{\partial \psi}{\partial y} \quad h_1 w = \frac{\partial \phi}{\partial y} \quad h_1 h_2 v = - \left(\frac{\partial \psi}{\partial x} + \frac{\partial \phi}{\partial z} \right) . \quad (6)$$

Here s_1 , which denotes the arc length along the x coordinate, is defined by

$$s_1 = \int_0^x h_1 dx . \quad (7)$$

In addition, the dimensionless variables f and g related to ψ and ϕ are defined by

$$\psi = (u_e \nu s_1)^{1/2} h_2 f(x, z, \eta) \quad (8a)$$

$$\phi = (u_e \nu s_1)^{1/2} h_1 (w_e / u_e) g(x, z, \eta) . \quad (8b)$$

The parameters K_1 and K_2 in Eqs. (2) and (3) are defined by

$$K_1 = - \frac{1}{h_1 h_2} \frac{dh_1}{dz} \quad \text{and} \quad K_2 = - \frac{1}{h_1 h_2} \frac{dh_2}{dx} .$$

3.0 Eddy Viscosity Model

Equations (2) and (3) contain the Reynolds shear stress terms $-\overline{u'v'}$ and $-\overline{v'w'}$. These quantities must be further characterized in order to

solve Eqs. (1)-(3). To achieve this closure, suitable eddy viscosity models are employed. In two dimensions, such simulations have been successfully used by Dvorak,⁶ Ramaprian,⁷ and Wilson and Goldstein.⁸ On a heuristic basis subject to future experimental validation, extension to three-dimensional flows was effected analogous to an approach used for the same purpose in connection with boundary layer flows in Ref. 5. For the wall jet case, we assume that the curvature effects are modeled in terms of the principal curvatures in the x and z directions. With this viewpoint, the Reynolds shear stress terms are assumed to be in the form

$$\begin{aligned}\overline{-u'v'} &= \epsilon u_y = \bar{\epsilon} \left[1 - \frac{uK_2}{1+K_2y} \frac{1}{u_y} \right] u_y \\ \overline{-v'w'} &= \epsilon w_y = \bar{\epsilon} \left[1 - \frac{wK_1}{1+K_1y} \frac{1}{w_y} \right] w_y\end{aligned}\quad (9)$$

The quantity $\bar{\epsilon}$ in Eq. (9) is the eddy viscosity of the corresponding plane flow. It is assumed to be the same in both the x and z directions and is represented by a two-layer model. The second term inside the bracket in Eq. (9) is due to curvature where K_1 and K_2 denote the radius of curvature of $z=\text{constant}$ and $x=\text{constant}$ lines. Similar curvature terms for two-dimensional wall jets have been used by Dvorak⁶ and Wilson.⁸ Referring to Figure 4, the structure of the two-layer eddy viscosity model is as follows:

First Layer

$$\bar{\epsilon} = (0.435 y)^2 \sqrt{u_y^2 + w_y^2} \quad 0 \leq y \leq y^*$$

Second Layer

$$\bar{\epsilon} = (0.125 y_1)^2 \sqrt{u_y^2 + w_y^2} \quad y \geq y^* \quad (10)$$

where at $y = y_1$

$$\frac{|\sqrt{u_e^2 + w_e^2} - \sqrt{u^2 + w^2}|}{\sqrt{u_e^2 + w_e^2}} \approx 0.01$$

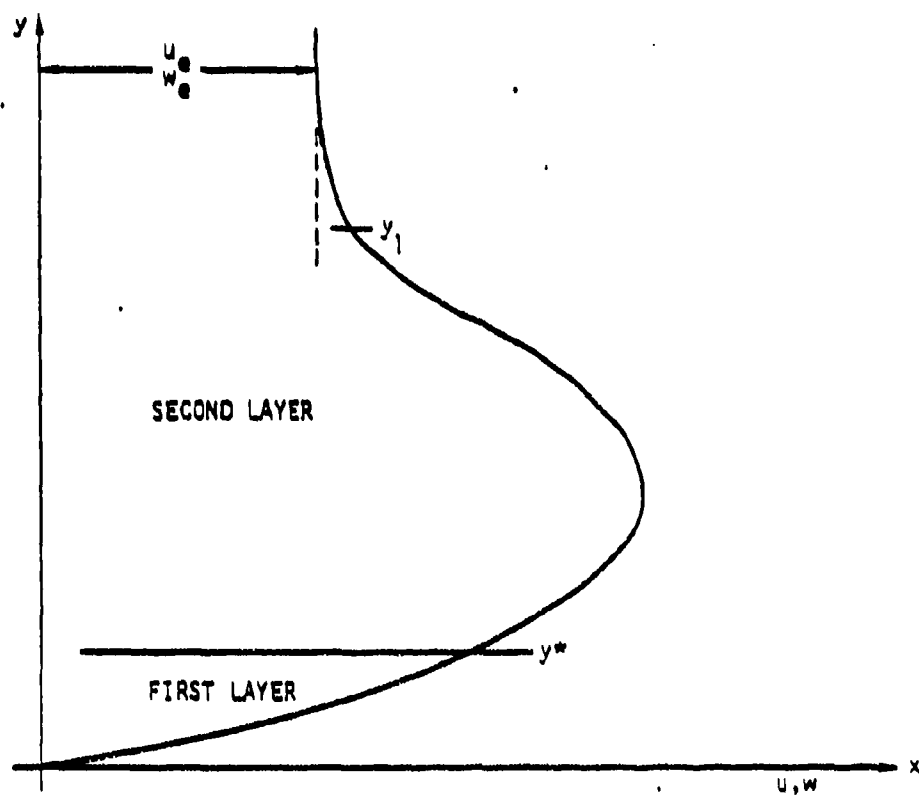


Figure 4. Two-Layer Eddy Viscosity Model

and y^* is obtained by imposing continuity in $\bar{e}$ at $y=y^*$. This yields $y^* = \frac{0.125}{0.435} y_1$. Two-layer eddy viscosity models similar to Eq. (10) have been employed by Ramaprian⁷ for two-dimensional wall jets which give good comparisons with experiments. In these cases, the term w_y appearing inside the square root in Eq. (10) is zero.

With the concept of eddy viscosity and with the previous transformed variables, it can be shown that the system of Eqs. (1) through (4) can be written as

x-Momentum Equation

$$\begin{aligned} (bf'')' + P_1 ff'' + P_2 [1 - (f')^2] + P_3 [1 - f'g'] \\ + P_6 f''g + P_8 [1 - (g')^2] \\ = xP_{10} \left[f' \frac{\partial f'}{\partial x} - f'' \frac{\partial f}{\partial x} + P_7 \left(g' \frac{\partial f}{\partial z} - f'' \frac{\partial g}{\partial z} \right) \right]. \end{aligned} \quad (11)$$

z-Momentum Equation

$$\begin{aligned} (bg'')' + P_1 fg'' + P_4 (1 - f'g') + P_3 [1 - (g')^2] \\ + P_6 gg'' + P_9 [1 - (f')^2] \\ = xP_{10} \left[f' \frac{\partial g'}{\partial x} - g'' \frac{\partial f}{\partial x} + P_7 \left(g' \frac{\partial g'}{\partial z} - g'' \frac{\partial g}{\partial z} \right) \right] \end{aligned} \quad (12)$$

$$\eta = 0 \quad f = g = g' = g'' = 0 \quad (13a)$$

$$\eta = \eta_\infty \quad f' = g' = 1. \quad (13b)$$

Here the primes denote differentiation with respect to η , and

$$b = 1 + \epsilon^+ \quad \epsilon^+ = \epsilon/v \quad f' = u/u_e \quad g' = w/w_e. \quad (14a)$$

The coefficients P_1 to P_{10} are functions of u_e , w_e , h_1 , h_2 , K_1 , and K_2 and are given by the following formulas:

$$P_1 = (M+1)/2 - s_1 K_2 \quad P_2 = M \quad P_3 = R$$

$$P_4 = \left(\frac{u_e}{w_e} \right) Q - s_1 K_2 \quad P_5 = \frac{w_e}{u_e} (N - s_1 K_1)$$

$$P_6 = R + \frac{w_e}{2u_e} \left(\frac{1}{h_1} \frac{\partial s_1}{\partial z} - N \right) - \left(\frac{w_e}{u_e} \right) s_1 K_1$$

$$P_7 = \frac{h_1}{h_2} \frac{w_e}{u_e} \quad P_8 = \left(\frac{w_e}{u_e} \right)^2 s_1 K_2$$

$$P_9 = \left(\frac{u_e}{w_e} \right) s_1 K_1 \quad P_{10} = \frac{s_1}{\pi h_1} \quad (14b)$$

$$M = \frac{s_1}{u_e h_1} \frac{\partial u_e}{\partial x} \quad N = \frac{s_1}{u_e h_2} \frac{\partial u_e}{\partial z}$$

$$Q = \frac{s_1}{u_e h_1} \frac{\partial w_e}{\partial x} \quad R = \frac{s_1}{u_e h_2} \frac{\partial w_e}{\partial z}$$

In order to solve eqs. (11) through (13), initial conditions are required at a starting plane. In the case of the boundary layer problem, the initial conditions at $x=0$ and $z=0$ planes are obtained by solving the limiting form of Eqs. (11) and (12). For a wall jet, initial velocity

profiles are prescribed at some downstream $x=x_0$ plane and along the $z=0$ plane, "attachment line" equations are solved. The attachment line equations are obtained by differentiating the z -momentum equation with respect to z and setting

$$w = \frac{\partial p}{\partial z} = \frac{\partial u}{\partial z} = \frac{\partial v}{\partial z} = \frac{\partial^2 w}{\partial z^2} = 0.$$

The resulting attachment line equations valid at the $z=0$ plane are

$$(bf'')' + P_1 f f'' + P_2 [1 - (f')^2] + P_3 g f'' = x P_{10} \left[f' \frac{\partial f'}{\partial x} - f'' \frac{\partial f}{\partial x} \right] \quad (15)$$

$$(bg'')' + P_1 f g'' + P_4 (1 - f' g') + P_3 [1 - (g')^2] + P_3 g g'' = x P_{10} \left[f' \frac{\partial g'}{\partial x} - g'' \frac{\partial f}{\partial x} \right]. \quad (16)$$

Here, g' is defined as w_z/w_{zz} , and its definition corresponds to L'Hospital's rule applied to the expression for g' used previously.

4.0 Finite Difference Equations

First, reduce the system (11)-(12) to the first order system

$$f' = u \quad (17)$$

$$u' = v \quad (18)$$

$$g' = w \quad (19)$$

$$w' = t \quad (20)$$

$$\begin{aligned}
 (bv)' + P_1 fv + P_2 (1 - u^2) + P_5 [1 - uw] \\
 + P_6 v g + P_8 [1 - w^2] = x P_{10} \left[u \frac{\partial u}{\partial x} - v \frac{\partial f}{\partial x} + P_7 \left(w \frac{\partial u}{\partial z} - v \frac{\partial g}{\partial z} \right) \right] \quad (21)
 \end{aligned}$$

$$\begin{aligned}
 (bt)' + P_1 ft + P_4 (1 - uw) + P_3 (1 - w^2) + P_6 gt \\
 + P_9 (1 - u^2) = x P_{10} \left[u \frac{\partial w}{\partial x} - t \frac{\partial f}{\partial x} + P_7 \left(w \frac{\partial w}{\partial z} - t \frac{\partial g}{\partial z} \right) \right] \quad (22)
 \end{aligned}$$

Using the notation of Ref. 1 associated with the box method described there and in Refs. 3, 4, 5, and 9, we let

$$\begin{aligned}
 x_0 &= \text{constant} & x_n &= x_{n-1} + k_n & n &= 1, 2, \dots, N \\
 z_0 &= 0 & z_i &= z_{i-1} + r_i & i &= 1, 2, \dots, I \\
 \eta_0 &= 0 & \eta_j &= \eta_{j-1} + h_j & j &= 1, 2, \dots, J.
 \end{aligned}$$

Then, using the box method, we have

$$\frac{f_j^{n,i} - f_{j-1}^{n,i}}{h_j} = u_{j-1/2}^{n,i} \quad (23)$$

$$\frac{u_j^{n,i} - u_{j-1}^{n,i}}{h_j} = v_{j-1/2}^{n,i} \quad (24)$$

$$\frac{g_j^{n,i} - g_{j-1}^{n,i}}{h_j} = w_{j-1/2}^{n,i} \quad (25)$$

$$\frac{w_j^{n,i} - w_{j-1}^{n,i}}{h_j} = t_{j-1/2}^{n,i} \quad (26)$$

We use the notation

$$\bar{P} = P_{j-1/2}^{n-1/2, i-1/2} \equiv P_{i-1/2}^{n-1/2}$$

and

$$\bar{v}_j = \frac{1}{4} \left(v_j^{n,i} + v_j^{n,i-1} + v_j^{n-1,i-1} + v_j^{n-1,i} \right)$$

$$\bar{u}_n = \frac{1}{2} \left(u_{j-1/2}^{n,i} + u_{j-1/2}^{n,i-1} \right)$$

$$\bar{u}_i = \frac{1}{2} \left(u_{j-1/2}^{n,i} + u_{j-1/2}^{n-1,i} \right).$$

Equation (21) becomes, with the box centered at $(x_{n-1/2}, z_{i-1/2}, \eta_{j-1/2})$

$$\begin{aligned} & (\bar{b}_j \bar{v}_j - \bar{b}_{j-1} \bar{v}_{j-1}) / h_j \\ &= -\bar{P}_1 (\bar{f} \bar{v})_{j-1/2} - \bar{P}_2 (1 - \bar{u}_{j-1/2}^2) - \bar{P}_5 (1 - \bar{u}_{j-1/2} \bar{w}_{j-1/2}) \\ & - \bar{P}_6 (\bar{v} \bar{g})_{j-1/2} - \bar{P}_8 (1 - \bar{w}_{j-1/2}^2) \\ & + x_{n-1/2} \bar{P}_{10} \left\{ u_{j-1/2} \frac{(\bar{u}_n - \bar{u}_{n-1})}{k_n} - \bar{v}_{j-1/2} \frac{(\bar{f}_n - \bar{f}_{n-1})}{k_n} \right. \\ & \left. + \bar{P}_7 \left[\bar{w}_{j-1/2} \frac{(\bar{u}_i - \bar{u}_{i-1})}{r_i} - \bar{v}_{j-1/2} \frac{(\bar{g}_i - \bar{g}_{i-1})}{r_i} \right] \right\}. \end{aligned} \quad (27)$$

Equation (22) and the attachment line equations (13)-(14) are discretized similarly. Details of the procedure are given in Ref. 9.

The solution procedures involve the following steps:

- (1) Solve the attachment line equations (15)-(16) with boundary conditions (13) at $x = x_1$ and $z = 0$ assuming initial conditions on $x = x_0$.
- (2) March in the z -direction along the plane $x = x_1$ and solve equations (17)-(22) with boundary conditions (13) for the unknowns (f, u, v, g, w, t) .
- (3) Repeat steps (1) and (2) for the next x -plane, $x = x_2$, and so on.

The most efficient way to solve the finite difference equations is to use a pseudo-Newton's relaxation scheme. These equations may be written as a system of nonlinear algebraic equations by writing

$$\phi(\underline{u}) = 0$$

where

$$\underline{u} = (f_j^{n,i}, u_j^{n,i}, v_j^{n,i}, g_j^{n,i}, w_j^{n,i}, t_j^{n,i})_{j=0}^J.$$

Then, the relaxed Newton's method becomes

$$\frac{\partial \phi}{\partial \underline{u}}^{(v-1)} \delta \underline{u}^{(v-1)} = - \phi(\underline{u}^{(v-1)}) \quad (28a)$$

$$\underline{u}^{(v)} = \underline{u}^{(v-1)} + \omega \delta \underline{u}^{(v-1)} \quad (28b)$$

for $v = 1, 2, \dots$

The method is said to have converged when

$$\|\delta u^{(v-1)}\|_{\infty} \leq \epsilon \text{ (a prescribed error tolerance) .}$$

We call Eq. (28) a pseudo-Newton's method because we linearize the b terms in Eqs. (21) and (22) by evaluating them at $v-2$ before computing the Jacobian matrix, $\partial \Phi / \partial u$. Consequently, this algorithm will not be quite quadratically convergent. We, therefore, employ relaxation ($\omega \neq 1$) to accelerate it. Remarkably, underrelaxation ($\omega < 1$) works very well, while overrelaxation ($\omega > 1$) diverges. Values of ω of 0.5, 0.6, 0.7, 0.8, and 0.9 all give good results with $\omega = 0.7$ found to be the overall best for some of our computational experiments.

An important feature of Keller's box method is that the Jacobian matrix can be put into block tridiagonal form and very efficient elimination schemes can be employed for solving Eq. (28a).

Minor Difficulties with the Numerical Algorithm

When starting at $x = 1$ with supplied completely merged wall jet velocity profiles described in Fig. 5, unnatural oscillations developed in the solution. This difficulty was eliminated completely by employing the following "trick." The first 10 mesh points in the x -direction were set at $k_n = 10^{-4}$. For the first five planes in the x -direction and all points in the z -direction in these planes, an average value was used for past points, i.e.,

$$f_j^{n-1,1} = 0.5 (f_j^{n-1,1} + f_j^{n-2,1}) \quad , \quad f_j^{n,1-1} = 0.5 (f_j^{n,1-1} + f_j^{n,1-2}) \quad ,$$

and

$$f_j^{n-1,1-1} = 0.5 (f_j^{n-1,1-1} + f_j^{n-2,1-1}) \quad .$$

SC78-3622

EXTERNAL VELOCITY

$$u_e = u_0 x^{-n} \cos \theta$$

$$w_e = u_0 x^{-n} \sin \theta$$

$$\theta = C_2 (z_{TP} - z) \frac{\pi}{2}$$

INITIAL PROFILE

$$\frac{u}{u_e} = \begin{cases} C_2 (e^{-y-1} + C_3 y) \\ 1 + \left(\frac{u_{MAX}}{u_e} - 1 \right) e^{-0.1 (y - y_{MAX})^2} \end{cases}$$

$$\frac{w}{w_e} = \text{SAME AS } \frac{u}{u_e} \text{ PROFILE}$$

● SMALL CROSS FLOW IS ACHIEVED BY SETTING C IN θ EXPRESSION VERY SMALL (10^{-15})

● $\frac{u_{MAX}}{u_e}, \frac{w_{MAX}}{w_e}$ CAN BE PRESCRIBED AS A FUNCTION OF z

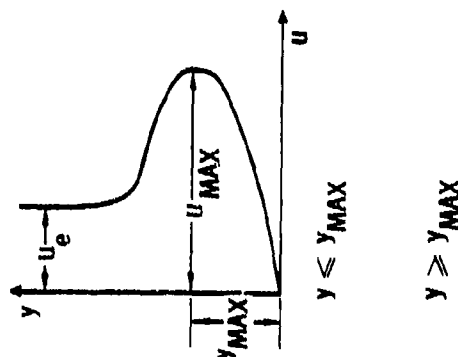


Figure 5. Initial and Boundary Conditions

Beginning with the sixth x-plane, the averaging was eliminated (the standard algorithm was employed). At the eleventh x-plane a geometric mesh-stretching algorithm of the following form was used:

$$k_n = 1.2k_{n-1}, \quad n = 11, 12, 13, \dots$$

No such stretching has been employed in the z-direction, but in the future it may also be required for rapidly changing profiles. It should be noted that our averaging algorithm was required in both the x and z directions to remove all oscillations.

A mesh refinement algorithm is used which adds or deletes points depending on the relative local variation in the truncation error of the difference equations. Roughly 80 grid points in the n-direction and 11 grid points in the z-direction are employed.

5.0 Results

Computations were performed on the Berkeley CDC 7600 machine. A typical calculation required about 6 minutes of CPU time. Figure 5 indicates the external and initial velocity distributions which have been used as a basis for our calculations. The parameter θ was introduced as shown to vary the initial cross flow while keeping the total velocity constant as a rough simulation of a fixed supply of engine mass flow. The velocity profile was selected to have a characteristic fully developed character associated with turbulent wall jet flows. Also, the surface shape is chosen planar for simplicity in the ensuing analysis. Future aspects of this effort should consider the "eating up" of the potential core which is assumed to occur upstream of the initial station of this analysis. The parameters C_2 and C_3 were chosen to provide slope and value continuity of the profile at $y = y_{\max}$. For $y > y_{\max}$ the profile has a half Gaussian character associated with a free jet. For $y < y_{\max}$ the profile has a boundary layer character. In the examples, the u and w initial profiles were assumed to be identical. Moreover, the θ

distribution was selected to be qualitatively similar to that observed by rake surveys representative of the XfV-12A. The zero cross flow case was achieved by setting C to 10^{-15} .

Figure 6 demonstrates decay of the peak velocity with the standardized distributions of Fig. 5, with and without cross flow. It is evident from the figure that initial cross flow has a dramatic effect on enhancing the decay of the maximum velocity. In the calculations, the decay exponent n in the external velocities is assumed as $1/2$, roughly in accord with a value obtained from a two-dimensional line sink simulating inflow originally proposed by G.I. Taylor.¹⁰ Both streamwise increasing and decreasing cross flow cases are shown. In the line sink model, the inflow at each position $x = x_0$ along the jet boundary is determined by streamwise rate of change of mass flux according to conservation of mass applied on a rectangular control surface in the jet layer. From an observation point P in the external flow which is assumed inviscid and irrotational, the velocity potential can therefore be represented as an isolated sink whose intensity is proportional to the inflow at $x = x_0$. The cumulative effect at P of all such sinks at $x = x_1$ is obtained by a superposition integral of all these contributions giving a line sink representation. In a more realistic model, these external velocity distributions should be corrected for three-dimensionality and elliptic interaction with the wall jet. A calculation of this type would be a more accurate representation than the present approach of planform and surface curvature effects. These developments are strongly recommended for future application. In this connection, we recognize that the present means of simulating taper, sweepback, and spanwise pressure gradients is solely through cross flow adjustment.

The three-dimensional inviscid potential ϕ can be characterized by surface sink distribution of the form (see Fig. 7)

$$\phi(x, y, z) = \frac{1}{4\pi} \iint_S \frac{\sigma(\xi, \zeta) d\xi d\zeta}{\sqrt{(x-\xi)^2 + y^2 + (z-\zeta)^2}} \quad (29)$$

SC79-3524-A

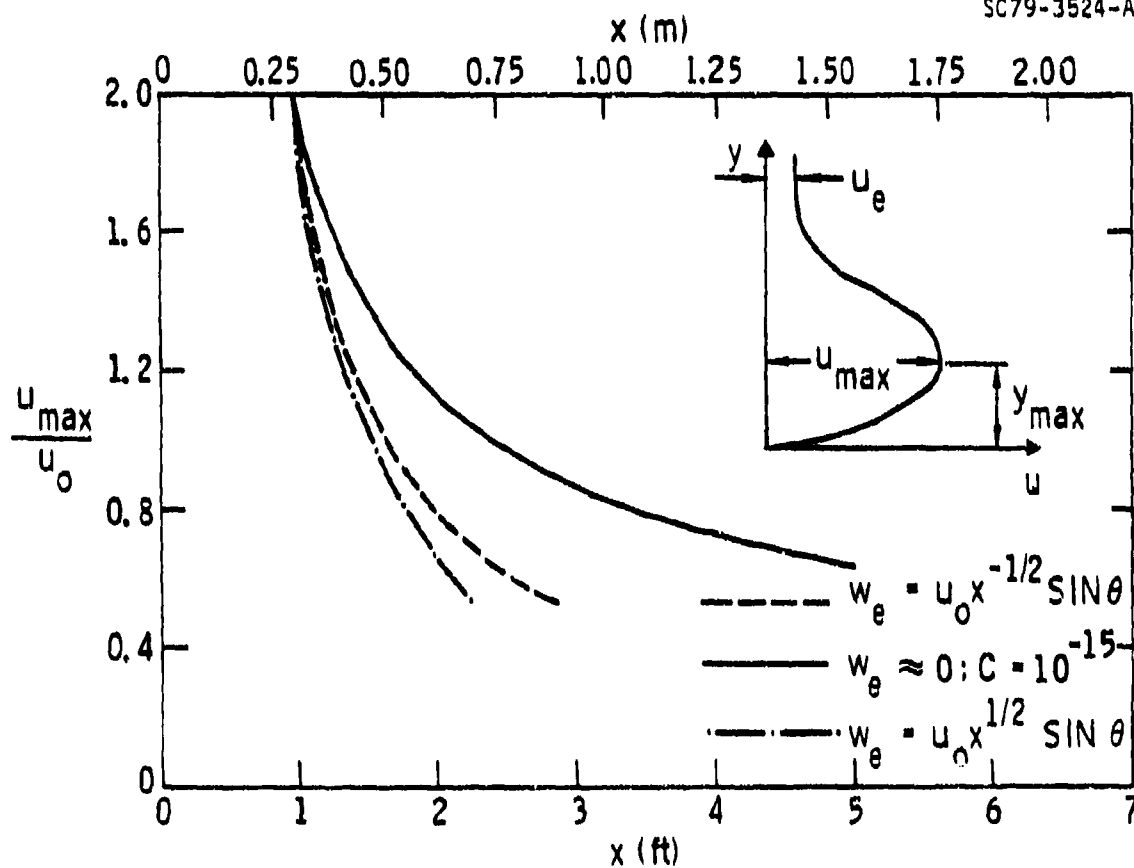


Figure 6. Effect of Cross Flow on Jet Growth, $\frac{u_{\max}}{u_e} = 2 - |.5 - z|$,

$u_e = u_0 x^{-1/2} \cos \theta$, $\theta = \frac{\pi}{2} z(z_{\text{tip}} - z)$, Standard Initial Profile (See Fig. 5), $z = z_1 = z_{\text{tip}}/2$.

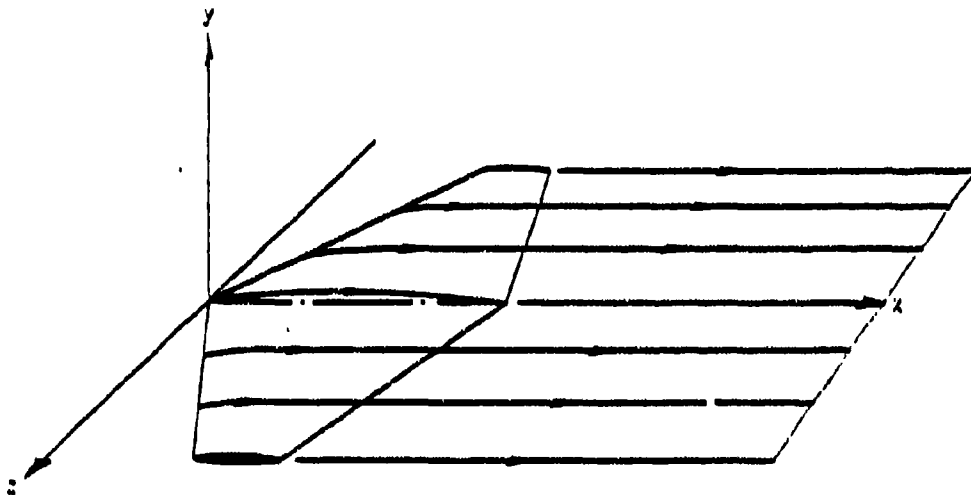


Figure 7. Upper Surface Blown Thrust Augmented Wing (TAW)

where S the area of integration refers to the total jet area on and off the wing. The quantity σ is the sink strength obtained by matching with an "outer limit" of the second order solution for the velocity normal to the body appearing in the viscous inner wall jet solution. Generalizing the previous concepts, Eq. (29) is obtained by superposition of a surface distribution of elementary sink solutions of the form $\phi = \frac{1}{4\pi R}$ where $R^2 = (x-\xi)^2 + y^2 + (z-\zeta)^2$. Note that the surface distributions do not interact in the sense that they produce no ϕ_y at locations other than their own (ξ, ζ) . Hence $\phi_y \sim$ local inflow analogous to that previously described for the two-dimensional case. The quantity σ for two-dimensional boundary layers is analogous to the streamwise gradient of the displacement thickness $\delta'(x)$. To include lifting surface effects, a surface doublet or vortex distribution should be added to Eq. (29). The local vortex strength can also be determined by matching.

The inflow velocity related to the sink intensity σ in Eq. (29) is in turn a function of the entrainment. This quantity is also significant from the standpoint of the tradeoff between skin friction, BLC, and rapid acceleration of the secondary in compact three-dimensional thrust augmenting ejectors such as those employed on the XFV-12A and upper surface blowing.

In Fig. 8, the comparison between cross flow and the absence of it gives the indicated nominal entrainment variations with streamwise distance where nominal entrainment Q is defined in the figure. In spite of the appreciable increase in decay of the maximum value of u shown in Fig. 6, and resultant shear stress in Fig. 9, only a slight difference in entrainment quantity and rate is shown in Fig. 8. The difference in maximum velocities which are similar for w , the spanwise component, are presumably related to the enhanced dissipation associated with cross flow and that implied by the eddy viscosity model. The lack of a corresponding increase in entrainment rate may be due to nonlinear compensating effects built into the turbulence model and cannot be readily explained on an intuitive basis at this time. In this connection, other calculations should be performed for which the streamwise component of the

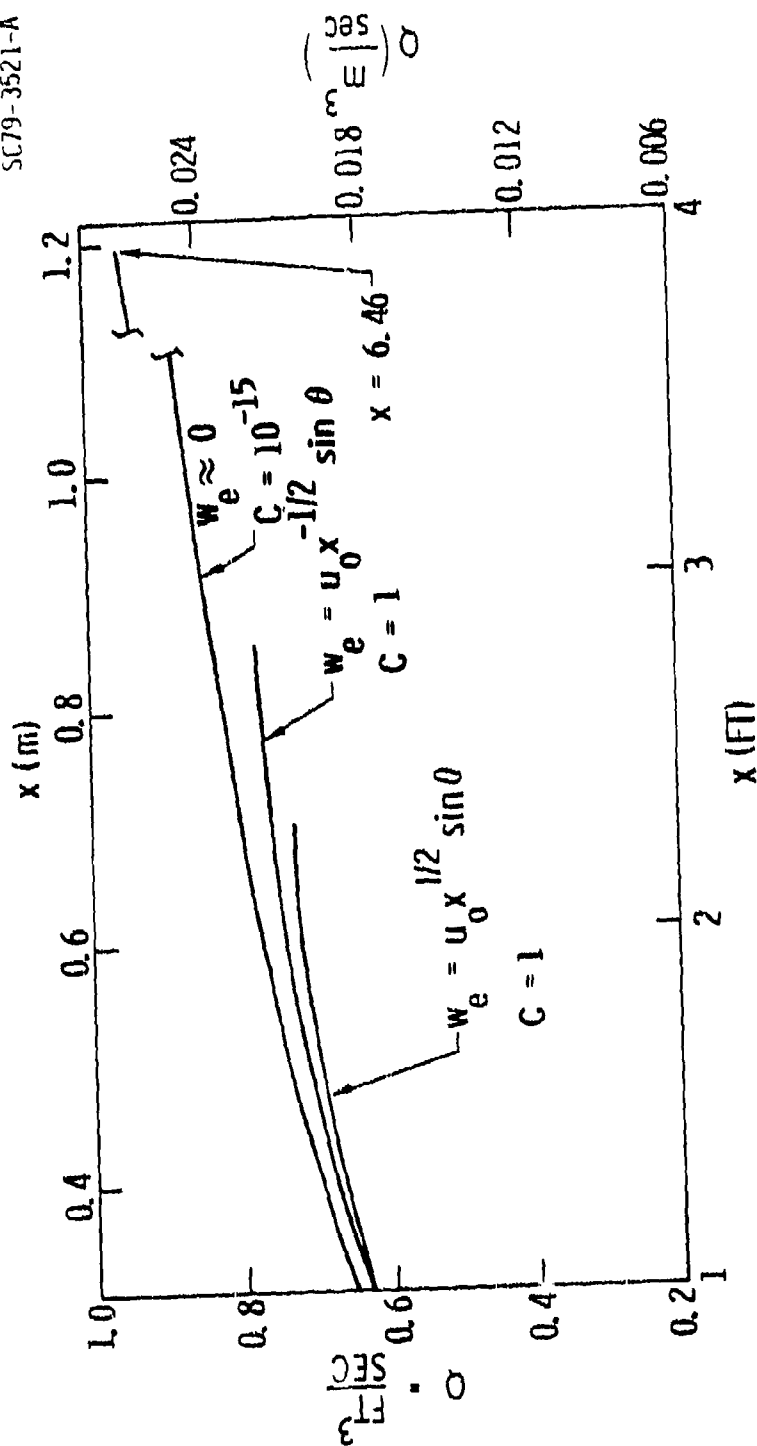


Figure 8. Effect of Cross Flow on Nominal Entrainment, Q with $u_e = u_0 x^{-1/2} \cos \theta$,

$$Q = \int_0^\infty \int_0^{z_{tip}} u dy dz, \quad \theta = C \frac{\pi}{2} z (z_{tip} - z), \quad \frac{u_{max}}{u_e} = 2 - |.5 - z|,$$

(Standard Initial Profile) $z = z_1 = z_{tip}/2$.

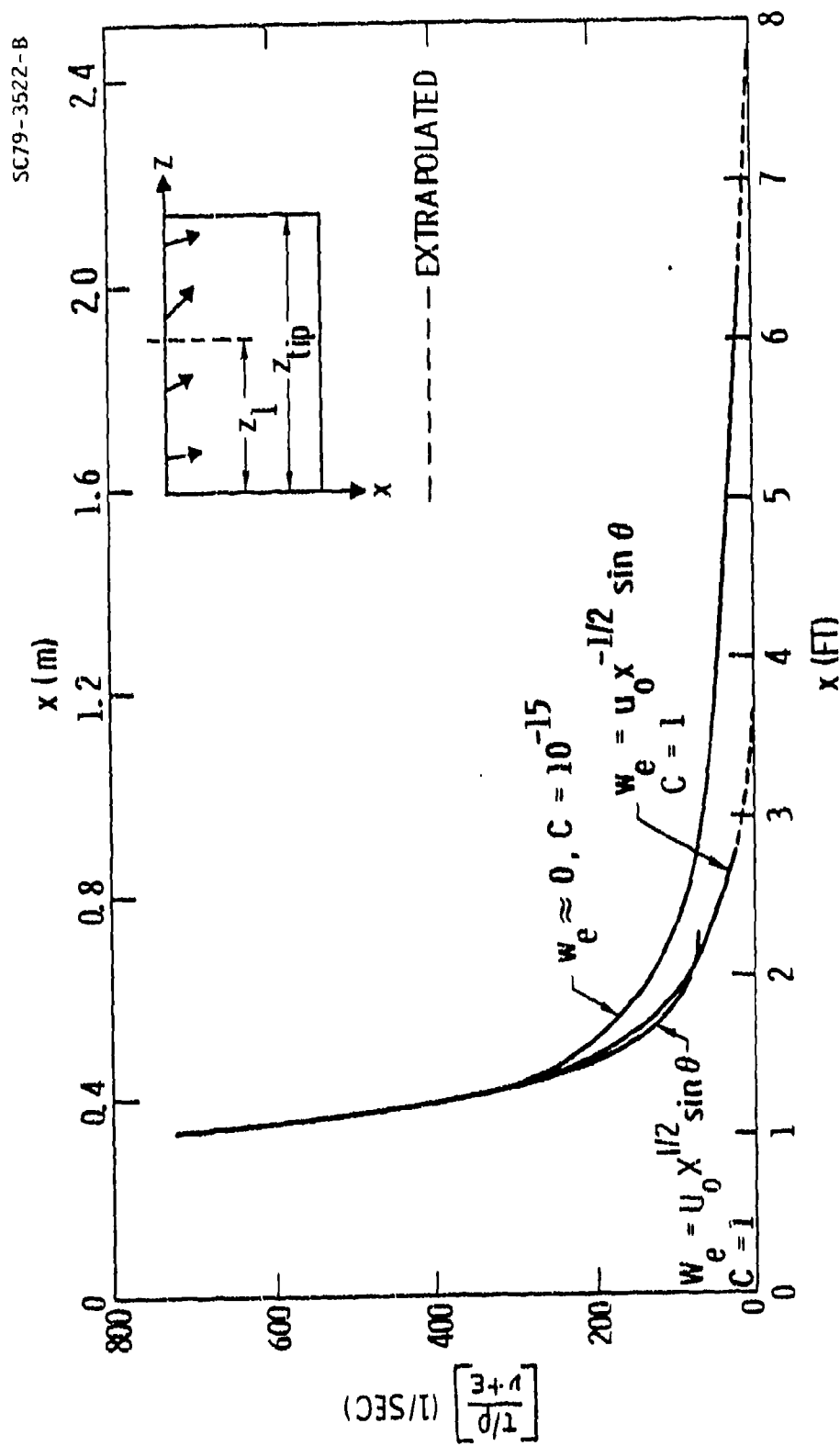


Figure 9. Effect of Cross Flow on Reduced Shear Stress, $u_e = u_0 x^{-1/2} \cos \theta$,

$$\theta = C_2^{\frac{\pi}{2}} (z_{tip} - z), \text{ Standard Initial Profile, } (z = z_1 = z_{tip}/2),$$

$$\frac{u_{max}}{u_e} = 2 - |.5 - z|.$$

initial velocity is held fixed rather than its overall magnitude on introduction of cross flow. Furthermore, trends involving an increase in decay rate of peak velocity with an increase in entrainment rate that occur in two-dimensional free jets based on conservation laws and similitude must be reassessed for non-similar three-dimensional wall jets such as those considered herein. Here, the variable streamline direction through the jet layer must be considered as well as the peak of the resultant velocity $q_{\max}$ rather than those of the individual velocity components. If w is the local streamline direction on the wall, under certain

circumstances, $\frac{\partial q_{\max}}{\partial s}$ may become more negative with increasing Q . It should be noted that the expression for entrainment Q given in Fig. 8 assumes that $w = 0$ at the tip $z = z_{\text{tip}}$. If this is not the case, an additional term must be added to this relation. As in Figs. 9 and 10, qualitatively similar behavior is obtained for the case in which $w \sim x^{1/2}$.

Associated with the previous results, Fig. 10 shows the effect of cross flow on jet spreading rate related to $y_{\max}$. As previously, only small differences are indicated for the cases considered. In Fig. 11, however, an important upstream movement of the nominal separation line is indicated with the introduction in cross flow. Here, nominal separation is defined to occur where $\tau = 0$. A more pertinent definition is $\frac{\partial q_n}{\partial y} = 0$ where q_n is the velocity component normal to an envelope of the surface streamlines. Implementation of the latter definition is envisioned in follow-on studies involving primary/secondary entrainment interactions. This result is significant with respect to penalties associated with taper and sweep in three-dimensional ejector diffusers.

In Fig. 12, another important consequence of cross flow is examined in connection with the surface streamline pattern. In the figure, two cases are compared involving differing amounts of cross flow. Lines such as AB and A'B' represent typical streamlines for the two different cross flow cases in which the downstream direction is in the direction of the arrows in the rectangular area corresponding roughly to the planform of a

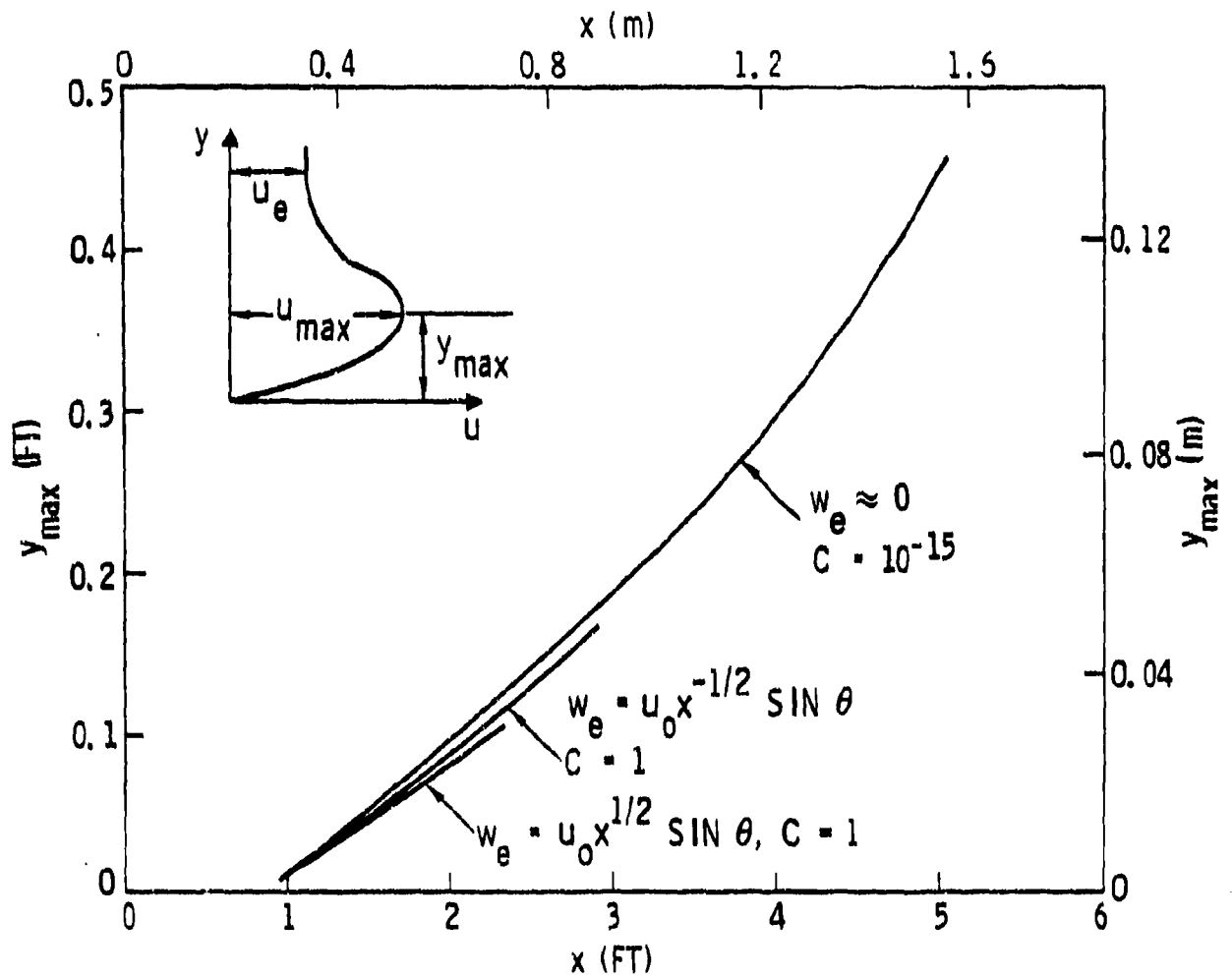


Figure 10. Effect of Cross Flow on Jet Spreading, $u_e = u_0 x^{-1/2} \cos \theta$,
 $\theta = C \frac{\pi}{2} (z_{tip} - z)$, Standard Initial Profile at Midspan
 $(z = z_{tip}/2)$, $\frac{u_{max}}{u_e} = 2 - |.5 - z|$.

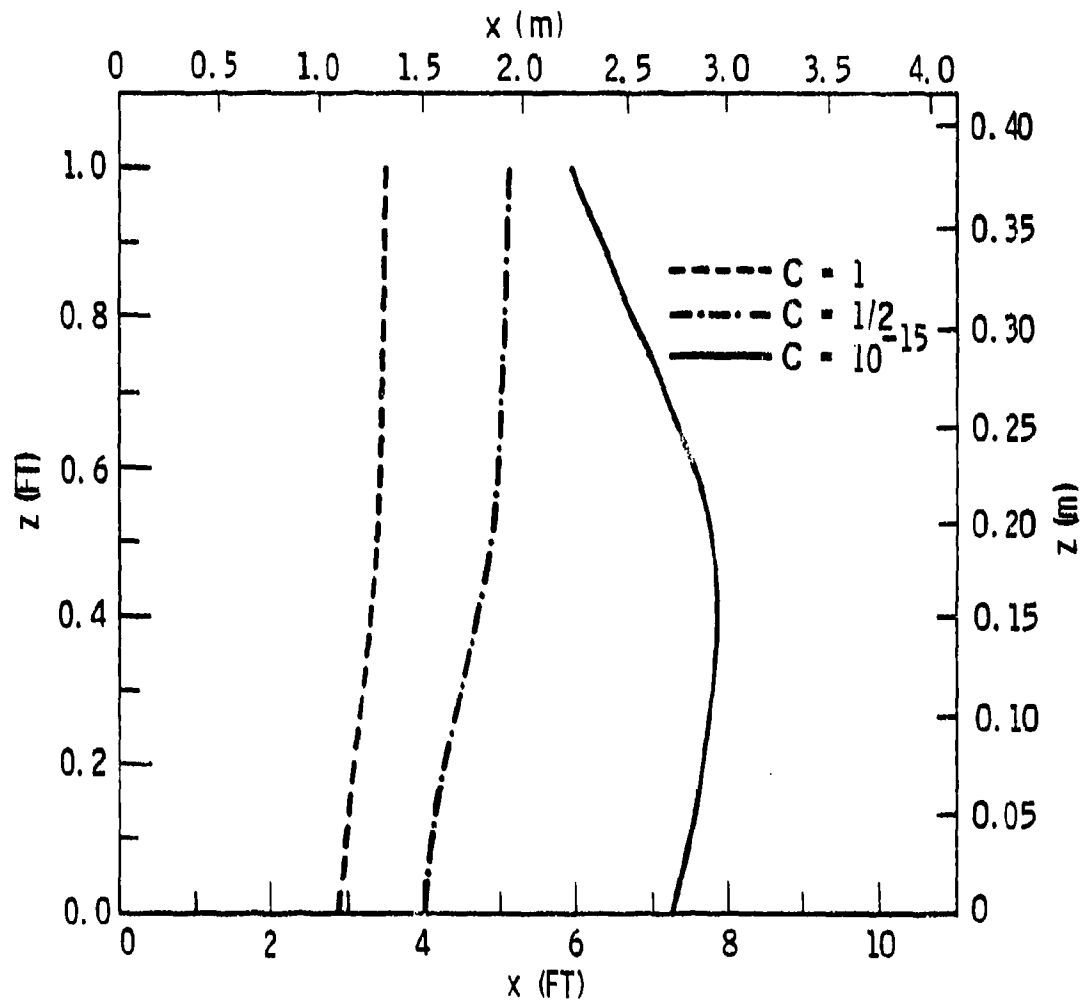


Figure 11. Effect of Cross Flow on the Locus of Nominal Separation, $u_e = u_0 x^{-1/2} \cos \theta$, $w_e = u_0 x^{-1/2} \sin \theta$,
 $\theta = C \frac{\pi}{2} (z_{tip} - z)$, $\frac{u_{max}}{u_e} = 2 - |.5 - z|$.

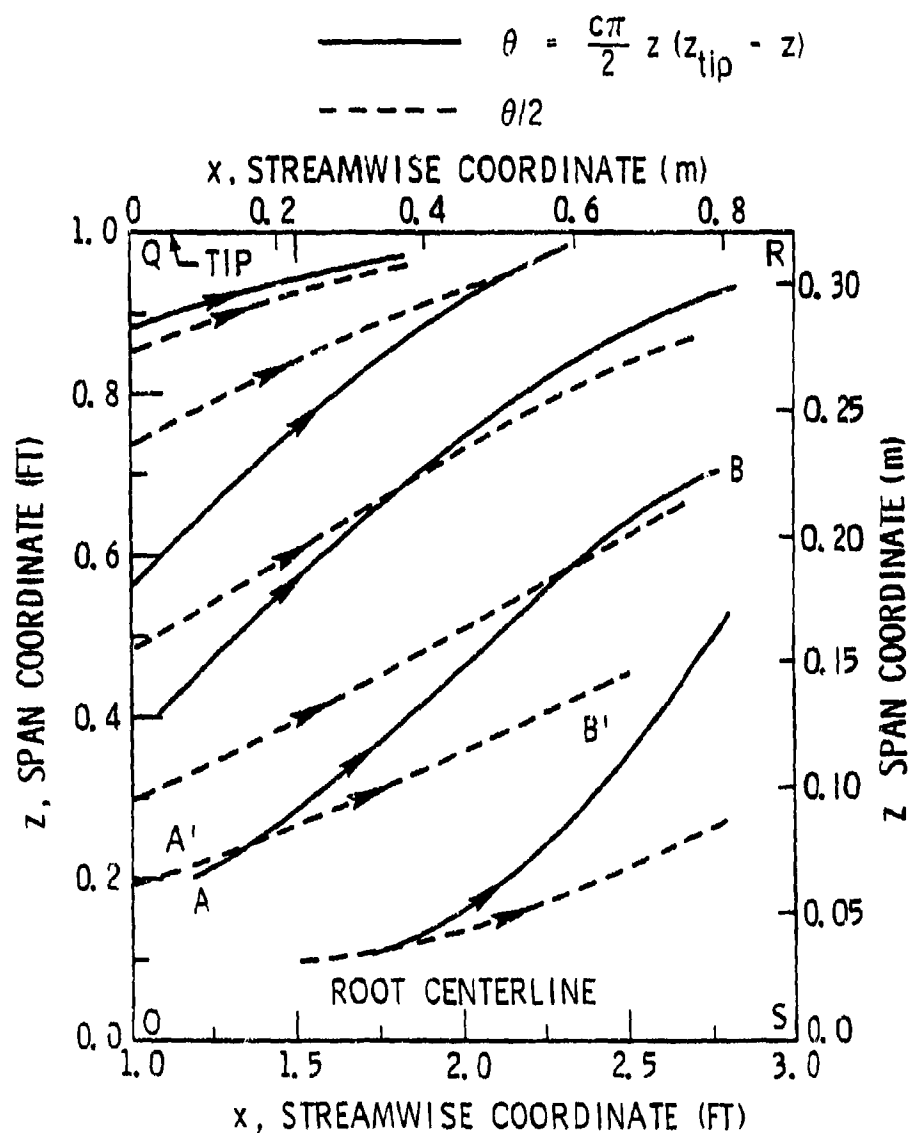


Figure 12. Cross Flow Effect on Jet "Shrink" and "End Wall Pullaway," $u_e = u_o x^{-1/2} \cos \theta$, $w_e = u_o x^{-1/2} \sin \theta$, $\frac{u_{\max}}{u_e} = 2 - |.5 - z|$.

rectangular wing. In this interpretation, OQ would be the leading edge, RS the trailing edge and QR its tip. Significant enhancement in downstream streamtube contraction is obvious with increase in cross flow. This contraction could presumably lead to end wall separation of the type observed in typical short ejectors. In Fig. 13, a similar picture is indicated for the increasing $w \sim x^{1/2}$ cross flow case. Although some qualitative similarity exists for the decaying case, it is evident that the possibility of a separation surface streamline envelope exists in the pattern of the streamlines.

6.0 Conclusions

A class of cases was investigated roughly possessing initial flow angularity and adverse pressure gradients prototypic of those on the blown surfaces of typical propulsive lift systems such as the Navy/Rockwell XFV-12A thrust augmented wing. Results obtained from the computational model indicate that if the initial total velocity is kept fixed, then the introduction of the cross flow enhances the freestream decay rate of the peak of the velocity component in the freestream direction. In addition, the entrainment quantity and its rate decrease with increased cross flow. The three-dimensional phenomena not only influence the effect of taper on the boundary layer control characteristics of a Coanda flap, but also indicate a "jet shrink" which could be a mechanism promoting end-wall separation. To our knowledge, our model is the first to quantify such trends. Both should be considered in the design of any propulsive lift system. Finally, the effect on the prescribed external adverse pressure gradient in the presence and absence of cross flow has also been examined. From the limited results, the spanwise separation line moves progressively further upstream with increasing cross flow.

SC79-3518

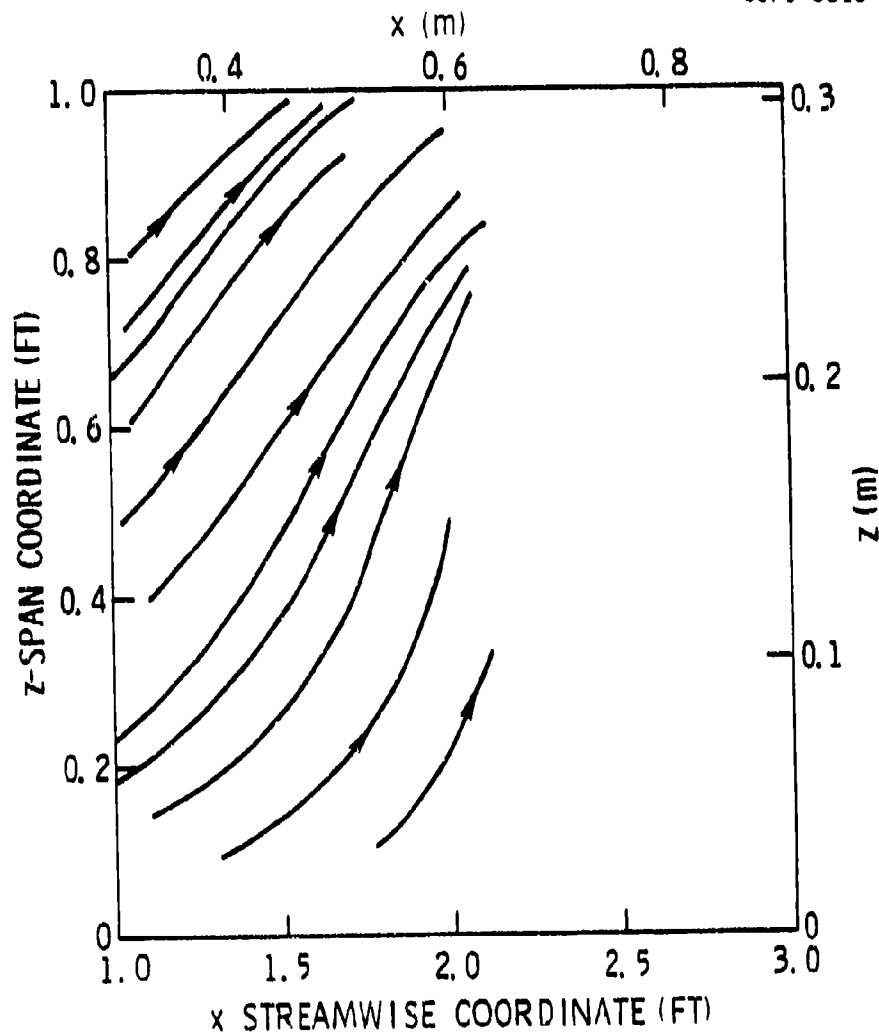


Figure 13. Cross Flow Effect on Jet "Shrink" with Streamwise Increase in Cross Flow, $u_e = u_0 x^{-1/2} \cos \theta$,
 $w_e = u_0 x^{1/2} \sin \theta$, $\frac{u_{\max}}{u_e} = 2 - |0.5 - z|$,
 $\theta = \frac{c\pi}{2} z(z_{\text{tip}} - z)$

7.0 References

1. Malmuth, N. and Szeto, R., "A Computational Model for Three-Dimensional Incompressible Small Cross Flow Wall Jets," NADC Final Report 76-105-30, December 15, 1977.
2. Reichardt, H., "Gesetzmassigkeiten der Freien Turbulenz," VDI-Forschungsheft, 414 (1942), 2nd ed. (1951).
3. Keller, H.B. and Cebeci, T., "Accurate Numerical Methods for Boundary Layers, I. Two-Dimensional Laminar Flows," Proceedings of the Second International Conference on Numerical Methods in Fluid Dynamics, Lecture Notes in Physics, Springer-Verlag, New York, Vol. 8, 1971.
4. Keller, H.B. and Cebeci, T., "Accurate Numerical Methods for Boundary Layers, II. Two-Dimensional Turbulent Flows," AIAA Journal, Vol. 10, September 1972, pp. 1197-1200.
5. Cebeci, T., "Calculation of Three Dimensional Boundary Layers, II. Three Dimensional Flows in Cartesian Coordinates," AIAA J., Vol. 13, No. 8, p. 1056, 1975.
6. Dvorak, F.A., "Calculation of Turbulent Boundary Layers and Wall Jets Over Curved Surfaces," AIAA J., Vol. 11, No. 4, pp. 517-524, 1973.
7. Ramaprian, B.R., "Turbulent Wall Jets in Conical Diffusers," AIAA J., Vol. 11, No. 12, pp. 1684-1690, September 1973.
8. Wilson, D.J., and Goldstein, R.J., "Turbulent Wall Jets with Cylindrical Streamwise Surface Curvature," Journal of Fluids Engineering, pp. 550-557, September 1976.
9. Murphy, W.D., Shankar, V., and Malmuth, N.D., "Three Dimensional Wall Jets," Quarterly Progress Report No. 2, NADC Contract No. N62269-77-C-0412.
10. Taylor, G.I., J. of Aero. Sci., Vol. 25, 7, pp. 464-465 (1958).

PART II. USERS MANUAL

Although the computer program given in the Appendix was designed to run on the Lawrence Berkeley Laboratory CDC 7600, it may be run on other CDC 6600 or CDC 7600 computers by either removing the plotting routines from the MAIN program or by adapting them to meet the requirements of any other facility.

1. Dack Setup (Berkeley CDC 7600 with plots)

```
Job Card
*NOTAPES
RUN76.
LGO,NL=60000.
DISPOSE,PLOT=PL,R=[SEND PLOTS TO W.D. MURPHY/SCIENCE
+CENTER/ROCKWELL INTERNATIONAL/
+P.O. BOX 1085/
+THOUSAND OAKS, CA. 91360], SC=BKY.
7/8/9
```

Source program

```
7/8/9
CASE 7 UE=1.0/SQRT(X) WE=1.0E-5*Z/SQRT(X) FLAT PLATE
$INPUTS XSUPPLY=.TRUE.,YSUPPLY=.TRUE.,ZSUPPLY=.TRUE.,OPTPT=.TRUE.$
↑
second column
6/7/8/9
```

Note that the DISPOSE card merely tells Berkeley where to send the plots and may be changed for other users. Also, [= 7/8 punch and] = 0/2/8 punch. For other computer facilities the RUN76 card may be replaced with FTN, and the *NOTAPES and DISPOSE cards should be removed.

2. Estimate of Running Time

The sample case (see the next section and the Appendix) required 389 seconds on the Berkeley CDC 7600. In general, the running time will depend upon the grid selected.

3. Type and Configuration of Computer Used in Program Development

- (i) Berkeley CDC 7600 (special feature = Calcomp plotter).
- (ii) Any other CDC 6600, CDC 7600, CDC 173, or CDC 176 may be used if the plotting routines are removed from the MAIN program.

4. Name and Level of Programming Language Used in Program

FORTRAN IV.

A. Input-Output Information

Glossary of Input Parameters

- XSUPPLY Logical variable which is .TRUE. if user supplies the streamwise mesh (x-mesh). Default value = .FALSE..
- YSUPPLY Logical variable which is .TRUE. if user supplies the η -mesh. Default value = .FALSE..
- ZSUPPLY Logical variable which is .TRUE. if user supplies the z-mesh. Default value = .FALSE..

XA Real variable. The starting streamwise station. Default value = 0.0. XA may be initialized in subroutine XMESH if XSUPPLY=.TRUE..

XB Real variable. The last streamwise station. Default value = 1. XB may be initialized in subroutine XMESH if XSUPPLY=.TRUE..

YA Real variable. The left end-point of the η -mesh. Default value = 0.0.

YB Real variable. The right end-point of the η -mesh. Default value = 20. YB may be initialized in subroutine YMESH if YSUPPLY=.TRUE..

ZA Real variable. The left end-point of the z-mesh. Default value = 0.0.

ZB Real variable. The right end-point of the z-mesh. Default value = 1. ZB may be initialized in subroutine ZMESH if ZSUPPLY=.TRUE..

HKS Real variable initializing the x-mesh in setup of the streamwise mesh and used only if XSUPPLY=.FALSE.. Default value = 1.E-5.

FACX Real variable representing the multiplication factor in the setup of the x-mesh if XSUPPLY=.FALSE., i.e., $X(N)=FACX*X(N-1)$ for the first few points. Default value = 1.2.

NX Integer variable. The last streamwise station. This may be supplied in subroutine XMESH if XSUPPLY=.TRUE.. Default value = 60.

J Integer variable. Total number of η -points which equals the number of internal intervals plus 1. J may be supplied by user in subroutine YMESH if YSUPPLY=.TRUE.. Default value = 101.

I Integer variable. Total number of z-points. I may be supplied by user in subroutine ZMESH if ZSUPPLY=.TRUE.. Default value = 11. The maximum value is also 11 unless the COMMON BLOCKS in the MAIN program are enlarged.

REFINE Logical variable which is .TRUE. if the η -mesh is to be refined. Default value = .TRUE..

HMAX Real variable. Maximum η -mesh interval permitted in refinement routine. Default value = 2.1.

OPTPT Logical variable which is true if user supplies streamwise stations to be printed on paper. These will also be plotted if the Berkeley 7600 plot routines are being employed. Default value = .FALSE.. User supplies these stations in subroutine PRMESH.

NPRINT Integer variable. Number of uniform streamwise stations at which solution is to be printed $((XB-XA)/NPRINT)$. Default value = 10.

USUPPLY Logical variable which is .TRUE. if user supplies the initial velocity profiles in subroutine PROFILE. Default value = .TRUE.. If these profiles are not supplied, similarity solutions will be generated and used as starting conditions.

R Real variable. Reynolds number. Default value = $1.0/14.216862E-5$.

Input

The first card in the input stream contains a TITLE card FORMAT(8A10) describing the problem to be solved. The remaining cards contain NAMELIST data of the variables in the above glossary. The list is defined by

NAMELIST /INPUTS/ XSUPPLY,YSUPPLY,ZSUPPLY,XA,....

The user must also supply the routines XMESH, YMESH, ZMESH, PRIMESH, and PROFILE if the variables XSUPPLY, YSUPPLY, ZSUPPLY, OPTPT, and USUPPLY have the value .TRUE., respectively. How these subroutines are written is fully described in the program listing in the Appendix.

In addition, the user must supply the routine PREPP which defines the values of $P_1, P_2, \dots, P_{10}$ (equation (14b)) in terms of functions of u_e, w_e, h_1, h_2, K_1 , and K_2 . In the example given in the Appendix, $u_e = 1/\sqrt{x}$, $w_e = 10^{-5} z/\sqrt{x}$, $h_1 = h_2 = 1$, and $K_1 = K_2 = 0$.

Plots

The program was designed to run on the Berkeley CDC 7600 and to use the Calcomp plotter subroutine package available at Berkeley. It may be run on other CDC computers by either removing the calls to CCNEXT, CCGRID, CCLTR, CCPLT and CCEND in the MAIN program or by adapting the plot routines to meet the requirements of the new facility.

Graphs are made of the following quantities:

1. y versus u/u_e at z_1, z_4, z_6, z_8 , and z_{11}
2. y versus w/w_e at z_1, z_4, z_6, z_8 , and z_{11}
3. Shear stress versus x
4. Stream lines
5. $u_{\max}$ versus x
6. Jet spreading versus x
7. Entrainment quantity versus x .

The labels for the individual plots are coded in LABEL1, LABEL2, ..., LABEL7 of the MAIN program and may be changed for each individual case study. For these plots the value of u_e must be coded for each new case in the MAIN program and in the subroutine NEWTON where w_e is also required. In the example in the Appendix $u_e = 1/\sqrt{x}$ and $w_e = 10^{-5} z/\sqrt{x}$.

Example (sample input deck)

```

CASE 7 UE=1.0/SQRT(X) WE=1.0E-5*Z/SQRT(X) FLAT PLATE
$INPUTS XSUPPLY=.TRUE.,YSUPPLY=.TRUE.,ZSUPPLY=.TRUE.,
OPTPT=.TRUE.$

```

The first card is the title card which is printed on paper, and the second card is a NAMELIST card which implies that the user will supply his own x-mesh, η -mesh, z-mesh, and the streamwise stations to be printed on paper. Examples of these subroutines can be found in the Appendix. All other variables will be assigned their default values.

The initial velocity profiles are coded in subroutine PROFILE and represent the initial conditions given on page 18 of Part I of this report. These initial profiles may easily be changed by following the instructions given in the subroutine.

Output

The title card is printed at the top of the page. If REFINES=.FALSE., the grid points (x- η -z) and y (assuming $u_0=1$) are printed. This is followed by the initial profiles where $F=f$, $DF=f'$, $DDF=f''$, $G=g$, $DG=g'$, and $DGG=g''$. The indices IZ and L represent the subscripts on the z- and η -mesh points, respectively. If REFINES=.TRUE., the program refines the initial mesh and prints the profiles on the new mesh before printing the refined grid. The NAMELIST is printed next.

Every time Newton's method converges for a given streamwise station and z-point the following data is recorded:

1. the number of iterations for convergence (ITER)
2. maximum absolute error (ERROR)
3. element of the solution vector where the maximum error occurs
4. y^* defined on page 11 of Part I (YTC)
5. $f''(0)$ (U(3,1))
6. $g''(0)$ (T)
7. $b_{10}, b_{20}, b_{30}, b_{40}, b_{50}, b_j$ where $b = 1 + \epsilon^+$.
8. shear stress (TAU).

After a complete z-plane is swept the following data is printed:

1. entrainment quantity (Q), page 24 of Part I
2. $u_{\max}$ at z_1, z_3, z_6, z_8 , and z_{10}
3. $y_{\max}$ at z_1, z_3, z_6, z_8 , and z_{10} .

Note that $u_{\max}$ and $y_{\max}$ are defined on page 27 of Part I.

The complete solution is printed at those streamwise stations previously specified by the user in PRMESH or at default values. The format for this printing is the same as that for the initial profiles. Plot data will also be loaded into storage vectors at these stations for possible later use by the plotting subroutines in the MAIN program.

Example (sample output listing)

Because of the three-dimensional nature of this problem, a complete listing of the output would be too exhaustive for this report. Therefore, we give below the first page of output that resulted from the sample input deck given earlier.

CASE 6 UE=COS(TH)/SURT(X) WE=SIN(TH)/SURT(X) FLAT PLATE

NEWTON ITER=4 ERROR=7.88503E-03 ATT 6, 3 YTC=1.96462E+00 U(3,1)=2.71916E+00 3.75751E+00
 B=1.52105E+00 6.83638E+00 8.83161E+00 1.28877E+00 1.89424E+00 1.83168E+00 1.83168E+00 1.83168E+00 1.83168E+00 1.83168E+00

INITIAL PROFILES

L	F	HF	DOF	6	DO6	DO6
1	1	0.	3.58832E+00	0.	3.58832E+00	3.58832E+00
2	1	7.8614E-02	3.43586E-01	1.78614E-02	3.43586E-01	3.82764E+00
3	2	7.7092E-02	4.19928E-01	2.7734E+00	4.19928E-01	2.39734E+00
4	4	2.0615E-02	5.07261E-01	2.74535E+00	5.07261E-02	2.74535E+00
5	1	6.27398E-02	6.05957E-01	2.56988E+00	6.27398E-02	2.56988E+00
6	1	9.22058E-02	7.15928E-01	2.36681E+00	9.22058E-02	2.36681E+00
7	1	1.31733E-01	8.36240E-01	2.13398E+00	1.31733E-01	2.13398E+00
8	1	1.91530E-01	9.64715E-01	1.87144E+00	1.91530E-01	1.87144E+00
9	2	7.0959E-01	1.89743E+00	2.70959E-01	1.89743E+00	1.57785E+00
10	3	3.3279E-01	1.16650E+00	1.41226E+00	1.16650E+00	1.41226E+00
11	3	3.78629E-01	1.22810E+00	1.25414E+00	1.22810E+00	1.25414E+00
12	4	4.84525E-01	1.29261E+00	1.67380E+00	1.29261E+00	1.67380E+00
13	5	5.21740E-01	1.34739E+00	9.03184E-01	1.34739E+00	9.03184E-01
14	6	6.13285E-01	1.40103E+00	7.10556E-01	1.40103E+00	7.10556E-01
15	7	7.07827E-01	1.44225E+00	5.38332E-01	1.44225E+00	5.38332E-01
16	7	7.95277E-01	1.46685E+00	3.94545E-01	1.46685E+00	3.94545E-01
17	8	8.63848E-01	1.48448E+00	2.65799E-01	1.48448E+00	2.65799E-01
18	9	9.43177E-01	1.49527E+00	9.43177E-01	1.49527E+00	1.43729E-01
19	10	1.01494E+00	1.49966E+00	3.92870E-02	1.49966E+00	3.92870E-02
20	1	1.88681E+00	1.49966E+00	-2.92815E-03	1.49966E+00	-2.92815E-03
21	1	1.5869E+00	1.49970E+00	-7.71600E-03	1.49970E+00	-7.71600E-03
22	1	1.23054E+00	1.49922E+00	-1.24932E-02	1.49922E+00	-1.24932E-02
23	1	1.34545E+00	1.49797E+00	-2.00983E-02	1.49797E+00	-2.00983E-02
24	1	1.46823E+00	1.49514E+00	-2.76327E-02	1.49514E+00	-2.76327E-02
25	1	1.57466E+00	1.49373E+00	-3.50702E-02	1.49373E+00	-3.50702E-02
26	1	1.7062E+00	1.48799E+00	-4.81319E-02	1.48799E+00	-4.81319E-02
27	1	1.98547E+00	1.48047E+00	-6.86533E-02	1.48047E+00	-6.86533E-02
28	2	2.29733E+00	1.46924E+00	-7.47787E-02	1.46924E+00	-7.47787E-02
29	2	2.47195E+00	1.45577E+00	-8.77280E-02	1.45577E+00	-8.77280E-02
30	3	3.04261E+00	1.41558E+00	-1.13030E-01	1.41558E+00	-1.13030E-01
31	3	3.37683E+00	1.38727E+00	-1.23862E-01	1.38727E+00	-1.23862E-01
32	3	3.70405E+00	1.35681E+00	-1.31083E-01	1.35681E+00	-1.31083E-01
33	4	4.36585E+00	1.28468E+00	-1.35333E-01	1.28468E+00	-1.35333E-01
34	4	4.99395E+00	1.22359E+00	-1.26868E-01	1.22359E+00	-1.26868E-01
35	5	6.16310E+00	1.11471E+00	-8.40266E-02	1.11471E+00	-8.40266E-02
36	6	7.1084E+00	1.07623E+00	-6.61194E-02	1.07623E+00	-6.61194E-02

NADC-77163-30

B. Subroutine Description

All subroutines contain many comment cards describing the routine and its function. Our purpose here is not to duplicate all this information but instead to discuss some of the technical details of each routine. An * denotes the routine is user supplied.

SUBROUTINE NAME: BC

PURPOSE: This routine computes the boundary conditions and the Jacobian of the boundary data. See equation (13) of Part I.

DESCRIPTION: The solution vector is written in the form $u = (f, f', f'', g, g', g'')^T$, and the variables A and B denote left and right boundary, respectively. The boundary conditions are assumed to be written in homogeneous form and the left boundary conditions are coded before the right ones. For example, the right boundary condition

$$f'(\eta_\infty) = 1$$

translates into code as

$$G(5) = UB(2) - 1.0.$$

SUBROUTINE NAME: BETASV

PURPOSE: This subroutine solves for β in the LU-decomposition of the block tridiagonal Jacobian matrix (equation (28a) in Part I).

DESCRIPTION: The block tridiagonal matrix is written as

$$/A \equiv [B_i, A_i, C_i] \quad (i=1, 2, \dots, J)$$

where A_i , B_i , and C_i are matrices of order 6.

$/A$ is decomposed as

$$/A \equiv [\beta_i \ I \ 0][0 \ \alpha_i \ C_i]$$

where

$$\alpha_i = A_i \quad (1a)$$

$$\beta_i \alpha_{i-1} = B_i \quad i=2, 3, \dots, J \quad (1b)$$

$$\alpha_i = A_i - \beta_i C_{i-1} \quad i=2, 3, \dots, J \quad (1c)$$

For simplicity, we write the LU decomposition of α_i as

$$\alpha_i = l_i u_i \quad (2)$$

If this decomposition doesn't exist, then permutation matrices for row and column pivoting can be introduced in equation (2). Assuming p rows in B_i , the solution of β_i in equation (1b) is given by

$$\left. \begin{aligned} u_{i-1}^T y_k &= \beta_i^T k \\ l_{i-1}^T \beta_k^T &= y_k \end{aligned} \right\} \quad k=1, 2, \dots, p \quad (3)$$

SUBROUTINE NAME: BLOCK1

PURPOSE: This subroutine decomposes the block tridiagonal Jacobian matrix into LU-form.

DESCRIPTION: The steps in equations (1-3) above are carried out by performing the indicated matrix multiplications and calling routines LUSOLV and BETASV.

SUBROUTINE NAME: BLOCK2

PURPOSE: This routine solves the block tridiagonal system $Ax = b$ assuming the factorized form.

DESCRIPTION: Write $z \equiv (z_1, z_2, \dots, z_J)^T$ and $x \equiv (x_1, x_2, \dots, x_J)^T$ where $z_\ell = (z_{1\ell}, z_{2\ell}, \dots, z_{6\ell})^T$ and $x_\ell = (x_{1\ell}, x_{2\ell}, \dots, x_{6\ell})^T$. Then BLOCK2 first solves

$$z_1 = b_1 \quad (4a)$$

$$z_k = b_k - \beta_k z_{k-1} \quad k=2,3,\dots,J \quad (4b)$$

and then

$$\alpha_J x_J = z_J \quad (5a)$$

$$\alpha_{\ell-1} x_{\ell-1} = z_{\ell-1} - C_{\ell-1} x_\ell \quad \ell=J,J-1,\dots,2 \quad (5b)$$

Equation (5a) is solved by calling USOLVE.

SUBROUTINE NAME: BOX

PURPOSE: This subroutine sets up the block tridiagonal system of equations for the attachment line equation (15-16) of Part I when KASE=2 and the 3-D wall jet equations (23-27) of Part I when KASE=3.

DESCRIPTION: The elements of $[B_1, A_1, C_1]$ are loaded by calling the routines BC, RHSF, and RHSF2D which contain Jacobian information for the boundary conditions and the various difference equations. Previous station information and updating of the turbulence model is included by incorporating the G matrix (see Appendix for the component description). Finally, the equations are rearranged to ensure the first diagonal block is nonsingular.

SUBROUTINE NAME: LUSOLV

PURPOSE: This subroutine decomposes a scalar matrix into LU-form using a mixed pivoting strategy.

DESCRIPTION: See Analysis of Numerical Methods by E. Isaacson and H.B. Keller for a standard discussion of Gaussian elimination (LU decomposition) with pivoting.

SUBROUTINE NAME: MAIN

PURPOSE: This is the main driving program.

DESCRIPTION: The MAIN program initializes solution vectors, sets default values, reads and writes NAMELIST data, and calls subroutines that initialize the grid and the velocity profiles. The marching scheme is called from MAIN as well as all plotting packages.

SUBROUTINE NAME: NETRUN

PURPOSE: This subroutine is used to add or delete η -mesh points in order to improve the smoothness of the solution especially in regions where large boundary layers exist.

DESCRIPTION: The local truncation error of the numerical scheme is approximated at the mid-point of each sub-interval of the initial grid. Points are added or deleted so that this error will remain constant on the whole interval. NETRUN is only called for the first x-point and z-point, if the user requests this option (REFINE=.TRUE.). This new improved grid is used for the entire 3-D region.

SUBROUTINE NAME: NEWTON

PURPOSE: This subroutine solves the 2-D attachment line equation when KASE=2 and the 3-D wall jet equation when KASE=3 by employing the underrelaxed Newton's method (equation (28) of Part I).

DESCRIPTION: The following notation is used:

N=N1 = number of unknowns in the solution vector = 6

NP=N2 = number of boundary conditions defined at the left boundary = 4

NQ=N3 = number of boundary conditions defined at the right boundary = 2.

Previous station information is loaded from subroutine PREPG and the eddy-viscosity from PREPB. After the proper block tridiagonal Jacobian matrix is loaded (KASE=2 or 3), it is solved using BLOCK1 and BLOCK2. Underrelaxation ($\omega=.7$) is performed on this solution, and a maximum absolute or relative (REL=.TRUE.) error between iterations is computed. If this error is greater than $\text{EPSERR}=10^{-2}$ (loaded from MAIN), the above steps are repeated; otherwise, plot and print vectors are loaded. u_e and w_e are coded into this subroutine and must be changed for each problem.

SUBROUTINE NAME: OUTPT

PURPOSE: This subroutine writes the solution on paper for the plane $X=XN$.

DESCRIPTION: The logic is set up so that the initial profile will only be printed once. The x-station is loaded into the plot vector for later use. The array ULAST contains f' and g' for future plots. The solution vector is printed on paper using the format described in the Output section.

SUBROUTINE NAME: PREP

PURPOSE: This routine sets up the initial grid and loads initial conditions into the solution vector. The grid is printed from this subroutine.

DESCRIPTION: Mesh points use default values if XSUPPLY, YSUPPLY, and ZSUPPLY are .FALSE.. For each .TRUE. value a user supplied subroutine is called. Values of x , η , ξ , and y are printed where y is computed from u_0 at some x station, usually x_p . The user should change u_0 with every new case study. If USUPPLY=.TRUE., initial conditions are loaded from the user supplied routine PROFILE; otherwise, a similarity solution is generated from initial exponential decaying functions given by statement functions defined by F1(T), F2(T), and F3(T). Grid refinement on either of these initial conditions will be performed if REFINES=.TRUE..

SUBROUTINE NAME: PREPB

PURPOSE: This subroutine models turbulence using the two layers discussed on pages 8 through 11 of Part I.

DESCRIPTION: The values of c_1 and c_2 (TC1 and TC2, respectively) have been set to zero using a DATA STATEMENT but they may be assigned values between zero and three. The program is divided into two parts depending

upon whether the attachment line equations ($IZ=1$) or the 3-D wall jet equations ($IZ\neq 1$) are being solved. In each part the value of y^* (YTC) is determined, and the quantities $b = 1+\epsilon^+$ in equations (11) and (12) of Part I are computed and stored in the second and sixth rows of the G matrix, respectively. Previous station data is also updated and stored in the third and fourth rows.

SUBROUTINE NAME: PREPG

PURPOSE: This subroutine computes previous station data for the full 3-D wall jet equations.

DESCRIPTION: The actual variables coded in this routine are T (G(3,L)) and S (G(4,L)) which were first derived in Ref. 9 of Part I. The following notation is used for the FORTRAN variables:

$$UHX = u_{j-1/2}^{n,i-1}$$

$$UHDX = u_{j-1/2}^{n-1,i}$$

$$UHDOX = u_{j-1/2}^{n-1,i-1}$$

$$UT = \tilde{u}_{j-1/2}$$

SUBROUTINE NAME: PREPG2

PURPOSE: This subroutine computes previous station data for the attachment line equation (equations (15-16) of Part I).

DESCRIPTION: The quantities $T_{j-1/2}^{n-1}$ (G(3,L)) and $S_{j-1/2}^{n-1}$ (G(4,L)) of Ref. 9 are coded in this subroutine. UN denotes the variable $u_{j-1/2}^{n-1}$.

SUBROUTINE NAME: PREPP*

PURPOSE: The variables $P_1, P_2, \dots, P_{10}$ given by equation (14) of Part I are coded in this subroutine.

DESCRIPTION: For each new problem the following quantities must be coded into this subroutine:

$$UE = u_e$$

$$UEX = \partial u_e / \partial x$$

$$UEZ = \partial u_e / \partial z$$

$$WE = w_e$$

$$WEX = \partial w_e / \partial x$$

$$WEZ = \partial w_e / \partial z$$

$$WEZX = \partial^2 w_e / \partial z \partial x$$

The values of $P_1, P_2, \dots, P_{10}$ in this routine are written for the special case of a flat plate; however, they may easily be changed to exactly those

given by equation (14). Note that when IZ=1, the routine computes the corresponding values of P for the attachment line equations.

SUBROUTINE NAME: PRMESH*

PURPOSE: This routine allows the user to supply the streamwise stations at which solutions are to be printed on paper. These stations will also serve as those where plot vectors will be loaded.

DESCRIPTION: See program listing in the Appendix.

SUBROUTINE NAME: PROFILE*

PURPOSE: This routine sets up the initial velocity profiles.

DESCRIPTION: In the sample case given in the Appendix we have coded the initial conditions described on page 18 of Part I. The quantity u_{max}/u_e varies from 1.5 to 2.0. The first six rows of the USTORE matrix contain the values of (f,f',f'',g,g',g'') as a function of η_g and z_m . If REFIN=TRUE., the mesh refinement routine alters the η -mesh in order to make the initial conditions smoother between grid points. New initial conditions are computed and printed at the refined points.

SUBROUTINE NAME: RHSF

PURPOSE: This routine computes the quantities on the right-hand side of the equations given by (31-36) in Ref. 9. The Jacobian matrix is also evaluated and stored in the A matrix.

DESCRIPTION: The following notation is used:

UT = $\bar{u}/4.0$

UBAR = $\bar{u}$

U = current value of $u_{j-1/2}^{n,i}$

UX = pass value $u_{j-1/2}^{n,i-1}$

UXX = pass value $u_{j-1/2}^{n-1,i}$

UXXX = pass value $u_{j-1/2}^{n-1,i-1}$

The values of the right-hand side of equations (31-36) are stored in F(1), F(2), ..., F(6), respectively

SUBROUTINE NAME: RHSF2D

PURPOSE: RHSF2D is similar to RHSF except the attachment line equations given by (38) in Ref. 9 and the corresponding Jacobian matrix are coded.

DESCRIPTION: See RHSF above.

SUBROUTINE NAME: TRUN

PURPOSE: This subroutine computes the local truncation error of the centered-Euler scheme.

DESCRIPTION: See H.B. Keller, "Accurate Difference Methods for Nonlinear Two-point Boundary Value Problems," SIAM J. Numer. Anal., 11 (1974), pp. 305-320.

SUBROUTINE NAME: USOLVE

PURPOSE: This routine solves the scalar matrix equation $Ax = f$ after A has been put in LU-form.

DESCRIPTION: See E. Isaacson and H.B. Keller, Analysis of Numerical Methods, J. Wiley & Sons, New York, 1966.

SUBROUTINE NAME: VANDET

PURPOSE: This function computes the determinant of an $n \times n$ Vandermonde matrix for $1 < n < 7$. This routine is called by TRUN in order to compute the local truncation error of the centered-Euler or box scheme.

DESCRIPTION: The Vandermonde determinant is given by

$$V = \begin{vmatrix} 1 & x_0 & \dots & x_0^n \\ 1 & x_1 & \dots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^n \end{vmatrix} = \prod_{i>j} (x_i - x_j).$$

SUBROUTINE NAME: VELMAX

PURPOSE: This subroutine computes $u_{\max}$ and $y_{\max}$ described on page 18 of Part I.

DESCRIPTION: $u(\eta_j)$ is computed for each mesh point η_j and the maximum value is determined, say $u_{\max} = u(\eta^*)$. Then $y_{\max}$ is given by $y_{\max} = \eta^* \sqrt{vx/u_a}$.

SUBROUTINE NAME: WALJET

PURPOSE: This is the main marching routine which calls NEWTON to solve either the 3-D wall jet problem (equations (11-14) of Part I), or the 2-D attachment line equations (equations (15-16) of Part I).

DESCRIPTION: The marching direction is discussed on page 16 of Part I. Unnatural oscillations are damped out by the averaging procedure discussed on pages 17 and 19 of Part I. The OUTPT routine is called at the appropriate point from this subroutine depending upon whether OPTPT is .TRUE. or .FALSE..

If separation ($f'(0) < 0$) occurs, the marching is terminated and the solution at the previous streamwise station is printed.

SUBROUTINE NAME: XMesh*

PURPOSE: This routine allows the user to input his own x-mesh.

DESCRIPTION: See the Appendix for an example and a complete explanation.

SUBROUTINE NAME: YMesh*

PURPOSE: This routine allows the user to input his own y-mesh.

DESCRIPTION: See the Appendix for an example and a complete explanation.

SUBROUTINE NAME: ZMesh*

PURPOSE: This routine allows the user to input his own z-mesh.

DESCRIPTION: See the Appendix for an example and a complete explanation.

NADC-77163-30

APPENDIX
PROGRAM LISTING

A-1

```

PROGRAM MAIN(INPUT,OUTPUT,TAPES=INPUT,TAPE6=OUTPUT,PLOT,
X
TAPF9=101,TAPF99=PL01)
00000010
00000012
00000020
00000030
00000040
00000050
00000060
00000070
00000080
00000090
00000100
00000110
00000120
00000130
00000140
00000150
00000160
00000170
00000180
00000190
00000200
00000210
00000220
00000230
00000240
00000250
00000260
00000270
00000280
00000290
00000300
00000310
00000320
00000330
00000340
00000350
00000360
00000370
00000380
00000390
00000400
00000410
00000420
00000430
00000440
00000450

C THIS IS THE MAIN PROGRAM FOR SOLVING THE THREE-DIMENSIONAL INCOMPRESSIBLE WALL JET EQUATION WITH LARGE CROSS FLOW. H. B. KELLER'S BOX SCHEME (REFERENCE 1) IS USED TO DISCRETIZE THE NONLINEAR SYSTEM OF PARTIAL DIFFERENTIAL EQUATIONS. THE DISCRETIZED SYSTEM IS THEN SOLVED BY NEWTON'S METHOD.
C THE FIRST CARD IN THE INPUT STREAM IS THE TITLE CARD (FORMAT(BA10)).
C THE REMAINING DATA IS ENTERED BY NAMELIST ** INPUTS **. THOSE VARIABLES THAT ARE INPUT PARAMETERS IN THIS NAMELIST ARE ENCLOSED C BY ** .... ** AT THE END OF THE DESCRIPTION BELOW.
C THE PLOTTING ROUTINE'S GIVE GRAPHS OF U/V/E VS Y, W/E VS Y, SHEAR C STRESS VS X, STREAM LINES, UMAX VS X, JET SPREADING VS X, AND THE C ENTRAINMENT QUANTITY VS X.
C DESCRIPTION OF MAIN VARIABLES
C A MAIN DIAGONAL BLOCKS OF THE BLOCK-TRI-DIAGONAL MATRIX
C (U A C) OBTAINED FROM LINEARIZATION OF THE FINITE-DIFFERENCE APPROXIMATION OF THE GOVERNING EQUATIONS
C AJA JACOBIAN MATRIX USED IN SETUP OF THE BLOCK TRI-DIAGONAL MATRIX
C H LOWER DIAGONAL BLOCKS. SEE DESCRIPTION OF VARIABLE A
C C UPPER DIAGONAL BLOCKS. SEE DESCRIPTION OF VARIABLE A
C CCEFACT REAL VARIABLES IN THIS COMMON BLOCK ARE USED FOR PLOTTING ON THE GENESEE CDC 7600
C CCEPOML MONE PLOTTING VARIABLES IN THIS COMMON BLOCK
C C DIH REAL VECTOR VARIABLE. THE DIFFERENCE BETWEEN TWO NEWTON ITERATES
C EPSLNN REAL CONSTANT VARIABLE. CONVERGENCE CRITERION OF NEWTON'S METHOD
C EIA REAL VECTOR, E12 MESH POINTS
C F REAL VECTOR VARIABLE. RIGHT HAND SIDE OF THE BLOCK TRI-DIAGONAL SYSTEM OF EQUATIONS
C FACX REAL CONSTANT VARIABLE. MULTIPLICATION FACTOR IN SETUP OF STREAMLINE MESH. **INPUT, DEFAULT VALUE=1.2**
C FF REAL VECTOR VARIABLE. USED IN SETUP OF BLOCK TRI-DIAGONAL SYSTEM OF EQUATIONS
C G REAL VECTOR VARIABLE. USED IN SETUP OF BLOCK TRI-DIAGONAL SYSTEM OF EQUATIONS (THE THIRD AND FOURTH COMPONENTS CONTAIN CONTRIBUTIONS FROM A PREVIOUS X OR Z STATION, AND THE REMAINING COMPONENTS ARE USED IN THE TWO LAYER TURBULENCE MIXING LENGTH MODEL).
C HKS REAL CONSTANT VARIABLE. INITIAL MESH-SIZE IN SETUP OF STREAMLINE MESH. **INPUT, DEFAULT VALUE=1.E-5**
C HMX REAL VECTOR VARIABLE. STREAMLINE MESH

```



```

C      REAL CONSTANT VARIABLE, REYNOLDS NUMBER, **INPUT, DEFAULT
C      VAL(4)=1.0/14.21662E-5 **
C      LOGICAL VARIABLE,=.TRUE., IF VERTICAL MESH IS TO BE DEFINED,0000990
C      **INPUT, DEFAULT VALUE=.TRUE.,**
C      REAL CONSTANT VARIABLE, KINEMATIC VISCOSITY
C      LOGICAL VARIABLE, TAKES ON THE VALUE .TRUE., IF SEPARATION
C      OCCURS
C      REAL VECTOR, SHEAR STRESS AT BODY
C      TITLE
C      REAL VECTOR, CONTAINS TITLE OF CASE BEING SOLVED, **INPUT,
C      NO DEFAULT VALUE**
C      REAL VECTOR, STREAM LINE SLOPES
C      REAL VECTOR VARIABLE, USED IN SETUP OF BLOCK TRIANGULAR
C      SYSTEM
C      REAL VECTOR VARIABLE, USED IN SETUP OF BLOCK TRIANGULAR
C      SYSTEM
C      REAL VECTOR, MAXIMUM VALUE OF U
C      REAL STORAGE MATRIX CONTAINING THE SOLUTION
C      LOGICAL VARIABLE,=.TRUE., IF USER SUPPLIES HIS OWN INITIAL
C      CONDITIONS (VELOCITY PROFILES). IF .FALSE., SIMILARITY
C      SOLUTIONS WILL BE GENERATED AND USED AS STARTING CONDITIONS
C      **INPUT, DEFAULT VALUE=.TRUE., **
C      REAL VECTOR VARIABLE, STREAMWISE VELOCITY COMPONENT AT THE
C      PRESENT STREAMWISE STATION, U(1,1) HAS THE VALUE OF KTH
C      COMPONENT OF U AT ETA = ETAL(1), WHERE ETAL(1)=YA=0 AND
C      ETAL(1)=ETAM-1/(H(1)-1), M=2,3,...,J-1. THUS U(1,3) HAS
C      VALUE OF F AT ETAL(3)
C      REAL VECTOR VARIABLE, STREAMWISE VELOCITY COMPONENT AT THE
C      PREVIOUS STATION
C      REAL CONSTANT VARIABLE, STARTING STREAMWISE STATION,
C      **INPUT, DEFAULT VALUE=0.0 **
C      REAL CONSTANT VARIABLE, THE RIGHT END-POINT OF STREAMWISE
C      MESH, I.I.E. THE LAST STREAMWISE STATION, **INPUT, DEFAULT
C      VALUE=1.0**
C      REAL VECTOR, STREAMWISE STATIONS FOR FIRST TWO PLOTS
C      REAL CONSTANT VARIABLE, UNIFORM INTERVAL TO WHICH SOLUTION
C      IS TO BE PRINTED =(XJ-XJ)/NPRINT
C      LOGICAL VARIABLE,=.TRUE., IF USER SUPPLIES THE STREAMWISE
C      MESH, **INPUT, DEFAULT VALUE=.FALSE.,**
C      REAL VECTOR, STREAMWISE STATIONS USED IN PLOTTING
C      REAL CONSTANT VARIABLE, THE LEFT END-POINT OF VERTICAL
C      MESH, **INPUT, DEFAULT VALUE=0.0 **
C      REAL CONSTANT VARIABLE, THE RIGHT END-POINT OF VERTICAL
C      MESH, **INPUT, DEFAULT VALUE=20.0**
C      REAL VECTOR, VALUE OF Y WHERE Y ATTAINS ITS MAXIMUM
C      LOGICAL VARIABLE,=.TRUE., IF USER SUPPLIES THE VERTICAL
C      MESH, **INPUT, DEFAULT VALUE=.FALSE.,**
C      REAL CONSTANT VARIABLE, THE LEFT END-POINT OF Z-MESH,
C      **INPUT, DEFAULT VALUE=0.0 **
C      REAL CONSTANT VARIABLE, THE RIGHT END-POINT OF Z-MESH,
C      **INPUT, DEFAULT VALUE=1.0 **
C      LOGICAL VARIABLE,=.TRUE., IF INP=1000 THEN MESH AND Z-MESH

```



```

      NH=1.
      YH=20.
      ZH=1.0
      J=101
      I=11
      NX=00
      HMAX=2.1
      APT=XH/MPH*NI
      KC=1
      N=1.0/15.216062E-5
      RMU=14.216062E-5

      C
      C INITIALIZE
      C
      JMAX=101
      MAXITS=50
      EPSERR=1.0E-2
      REL=.FALSE.
      CUTOFF=1.0E-3
      MTEST=5
      KSTART=1
      KLEN=1
      DO 5 I=1,N5
      5 QINIT(I)=0.0
      DO 30 L=1,JMAX
      DO 20 K=1,6
      6 K(L)=1.
      UTR(K,L)=0.
      20 CONTINUE
      30 CONTINUE
      READ(5,35) TITLE
      35 FORMAT(A10)
      WRITE(5,36) TITLE
      36 FORMAT(1H1,10X,BALD,/)
      READ(5,*) INPUTS
      NSORT=SUM(IH)
      CALL PREP(KSTART)
      WRITE(6,*) INPUTS
      CCAMIN=100.
      CCYMIN=100.
      M1=10
      CCAMAX=600.
      CCYMAX=600.
      CCZ=.5*(CCAMAX-CCYMIN)-50.
      CCY=650.
      MYI=10
      FACLOM=1.0
      XMIN=0.0
      YMIN=0.0
      XMAX=2.0
      YMAX=5.

```

C C

A 7

```

0003000
0003010
0003020
0003030
0003040
0003050
0003060
0003070
0003080
0003090
0003100
0003110
0003120
0003130
0003140
0003150
0003160
0003170
0003180
0003190
0003200
0003210
0003220
0003230
0003240
0003250
0003260
0003270
0003280
0003290
0003300
0003310
0003320
0003330
0003340
0003350
0003360
0003370
0003380
0003390
0003400
0003410
0003420
0003430
0003440
0003450
0003460
0003470
0003480
0003490
0003500

CALL WALFTR(ISTART)
CALL PHOFL

C
C REMOVE ALL OF THE NEXT CARDS EXCEPT THE LAST TWO IF THIS PROGRAM IS
C NOT BEING RUN ON THE HENKLEY CDC 7600 ON IF PLOTS ARE NOT DESIRED.
C
      JJ=J
      PHO02=1.6
      IF (I.EQ.1) GO TO 1010
      IF (SEPMAT) GO TO 210
      DO 200 K=1,5
      IT=ITSTUN(K)
      CALL CCNEXT
      CALL CCGRID(MA),2*H(LABELS,MY1,2)
      CALL CCLTRICCA.CCY,0,2,LABEL1,20)
      DO 100 IP=1,3
      PHO01=PHO02

C ***** THE NEXT OF CARD MUST BE CHANGED FOR EACH CASE STUDY *****
C *
      UE=1.0/SQRT(PI*IP))
C *
      PHO02=SQRT(PI*IP)*KHI/UE)
      LI=1
      DO 95 L=1,JJ
      ETAIL)=PHO02*ETAIL)/PHO01
      IF (ETAIL).GT.YMAX.AND.ETA(L).LE.YMAX) J=L
      LI=L
      HNEW(L)=URAST(IP,L,II)
      IF (HNEW(L).GT.XMAX) HNEW(L)=XMAX
      95 CONTINUE
      100 CALL CCLPLOT(HNEW,ETA,J,AMJ0IN)
      CALL CCNEXT
      CALL CCGRID(MA),2*H(LABELS,MY1,2)
      CALL CCLTRICCA.CCY,0,2,LABEL2,20)
      DO 120 IP=4,6
      PHO01=PHO02

C ***** THE NEXT OF CARD MUST BE CHANGED FOR EACH CASE STUDY *****
C *
      UE=1.0/SQRT(PI*IP-3))
C *
      PHO02=SQRT(PI*IP-3)*KHI/UE)
      LI=1
      DO 110 L=1,JJ
      ETAIL)=PHO02*ETAIL)/PHO01
      IF (ETAIL).GT.YMAX.AND.ETA(L).LE.YMAX) J=L
      HNEW(L)=URAST(IP,L,II)
      IF (HNEW(L).GT.XMAX) HNEW(L)=XMAX
      LI=L
      110 CONTINUE

```

```

120 CALL CCLPLOT(X1AU(2),Y1AU(2),MX,4HJUN)
200 CONTINUE
210 CONTINUE
  NX=NX+1
  AMIN=XA
  AMAX=XH
  YMAX=Y+OE 3
  CALL CCNFI
  CALL CCGRID(X1,2,4H,AMELS,MY1,2)
  CALL CCLIR(CCA,CY,0,2,LABEL3,20)
  DO 250 K=1,5
    CALL CCLPLOT(X1AU(2),Y1AU(2),K),MX,4HJUN)
    YMAX=Y+2,0
  CALL CCNFI
  CALL CCGRID(X1,2,4H,AMELS,MY1,2)
  CALL CCLIR(CCA,CY,0,2,LABEL5,20)
  DO 260 K=1,5
    CALL CCLPLOT(X1AU(2),Y1AU(2),K),MX,4HJUN)
    YMAX=Y+2
  CALL CCNFI
  CALL CCGRID(X1,2,4H,AMELS,MY1,2)
  CALL CCLIR(CCA,CY,0,2,LABEL6,20)
  DO 270 K=1,5
    CALL CCLPLOT(X1AU(2),Y1AU(2),K),MX,4HJUN)
    YMAX=Y+0,0
  DO 280 L=2,MX
    IF(YMAX-64,0)INT(L) GO TO 288
    YMAX=Y+INT(L)
    280 CONTINUE
    YMAX=YMAX+.001
    CALL CCNFI
    CALL CCGRID(X1,2,4H,AMELS,MY1,2)
    CALL CCLIR(CCA,CY,0,2,LABEL7,20)
    CALL CCLPLOT(X1AU(2),Y1AU(2),MX,4HJUN)
    L=L+1
    IF(NX-LY,42) GO TO 505
    DO 500 LL=62,MX
      L=L+1
    500 IF(L)=1
    505 CONTINUE
    AMIN=XA
    YMAX=Y+H
    CCAMAX=500.
    MX1=4
    MY1=5
    CCA=5+(CCAMAX-CCAMIN)/50.
    CALL CCNFI
    CALL CCGRID(X1,2,4H,AMELS,MY1,2)
    CALL CCLIR(CCA,CY,0,2,LABEL4,20)
    I1=I-1
    L=NH=1

```

```

00044824
00044830
00044840
00044850
00044860
00044870
00044880
00044890
00044900
00044910
00044920
00044930
00044940
00044950
00044960
00044970
00044980
00044990
00045000
00045010
00045020
00045030
00045040
00045050
00045060
00045070
00045080
00045090
00045100
00045110
00045120
00045130
00045140
00045150
00045160
00045170
00045180
00045190
00045200
00045210
00045220
00045230
00045240
00045250
00045260
00045270
00045280
00045290
00045300
00045310
00045320
00045330
00045340
00045350
00045360
00045370
00045380
00045390
00045400
00045410
00045420
00045430
00045440
00045450
00045460
00045470
00045480
00045490
00045500
00045510
00045520

A=XA
DO 550 L=2,NX
  A=X*IKX(L)
  IF (L.NE.1) (P(LSUM)) GO TO 600
  HMF(L)=X
  LSUM=LSUM+1
  Z=ZA
  DO 550 I=1,II
    ETA(I)=Z
    IN=THE TAIL,I
    CTH=COS(TH)
    SH=SIN(TH)
    HNEW(I)=Z*SCIN+Z
    ETA(I)=Z*Z*SCIN
    IF (HNEW(I).GT.XMAX.OR.FIA(1).GT.ZH) GO TO 550
    IF (HNEW(I).GT.XMAX.OR.ETA(2).GT.ZH) GO TO 550
  540 CALL CCPLUT(HMF,FTA,Z*AHJOIN)
  550 Z=Z+HR/(1)
  600 CONTINUE
  1000 CALL CCRM
  1010 STOP
  END
SUBROUTINE NEWTON(KASE,NI,N2,N3)
C
C THIS SUBROUTINE SOLVES THE 2-D ATTACHMENT LINE EQUATION WHEN KASE=2
C AND THE 3-D WALL JET EQUATION WHEN KASE=3. NEWTON ITERATIONS ARE
C EMPLOYED.
C
LOGICAL REL
COMMON /TA/ THE TA(65,1)
COMMON /G/ G(6,1)
COMMON /PARNI/ N,NP,NQ
COMMON /PARMI/ I,J,LZ
COMMON /PARMI/ KIMH,XIN,XK,XZ,NH,Z
COMMON /UT/ UT(6,1) /UI/ UI(6,1)
COMMON /TAIRU/ TAIRU(100,5) /XTAU/ XTAU(100)
COMMON /GINT/ GINT(65)
COMMON /MSHY/ MSHY(1) /MSHZ/ MSHZ(1)
COMMON /HRI/ HRI
COMMON /IMAX/ IMAX(65,5)
COMMON /YHIG/ YHIG(65,5)
COMMON /PHFPH/ PHFPH,YTC,YTOL,D2,IER,YTCC
EXTERNAL MHSF,MHCF,MHSF20
DATA PI/3.14159265358979/
C INITIALIZE
C
N=NI
NP=N2

```

```

0004533      NU=N3
0004540      IFLAG=1
0004550      YTC=0.
0004560      YTDJ=0.
0004570
0004580
0004590
0004600
0004610
0004620
0004630
0004640
0004650
0004660
0004670
0004680
0004690
0004700
0004710
0004720
0004730
0004740
0004750
0004760
0004770
0004780
0004790
0004800
0004810
0004820
0004830
0004840
0004850
0004860
0004870
0004880
0004890
0004900
0004910
0004920
0004930
0004940
0004950
0004960
0004970
0004980
0004990
0005000
0005010
0005020
0005030

C CONTINUATION FROM PREVIOUS STATIONS
C
C      IF (KASE.EQ.3) GO TO 10
C      CALL PHFPG2
C      IF (KCON1.EQ.1) GO TO 15
C      CALL PHFPP(KN1/3)
C      GO TO 15
C      10 CALL PHFPG
C      15 CONTINUE
C
C NEWTON ITERATION
C
C      DO 300 M=1,MAXIT5
C      ITER=M
C
C      SOLVE 2-D PROBLEM
C
C
C      IF (KCON1.EQ.1) CALL PHFPP
C      IF (KASE.EQ.2) CALL BOX(KASE,RHSF2D,BC)
C
C      SOLVE 3-D PROBLEM
C
C      IF (KASE.EQ.3) CALL BOX(KASE,RHSF,RC)
C
C CALL BLOCK TRIANGULAR MATRIX SOLVER
C
C      CALL HLUCK1
C      CALL HLUCK2
C
C
C ERROR TESTING BETWEEN TWO NEWTON ITERATES
C
C      ENMAX=0.
C      DO 200 L=1,J
C
C      DO 200 K=1,M
C      U(L,K)=UT(K,L)+PHF6(K,NU,K,L)
C      ENMAX=ABS(DU(K,L))
C
C USEX=RELAXATION
C
C      DO 200 K=1,M
C      U(L,K)=UT(K,L)+PHF6(K,NU,K,L)
C      ENMAX=ABS(DU(K,L))
C
C ABSOLUTE ERROR CRITERION IS USED IF VARIABLE ** REL ** IS SET TO
C .FALSE.
C
C      IF (.NOT.MF1) GO TO 195
C      IF (ABS(ENMAX)-1.E-11) GO TO 195
C      ENMAX=ENMAX/ABS(UT(K,L))+0.5*DU(K,L))

```

```

195 CONTINUE
  IF (EMAX.GE.ERROR) GO TO 200
  LOCK=1
  LUCT=1
  EMAX=EMIN
200 CONTINUE
  C
  IF (EMAX.GE.ERROR) GO TO 205
  WRITE(6,6400) M,EMAX,LOCK,LUCT,YICC,UT(3,1),UT(6,1)
  C
  C COMPUTE PLOT DATA
  C
  C *** THE MEAT (E AND) OF CARDS MUST BE CHANGED FOR EACH CASE STUDY ***
  C
  UK=1.0/SUM(1:N)
  ME=1.0E-5*UE*ZM
  C
  C *****
  IEPR=SUM(1:UE*MP*EN)
  JJ=J-1
  SUM=0.0
  DO 700 I=1,J)
    SUM=SUM+SUM*Y(I)*(UI(2,I)*UT(2,I,1))
    SUM=IEPR*SUM
  700 IF (I.EQ.1) GO TO 710
    QIN(KOUNT)=QIN(KOUNT)+5*ME*7(I,1)*(SUM+SUMI)
  710 SUMI=SUM
    ARB=ME*UT(6,1)/(UE*UT(3,1))
    IEPR(KOUNT,1,2)=ATAN(ARB)
    PH=PI*IEPR/KOUNT*AN
    Q=SUM(1:PH)*((UE*UT(3,1))*2*(UE*UT(6,1))*2)
    IF (I,ME,1) GO TO 201
    TAU(KOUNT,1)=Q
    TAU(KOUNT)=K
    CALL VFLMAX(UMAX(KOUNT,1),YHIG(KOUNT,1),IE)
    GO TO 750
  201 IF (I,ME,1) GO TO 202
    TAU(KOUNT,2)=Q
    CALL VFLMAX(UMAX(KOUNT,2),YHIG(KOUNT,2),UE)
    GO TO 750
  202 IF (I,ME,1) GO TO 203
    TAU(KOUNT,3)=Q
    CALL VFLMAX(UMAX(KOUNT,3),YHIG(KOUNT,3),UE)
    GO TO 750
  203 IF (I,ME,1) GO TO 204
    TAU(KOUNT,4)=Q
    CALL VFLMAX(UMAX(KOUNT,4),YHIG(KOUNT,4),UE)
    GO TO 750
  204 IF (I,ME,1) GO TO 750
    TAU(KOUNT,5)=Q
    CALL VFLMAX(UMAX(KOUNT,5),YHIG(KOUNT,5),UE)

```



```

C * .....
C .....
P2=X*UEA/UE
P1=-5*1.0+P2
P3=X*WZ/UE
P5=W*E*X*UE/Z/(UE*UE)
P6=P3-.5*P5
P7=W*E/UE
P8=G.0
P9=0.0
P10=XX
IF (I7.FU.1) GO TO 10
P4=X*WEA/WF
WEFUNH
C
C ATTACHMENT LINE PARAMETERS
C
10 P4=X*WEZ/X*WEZ
WEFUNH
END
SUBROUTINE VELMAX(HIG,YHIG,UE)
C
C THIS ROUTINE COMPUTES THE MAXIMUM VALUE OF U AND THE CORRESPONDING
C VALUE OF Y.
C
COMMON /PARM4/ I,J,I2
COMMON /PARM6/ RINMT,RS,HA,X,7N,HZ,L
COMMON /UT/ U(16,1)
COMMON /ETA/ ETAT(1)
COMMON /RMI/ RMI
RIG=0.0
DO 20 L=1,J
TEST=0*UT(2,L)
IF (HIG.GE.TEST) (4) 10 20
RIG=TEST
HIG=L
20 CONTINUE
IF (RIG.GT.2.0) HIG=2.0
YHIG=FAI(HIG)*SINT(RMI*AN/UE)
RETURN
END
SUBROUTINE PREP(NSTART)
C
C THIS ROUTINE SETS UP THE PAGES ON CALLS FROM SUBROUTINES. EITHER
C INITIAL PROFILES ARE COMPUTED (USUPLY=.FALSE.) FROM SIMILARITY
C SOLUTIONS OR THEY ARE SUPPLIED BY THE USER.
C
LOGICAL RELX,SUPLY,Y,SUPLY,Z,SUPLY,NEFINE,OPTPT,USUPLY
COMMON /OPTION/ X,SUPLY,Z,SUPLY,Z,SUPLY,NEFINE,USUPLY
COMMON /UNIT/ API,OPTPT,IMPRINT
COMMON /MCSYS/ M(X,1) /M(SYS/ M(X))

```

```

COMMON /PARM3/ MAXITS,FPSEMR,KIER,REL,CUIOFF,MTEST
COMMON /PARM4/ I,J,LZ
COMMON /PARM5/ FACX,HKS,AA,XH,YA,YB,ZA,ZH,NX
COMMON /PARM6/ KINH,XN,XA,XA/N,NZ,LZ
COMMON /N6 I/ JMAX,PMAX,IFMAX,KSAME
COMMON /N7 I/ U1(A,I) /UTX/ UTX(A,I)
COMMON /USTORE/ USTORE(6,101,1)
COMMON /RNU/ RNU

C
F1(I)=1.-EXP(-I)/(1.-EXP(-I))
F2(I)=(1.-EXP(-I))/(1.-EXP(-I))
F3(I)=2.-EXP(-I)/(1.-EXP(-I))**2
N=6
NP=4
NU=2

C
C STREAMWISE MESH
C
IF (.NOT.ASUPPLY) GO TO 90
CALL AMESH
GO TO 150
90 CONTINUE
HKA(I)=HKS
KI=1
KL=1
KI=-HKS
A1=-HKS
C
100 CONTINUE
IF (KL.GT.100).OR.(XI.GT.AH) GO TO 140
XI=XI-HKA(KL)
IF (HKA(KL).GT.XPT) FACX=1.
IF (XI-GE-(KI-XPT)) GO TO 110
HKA(KI+1)=HKA(KL)*FACX
KL=KL+1
GO TO 100
110 CONTINUE
IF (FACX.EQ.1.) GO TO 130
IF ((XI-KI-XPT).GT.(HKA(KL)/20.)) GO TO 120
HKA(KI+1)=HKA(KL)*FACX*(KI-KI-XPT)
HKA(KL)=KI-XPT-(XI-HKA(KL))
KI=KI-XPT
KL=KL+1
KI=KI+1
GO TO 100
120 CONTINUE
HKA(KI+2)=HKA(KL)*FACX
HKA(KI+1)=KI-KI-XPT
HKA(KL)=KI-XPT-(XI-HKA(KL))
KL=KL+2
KI=KI+2

```

```

00006540
00006550
00006560
00006570
00006580
00006590
00006600
00006610
00006620
00006630
00006640
00006650
00006660
00006670
00006680
00006690
00006700
00006710
00006720
00006730
00006740
00006750
00006760
00006770
00006780
00006790
00006800
00006810
00006820
00006830
00006840
00006850
00006860
00006870
00006880
00006890
00006900
00006910
00006920
00006930
00006940
00006950
00006960
00006970
00006980
00006990
0007000
0007010
0007020
0007030
0007040

```

00007050
00007060
00007070
00007080
00007090
00007100
00007110
00007120
00007130
00007140
00007150
00007160
00007170
00007180
00007190
00007200
00007210
00007220
00007230
00007240
00007250
00007260
00007270
00007280
00007290
00007300
00007310
00007320
00007330
00007340
00007350
00007360
00007370
00007380
00007390
00007400
00007410
00007420
00007430
00007440
00007450
00007460
00007470
00007480
00007490
00007500
00007510
00007520
00007530
00007540
00007550

```

GO TO 100
130 CONTINUE
MX(KL)=A1*EXP1-(A1-HK*(KL))
X1=K1*EXP1
135 CONTINUE
MX(KL)=X1-KPI
K1=K1+1
KL=KL+1
X1=X1-MX(KL)
IF (A1.LE.AH) GO TO 135
140 CONTINUE
C
NR=KL-1
150 CONTINUE
C
IF (OPT1) CALL PRMF5H
C
C VERTICAL MESH
C
IF (.NOT.YSUP1) GO TO 350
CALL YMF5H
GO TO 400
350 CONTINUE
J1=J-1
JA=J-1+3./5.
J2=J-1-JA
J3=J-1-JA
J4=J-1-JA
DO 160 L=1,JA
H(L)=H1
160 CONTINUE
H1=2*YH/3./JC
DO 170 L=H1,J1
M(L)=H1
170 CONTINUE
M1=M1
400 CONTINUE
IF (.NOT.ZSUP1) GO TO 405
CALL ZMF5H
GO TO 420
405 DO 410 L=2,1
410 MX(L-1)=.1
420 CONTINUE
C
C OBTAIN INITIAL PROFILE
C
XPI=AA
YPI=YA
ZPI=ZA

```

C ***** THE NEXT DE CARD MUST BE CHANGED FOR EACH CASE STUDY *****
C *

```

C *
C *****
      UL=1.0
      PROD=SUM(I=1,N)*H(U/UL)
      YPT=PROD*YPT
      LMAX=MAX(I=1,N,K)
      WHI(16,5000)
      L=1
C
C PRINT MESH POINTS
C
      WRITE(16,5000) L,XPT,YPT,ZPT,YPT
      DO 450 L=2,LMAX
      LM=L-1
      XOUT=0.0
      YOUT=0.0
      ZOUT=0.0
      YYOUT=0.0
      IF (L.GT.NX) GO TO 430
      AP=XPT*HKX(L)
      XOUT=XPT
      430 IF (L.GT.J) GO TO 440
      YPT=YPT+H(L,M)
      YOUT=YPT
      YYOUT=YOUT+PROD
      440 IF (L.GT.1) GO TO 445
      ZPT=ZPT+HK7(L,M)
      ZOUT=ZPT
      445 WRITE(16,5000) L,XOUT,YOUT,ZOUT,YYOUT
      450 CONTINUE
      IF (USUPPLY) RETURN
C
      JI=J-1
C
C SOLVES FOR INITIAL SIMILAR SOLUTION
C
      IF (KSTAR.EGT.1) RETURN
      Y=YA
      H(J)=0.
      DO 190 L=1,J
      DO 190 K=1,N
      UTX(K,L)=0.
      190 CONTINUE
      DO 200 L=1,J
      U(1,L)=F(Y)
      U(2,L)=F2(Y)
      U(3,L)=F3(Y)
      U(4,L)=U(1,L)
      U(5,L)=U(2,L)
      U(6,L)=U(3,L)
      Y=Y+H(J)

```



```

00008570
00008580
00008590
00008600
00008610
00008620
00008630
00008640
00008650
00008660
00008670
00008680
00008690
00008700
00008710
00008720
00008730
00008740
00008750
00008760
00008770
00008780
00008790
00008800
00008810
00008820
00008830
00008840
00008850
00008860
00008870
00008880
00008890
00008900
00008910
00008920
00008930
00008940
00008950
00008960
00008970
00008980
00008990
00009000
00009010
00009020
00009030
00009040
00009050
00009060
00009070

LUMMUM /PHEPBY/ IFLAG,YIC,YTOLD,YTOLD2,IER,YICC
COMMON /KIC2/ K1,K2
C ICI AND IC2 CAN BE ASSIGNED VALUES BETWEEN 0 AND 3 BY CHANGING THE
C NEXT DATA STATEMENT
C DATA ICI,IC2/0.0,0.0/
C U HAS THE SAME MEANING AS ICI IN THE MAIN PROGRAM
C
IWM1=I/I-1
TERM=SQRT(1.0*(WE/WE)**2)
S=X
H=UE*5/14.216862E-5
NSORT=SQRT(RI)
IF (ITEM.LT.5) GO TO 90
IF (IFLAG-EO.0) GO TO 400
IF ((YIC-YTOLD).EQ.(-YTOLD-YTOLD2)) .AND. (YTOLD.NE.YIC)) IFLAG=0
IF (IFLAG.FQ.1) GO TO 90
YIC=AMAX1(YIC,YTOLD)
YI=-.435*YIC/.125
GO TO 400
90 CONTINUE
YTOLD2=YTOLD
YTOLD=YIC
YI=YH
JI=J-1
C
C DETERMINE POINT WHERE SECOND LAYER STARTS
C
IF (I2.EQ.1) GO TO 99
DO 95 LL=1,JI
L=J-LL
YI=YI-H(LL)
FP=-.25*(U(2,L)+UX(2,L)+US(2,L+12)+US(2,L+12M1))
GP=-.25*(U(5,L)+UX(5,L)+US(5,L+12)+US(5,L+12M1))
UMGM=ABS(TERM-SORT(FP**2+(WE*GP/UE)**2))/TERM
IF (UMGM.LT.0.01) GO TO 95
GO TO 200
95 CONTINUE
GO TO 200
99 CONTINUE
DO 100 LL=1,JI
L=J-LL
YI=YI-H(LL)
FP=.5*(U(2,L)+UX(2,L))
GP=.5*(U(5,L)+UX(5,L))
UMGM=ABS(TERM-SORT(FP**2+(WE*GP/UE)**2))/TERM
IF (UMGM.LT.0.01) GO TO 100
GO TO 200
100 CONTINUE
200 CONTINUE

```

```

0009086
0009090
0009100
0009110
0009120
0009130
0009140
0009150
0009160
0009170
0009180
0009190
0009200
0009210
0009220
0009230
0009240
0009250
0009260
0009270
0009280
0009290
0009300
0009310
0009320
0009330
0009340
0009350
0009360
0009370
0009380
0009390
0009400
0009410
0009420
0009430
0009440
0009450
0009460
0009470
0009480
0009490
0009500
0009510
0009520
0009530
0009540
0009550
0009560
0009570
0009580

YTC=YI*.125/.435
400 CONTINUE
YTC=YTC
PH000=.125*RSQRT*YI*YI*.125
Y=0.
IF (I1Z.NE.1) GO TO 450
DO 300 L=1,J
FP=.5*(U(2,L)+UX(2,L))
GP=.5*(U(4,L)+UX(4,L))
FPP=.5*(U(3,L)+UX(3,L))
GPP=.5*(U(6,L)+UX(6,L))
PH001=SQRT(FPP*FPP+(GP*GPP/UE1)**2)
PH002=1.0-TC1*(1-PK2)*S/(FPP*(RSQRT*(K2*S*YI))
PH003=1.0
PH004=.435*RSQRT*YI*YI*.435
IF (Y.GE.YTC) GO TO 250

C FIRST LAYER
C
C(2,L)=1.0*PRUN4*PR001*PH002
G(6,L)=1.0*PRUN4*PR001*PH003
GO TO 290

C SECOND LAYER
C
C
250 CONTINUE
PH002=1.0
G(2,L)=1.0*PRUN0*PR001*PH002
G(6,L)=1.0*PRUN0*PR001*PH003
290 Y=Y+H(L)
300 CONTINUE
DO 450 L=2,J
L1=L-1
G(3,L)=G(3,L1)+G(2,L1)*UX(3,L1)*UX(3,L1)-(G(2,L11)*UX(3,L11)+UX(3,L11)*UX(6,L11)
G(4,L)=G(4,L1)+G(6,L1)*UX(6,L11)*UX(6,L11)-(G(6,L11)*UX(5,L11)+UX(5,L11)*UX(6,L11)
350 CONTINUE
DO 460 L=1,J
FP=.25*(U(2,L)+UX(2,L))+UX(2,L12)+UX(2,L12)+UX(3,L12)+UX(3,L12)+UX(3,L12)+UX(3,L12)
GP=.25*(U(4,L)+UX(4,L))+UX(4,L12)+UX(4,L12)+UX(5,L12)+UX(5,L12)+UX(5,L12)+UX(5,L12)
GPP=.25*(U(3,L)+UX(3,L))+UX(3,L12)+UX(3,L12)+UX(6,L12)+UX(6,L12)+UX(6,L12)+UX(6,L12)
PH001=SQRT(FPP*FPP+(GP*GPP/UE1)**2)
PH002=1.0-TC1*(1-PK2)*S/(FPP*(RSQRT*(K2*S*YI))
PH003=1.0-TC2*(1-PK1)*S/(GPP*(RSQRT*(K1*S*YI))
PH004=.435*RSQRT*YI*YI*.435
IF (Y.GE.YTC) GO TO 475

C FIRST LAYER
C
C

```

```

00009590
00009600
00009610
00009620
00009630
00009640
00009650
00009660
00009670
00009680
00009690
00009700
00009710
00009720
00009730
00009740
00009742
00009750
00009752
00009760
00009770
00009780
00009790
00009800
00009810
00009820
00009830
00009840
00009850
00009860
00009870
00009880
00009890
00009900
00009910
00009920
00009930
00009940
00009950
00009960
00009970
00009980
00009990
00010000
00010010
00010020
00010030
00010040
00010050
00010060
00010070

G(2,L)=1.0*PRUD1*PRUD1*PRUD2
G(6,L)=1.0*PRUD4*PRUD1*PRUD3
GO TO 490

C SECOND LAYER
C
4/5 CONTINUE
PRUD2=1.0
PRUD3=1.0
G(2,L)=1.0*PRUD8*PRUD1*PRUD2
G(6,L)=1.0*PRUD8*PRUD1*PRUD3
490 Y=Y+H(L)
400 CONTINUE
DO 500 L=2,J
LJ=L-1
G(3,L)=G(3,L)+G(2,L)-G(1,L)*(UX(3,L)+US(3,L)+ZM1)
X -G(2,L)-G(1,L)*(UX(3,L)+US(3,L)+ZM1)+US(3,L)+Z1)
G(4,L)=G(4,L)+G(6,L)-G(5,L)*(UX(6,L)+US(6,L)+ZM1)+
X (G(6,L)-G(5,L))*(UX(6,L)+US(6,L)+ZM1)
500 CONTINUE
510 CONTINUE
DO 550 L=1,J
G(1,L)=G(2,L)
G(5,L)=G(6,L)
550 CONTINUE
RETURN
END
SUBROUTINE OUTPUT
C THIS SUBROUTINE WRITES THE SOLUTION ON PAPER FOR THE PLANE X=IN.
C
COMMON /ULAST/ ULAST(6,101,1)
COMMON /USTONE/ U(6,101,1)
COMMON /PARM4/ I,J,K
COMMON /PARMS/ FACX*MXS,AA,XH,YA,YB,ZA,ZH,NX
COMMON /PARM6/ KDM1,XN,MX,XZ,MH,Z
COMMON /EXPLOT/ XPLN(13)
DIMENSION IFSIN(5)
DATA IFSIN/0/
IF IFSIN-NE-0-AND-KDM1-EO-1 GO TO 310
WRITE(6,600)
II=1
DO 300 L=1,11
DO 300 L=1,J
WRITE(6,6000) LZ/1.0/(K+L+LZ),K=1,6)
300 CONTINUE
310 IFSIN=1
IF (KDM1-EO-1) IP=1
APLOT(IP)=IN

```

```

00010080
00010090
00010100
00010110
00010120
00010130
00010140
00010150
00010160
00010170
00010180
00010190
00010200
00010210
00010220
00010230
00010240
00010250
00010260
00010270
00010280
00010290
00010300
00010310
00010320
00010330
00010340
00010350
00010360
00010370
00010380
00010390
00010400
00010410
00010420
00010430
00010440
00010450
00010460
00010470
00010480
00010490
00010500
00010510
00010520
00010530
00010540
00010550
00010560
00010570

* (KIND=ED,1) XPL0(IIP)=XA
IP3=IP-J
DO 400 X=1,5
  IT=ITSTOR(K)
DO 400 L=1,4
  ULAST(IP,L,IT)=U(2,L,IT)
  ULAST(IP3,L,IT)=U(5,L,IT)
400 CONTINUE
  IP=IP+1
RETURN
6300 FORMAT(//2X,=1,=3X,=1,=7X,=F,=12X,=D,=10X,=ENDF,=11X,=G,=12X,
  X=IG,=10X,=DUG,=//)
6400 FORMAT(21A,6(1X,=12=5))
END
SUBROUTINE ZMESH
C
C THIS SUBROUTINE ALLOWS THE USER TO SUPPLY HIS OWN Z-MESH. HE
C SHOULD ALSO SUPPLY VALUES FOR THE FOLLOWING VARIABLES
C I---NUMBER OF Z-POINTS (MAXIMUM ALLOWED IS 11.
C OTHERWISE, THE DIMENSION IN THE MAIN PROGRAM OF THE
C VARIABLE% HKZ% USTORE, AND ULAST MUST BE INCREASED).
C ZA---MINIMUM VALUE OF Z, USUALLY ZA=0.0
C ZH---MAXIMUM VALUE OF Z
C THE MESH IS DEFINED BY HZ(L) WHERE Z(L+1)=Z(L)+HZ(L) WITH Z(1)=ZA
C AND Z(11)=ZH.
C IN THE EXAMPLE BELOW A UNIFORM Z-MESH IS PRESCRIBED.
C
COMMON /MESHZ/ HZ(11)
COMMON /PARAM/ I,J,L
COMMON /PARAMS/ FACK=HKZ,XA,XH,YA,YB,ZA,ZH,NX
ZA=0.0
ZH=1.0
Z=ZA
I=1
DO 10 L=1,10
  HZ(L)=1
Z=Z+HZ(L)
I=I+1
IF (ABS(Z-ZH).LE-.0E-5) GO TO 20
IF (Z-ZH).GT.0 GO TO 20
10 CONTINUE
20 RETURN
END
SUBROUTINE VAL_KT(XSTART)
C
C THIS SUBROUTINE COMPUTES THE SOLUTION ON THE 3-D WALL JET PROBLEM
C AND THE 2-D ATTACHMENT LINE EQUATION.
C
LOGICAL SFPHAI
LOGICAL RFL,XSUP,Y,YSUP,Y,ZSUP,Y,REFINE,USUPPLY
LOGICAL INEPT

```

[illegible]

00011890
00011100
00011110
00011120
00011130
00011140
00011150
00011160
00011170
00011180
00011190
00011200
00011210
00011220
00011230
00011240
00011250
00011260
00011270
00011280
00011290
00011300
00011310
00011320
00011330
00011340
00011350
00011360
00011370
00011380
00011390
00011400
00011410
00011420
00011430
00011440
00011450
00011460
00011470
00011480
00011490
00011500
00011510
00011520
00011530
00011540
00011550
00011560
00011570
00011580
00011590

```

      ZN=ZA
      Z=ZA
      IF (KK.GT.6) GO TO 950
C  SET UP AVERAGING TO ELIMINATE UNNATURAL OSCILLATIONS CAUSED BY
C  ARBITRARY INITIAL CONDITIONS
      DO 900 L=1,J
        G(1,L)=GSAVE(1,L)
        G(5,L)=GSAVE(2,L)
      DO 100 K=1,N
        UT(K,L)=USTORE(K,L,1)
        ULAST(K,L,1)=5*(ULAST(K,L,1)+USTORE(K,L,1))
      900 UTX(K,L)=ULAST(K,L,1)
      GO TO 975
      950 CONTINUE
      DO 1000 L=1,J
        G(1,L)=GSAVE(1,L)
        G(5,L)=GSAVE(2,L)
      DO 1000 K=1,N
        UT(K,L)=USTORE(K,L,1)
      1000 UTX(K,L)=USTORE(K,L,1)
      975 CONTINUE
C  C SOLVE ATTACHMENT LINE EQUATION
      CALL NF10H(KASE?N,MP,M2)
      DO 980 L=1,J
        GSAVE(1,L)=G(2,L)
        GSAVE(2,L)=G(5,L)
      980 CONTINUE
      GO TO 1159
      1090 CONTINUE
      HZ=MKZ(1/4)
      ZN=ZN*HZ
      IF (KK.GT.6) GO TO 1095
      DO 1091 L=1,J
        G(1,L)=G(2,L)
        G(5,L)=G(6,L)
      DO 1091 K=1,N
        SAVUT(K,L)=UT(K,L)
        UTX(K,L)=5*(UTX(K,L)+UT(K,L))
        IF (L.EQ.2) UTX(K,L)=UT(K,L)
        ULAST(K,L,1Z)=5*(ULAST(K,L,1Z)+ULAST(K,L,1Z))
        USTON(K,L,1Z)=ULAST(K,L,1Z)
      1091 CONTINUE
      GO TO 1150
      1095 CONTINUE
      DO 1140 L=1,J
        DO 1160 K=1,N

```

```

1100 G1A(K,L)=U(K,L)
1105 G1(L)=G1(L)
1110 CONTINUE
C SOLVE 3-D WALL JET EQUATION
C CALL NEWTONKASE3,H,NP,NG)
1150 CONTINUE
C EXIT IF NO CONVERGENCE
C
IF (ITER.EQ.0) GO TO 2100
IF (I2.FQ.1) GO TO 1495
IF (K-K1.EQ.1) GO TO 1485
DO 1480 K=1,N
DO 1480 L=1,J
1480 USTORE(K,L,I2M)=SAVUT(K,L)
GO TO 1500
1485 CONTINUE
DO 1490 K=1,M
DO 1490 L=1,J
1490 USTORE(K,L,I2M)=UTX(K,L)
GO TO 1500
1495 IF (K-K1.EQ.1) GO TO 1500
DO 1498 K=1,N
DO 1498 L=1,J
1498 USTORE(K,L)=U1(K,L)
1500 CONTINUE
DO 1510 K=1,M
DO 1510 L=1,J
1510 USTORE(K,L,I2)=U1(K,L)
C CHECK PRINT OUT OPTIONS
C
IF (.AND.(OPTPI.AND.(I2N.GE.-KXMP+IPR1))) GO TO 1200
IF (OPTPI.AND.(I2NINT.EQ.KPRI(KXMP))) GO TO 1200
GO TO 1300
1200 CONTINUE
KXMP=KXMP+1
1250 CONTINUE
C WHITE SOLUTION ON PAPER
C CALL OUTPT
1300 CONTINUE
C
IF (UT(I2,J).GE.0.) GO TO 2050
C SEPARATION
C

```

```

00012110
00012120
00012130
00012140
00012150
00012160
00012170
00012180
00012190
00012200
00012210
00012220
00012230
00012240
00012250
00012260
00012270
00012280
00012290
00012300
00012310
00012320
00012330
00012340
00012350
00012360
00012370
00012380
00012390
00012400
00012410
00012420
00012430
00012440
00012450
00012460
00012470
00012480
00012490
00012500
00012510
00012520
00012530
00012540
00012550
00012560
00012570
00012580
00012590
00012600
00012610

C
WRITE(6,6200)
SEPRAT=.TIME.
NRSANF=KRWNT-1
GO TO 2100
2050 CONTINUE
IF(XN.HX) GO TO 2130
2000 CONTINUE
2100 CONTINUE
C
6000 FORMAT(' STATION = *14.0, X =*E12.5*, X-STEP =*E12.5)
6100 FORMAT(' 0(M2) SOLUTION')
6200 FORMAT('////////XXXXXXXXXXXXXXXX SEPARATION XXXXXXXXXXXXXXXX//')
C
CALL GUESS
NRS=NRSAVE
IF (ITER.NF .8) GO TO 2200
SEPRAT=.TIME.
NRS=NR-1
2200 RETURN
END
SUBROUTINE PRMESH
C
C THIS ROUTINE ALLOWS THE USER TO SUPPLY THE STREAMWISE STATIONS AT
C WHICH SOLUTIONS ARE TO BE PRINTED ON PAPER. THE FIRST STATION
C (INITIAL CONDITIONS) IS AUTOMATICALLY PRINTED BY THE PROGRAM.
C THEREFORE KPHI(L).GT.1 AND SHOULD HAVE THE PROPERTY THAT KPHI(L)
C IS GREATER THAN KPHI(K) FOR L GREATER THAN K. KPHI(M)=1 MEANS THAT
C THE MTH PRINT OUT WILL BE FOR THE PLANE K=1(). NOTE ALSO THAT THE
C LAST STATION IS AUTOMATICALLY PRINTED BY THE PROGRAM, SO IT NEED NOT
C BE INCLUDED IN THE VECTOR KPHI.
C IN THE SAME CASE BELOW, PRINTING OF THE SOLUTION OCCURS AT THE
C FIRST, LAST, AND FORTIETH STREAMWISE STATION.
C
COMMON /KPHI/ KPHI(1)
KPHI(1)=0
KPHI(2)=1
NRTURN
END
SUBROUTINE YMRSH
C
C THIS SUBROUTINE ALLOWS THE USER TO SUPPLY HIS OWN ETA MESH. HE
C SHOULD ALSO SUPPLY VALUES FOR THE FOLLOWING VARIABLES
C J--TOTAL NUMBER OF ETA POINTS (MAXIMUM OF 101)
C YA--MINIMUM VALUE OF ETA (USUALLY YA=0.0)
C YB--MAXIMUM VALUE OF ETA
C THE MESH IS DEFINED BY HY(L) WHERE ETAIL*Y)=ETAIL*HY(L) WITH
C ETAIL=YA AND FIATJ=14.
C THE SAMPLE FRAME STARTS WITH A GEOMETRIC STRETCHING UNTIL HY(L) IS
C .GT-.5. IF IT REMAINS FIATJ AT .5 UNTIL L.GT.50 WHEN IT IS SET
C TO.
C

```

```

00012620
00012630
00012640
00012650
00012660
00012670
00012680
00012690
00012700
00012710
00012720
00012730
00012740
00012750
00012760
00012770
00012780
00012790
00012800
00012810
00012820
00012830
00012840
00012850
00012860
00012870
00012880
00012890
00012900
00012910
00012920
00012930
00012940
00012950
00012960
00012970
00012980
00012990
00013000
00013010
00013020
00013030
00013040
00013050
00013060
00013070
00013080
00013090
00013100
00013110
00013120

COMMON /MESH/ MX(1)
COMMON /PARAM/ I,J,17
COMMON /PARAMS/ FACI,MKS,AA,XH,YA,Y3,ZA,ZH,NX
YA=0.0
YH=60.0
J=2
MY(1)=.005
Y=MY(1)
DO 10 L=2,100
MY(L)=1.2*MY(L-1)
IF (MY(L).GT.50) MY(L)=.5
IF (L.GT.50) MY(L)=.75
Y=Y*MY(L)
J=J+1
IF (ABS(Y-YH).LT.1.0E-5) GO TO 20
IF (Y-GE-YH) GO TO 20
10 CONTINUE
20 Y=Y
RETURN
END
SUBROUTINE XPMESH
C THIS SUBROUTINE ALLOWS THE USER TO SUPPLY HIS OWN X-MESH. HE
C SHOULD ALSO SUPPLY VALUES FOR THE FOLLOWING VARIABLES
C NX---TOTAL NUMBER OF X-POINTS MINUS ONE (MAX 401)
C XA---INITIAL X-PLANE
C XH---FINAL X-PLANE (MAXIMUM VALUE OF X)
C THE MESH IS DEFINED BY MX(L) WHERE X(L)=X(L-1)+MX(L) WITH X(1)=XA
C AND X(NX)=XH. NOTE ONLY MX(2),MX(3),...,MX(NX) NEED BE LOADED.
C MX(NX) IS ALWAYS AUTOMATICALLY DECREASED SO THAT X(NX)=XH.
C IN THE EXAMPLE BELOW NX IS SET TO 4001 FOR THE FIRST 9 STEPS THAN
C QUANTIC STRETCHING ALGORITHM INCREASES THE NET UNTIL XH IS REACHED
COMMON /MESH/ MX(1)
COMMON /PARAMS/ FACI,MKS,AA,XH,YA,Y3,ZA,ZH,NX
XA=1.0
XH=10.0
MX(1)=.0001
X=XA
DO 10 L=2,401
NX=L
MX(L)=1.2*MX(L-1)
IF (L.LT.11) MX(L)=.0001
X=X+MX(L)
IF (ABS(X-XH).LT.1.0E-5) GO TO 20
IF (X.GT.XH) GO TO 20
10 CONTINUE
20 RETURN
END
SUBROUTINE MESH

```

```

C NET SELECTION - APPROXIMATELY CHOOSING HIKI WITH THE TRUNCATION ERROR
C TO BE A CONSTANT ON THE WHOLE INTERVAL
C
COMMON /DH/ US(6,1) /IT/ UT(6,1) /HMX/ HMX(1) /HESHY/ H(1)
COMMON /F/ F(6,1)
COMMON /NY/ JMAX, HMAX, HMX, KSAME
COMMON /PARM/ N, NP, NO
COMMON /PARMS/ I, J, IZ
COMMON /NYI/ KTYPE, KSINGK, KSING(1), SL, P(10), Q(6), R(6), IAU2(6)
COMMON /SETUP/ UN(6), SUM(6), UN2(6), FF(6), AJA(6,6)
DIMENSION Z(2000), KI(10)
LOGICAL DEL1, DEL2
C
KSINGK=1
KSING(1)=1
NEXCED=0
NETINC=0
AREA=0
KSAFE=0
IFMX=1
JI=J-1
DO 10 A=1,N
SUM(K)=0.
10 CONTINUE
C
C COMPUTE LOCAL TRUNCATION ERROR AT MID-POINT
C
KTYPE=-1
DO 300 L=1,J
KL=L
IF (L.EQ.1) GO TO 100
KTYPE=0
IF (L.EQ.KSINGK) GO TO 50
IF (L.NE.KSINGK) GO TO 100
C POINT BEFORE SINGULARITY ON RIGHT END-POINT
C
KTYPE=1
GO TO 100
50 CONTINUE
C POINT AFTER SINGULARITY
C
KTYPE=-1
KSINGK=KSINGK+1
100 CONTINUE
DO 150 K=1,N
UN(K)=(UT(K,L)-UN(K,L+1))/2.
150 CONTINUE

```

```

00013130
00013140
00013150
00013160
00013170
00013180
00013190
00013200
00013210
00013220
00013230
00013240
00013250
00013260
00013270
00013280
00013290
00013300
00013310
00013320
00013330
00013340
00013350
00013360
00013370
00013380
00013390
00013400
00013410
00013420
00013430
00013440
00013450
00013460
00013470
00013480
00013490
00013500
00013510
00013520
00013530
00013540
00013550
00013560
00013570
00013580
00013590
00013600
00013610
00013620
00013630

```

```

CALL IHUN
T=0.
DO 200 N=1,N
  F(K,L)=1AD2(K)
  T=T+AD2(K)**2
200 CONTINUE
  T=SQRT(T)
  Z(L)=T
  AREA=AREA+H(L)**2*T/2.
300 CONTINUE
  C
  CONSTC=AREA/J
  C
  C NET SELECTION
  C
  LA=1
  DO 2000 KOUT=1,KSLNGK
    LM=KSLNG(KOUT)-1
    IF (KOUT.GT.1) LA=KSLNG(KOUT)
    DEL1=.FALSE.
    DEL2=.FALSE.
    DO 1000 L=LA,LM
      FACI=H(L)**2*Z(L)/(2.*CONSTC)
      FACT=SQRT(FACT)
      Z(L)=FACT
      KJS=FACI*.5
      IF (KJS.LE.1) GO TO 700
    C
    C ADDITION
    C
    IF ((L+KJS.NETINC+1).GT._MAX) GO TO 2500
    ASAME=1
    IF MX=MAX0(IFMX,KJS)
      MI=H(L)/KJS
    DO 600 M=1,KJS
      HWE=MI*INC*M-1+L)=HI
    DO 600 K=1,N
      USIR=NETINC*M-1+1)=((M-1)*OUT(K,L+1)+(KJS-(M-1))*OUT(K,L))/KJS
    600 CONTINUE
    NETINC=NETINC+KJS-1
    DEL1=.FALSE.
    DEL2=.FALSE.
    GO TO 1000
  700 CONTINUE
  C
  IF (FACT.GF.0.5) GO TO 800
  DEL2=.TRUE.
  IF (DEL1) GO TO 900
  800 CONTINUE
  C
  900 CONTINUE
  C
  END

```

```

      IF (IL*NETINC+1).GT.JMAX) GO TO 2500
      DELI=DEL2
      DO 150 K=1,M
        US(K,L*NETINC)=UT(K,L)
      150 CONTINUE
      HNEW(L*NETINC)=H(L)
      GO TO 1000
    900 CONTINUE

C     DELETION
C
      DELI=DEL2
      IF (HNEW(L*NETINC-1)*H(L)).LE.HMAX) GO TO 825
      NACED=1
      GO TO 800
    825 CONTINUE
      KSAME=1
      HNEW(L*NETINC-1)=HNEW(NETINC+1-1)*H(L)
      NETINC=NETINC-1
    1000 CONTINUE
      KT(KOUT)=K*SIG(KOUT)*NETINC
    2000 CONTINUE
      DO 2100 K=1,K*SIG(K)
        K*SIG(K)=KT(K)
      CONTINUE
    2100 IF (K*SIG-K*0.0) HF IJHHW

C
      JNEW=J*NETINC
      DO 2200 K=1,M
        US(K,JNEW)=UT(K,J)
      CONTINUE
      J=JNEW
      HNEW(J)=0.
      DO 2300 L=1,J
        H(L)=H*H*H(L)
      DO 300 K=1,M
        UT(K,L)=US(K,L)
      CONTINUE
      HT TWIN
    2300 CONTINUE

C
    2500 CONTINUE
      KSAME=0
      KE IJHH
      ENP
      SUMH(I)H= IJHHW

C THIS SUBROUTINE COMPUTES THE LOCAL TRUNCATION ERROR OF THE BOX
C SCHEME. THE INTEGER VARIABLE ** TYPE ** INDICATES THE FOLLOWING
C TYPE = -1 LEFT BOUNDARY POINT

```

```

00014150
00014160
00014170
00014180
00014190
00014200
00014210
00014220
00014230
00014240
00014250
00014260
00014270
00014280
00014290
00014300
00014310
00014320
00014330
00014340
00014350
00014360
00014370
00014380
00014390
00014400
00014410
00014420
00014430
00014440
00014450
00014460
00014470
00014480
00014490
00014500
00014510
00014520
00014530
00014540
00014550
00014560
00014570
00014580
00014590
00014600
00014610
00014620
00014630
00014640
00014650

```

```

C      = 0 INTERNAL POINT
C      = 1 RIGHT BOUNDARY POINT
C SINGULAR POINTS ARE TREATED AS BOUNDARY POINTS
C
C      INTEGER TYPE
COMMON /MESH/ N11 /M1 /UT(6,1)
COMMON /PARAM/ N,MP,NO
COMMON /PARAMS/ FACX,HKS,AA,IX,YA,YB,ZA,ZH,NX
COMMON /MESH/ TYPE,KSTICK,KSTING(1),L,P1(1),P(1),R(1),T(1)
COMMON /SFTRP/ UH(16),UH(16),UD(16),UD(16),FF(16),A(16,6)
C
C IF (TYPE) 100, 200, 300
100 CONTINUE
C
C LEFT BOUNDARY POINT
C
C      A1=-H(L1)/2.
C      A2=-A1
C      A3=A2+H(L1)*1
C      A4=A3+H(L1)*2
C      A5=A4+H(L1)*3
C
C      L1=L
C      L2=L+1
C      L3=L+2
C      L4=L+3
C      L5=L+4
C
C      GO TO 400
200 CONTINUE
C
C INTERNAL POINT
C
C IF (L,F0,(KSTING(KSTING)-2)) GO TO 250
C
C      A1=-H(L1)-H(L1)/2.
C      A2=-H(L1)/2.
C      A3=H(L1)/2.
C      A4=A3+H(L1)*1
C      A5=A4+H(L1)*2
C
C      L1=L-1
C      L2=L
C      L3=L+1
C      L4=L+2
C      L5=L+3
C
C      GO TO 400
250 CONTINUE
C
C SPECIAL TREATMENT FOR 1 - (KSTING(KSTING) - 2) BECAUSE OF THIRD INFLUENTIATION

```

00014660
00014670
00014680
00014690
00014700
00014710
00014720
00014730
00014740
00014750
00014760
00014770
00014780
00014790
00014800
00014810
00014820
00014830
00014840
00014850
00014860
00014870
00014880
00014890
00014900
00014910
00014920
00014930
00014940
00014950
00014960
00014970
00014980
00014990
00015000
00015010
00015020
00015030
00015040
00015050
00015060
00015070
00015080
00015090
00015100
00015110
00015120
00015130
00015140
00015150

4-32


```

00015960
00015970
00015980
00015990
00016000
00016010
00016020
00016030
00016040
00016050
00016060
00016070
00016080
00016090
00016100
00016110
00016120
00016130
00016140
00016150
00016160
00016170
00016180
00016190
00016200
00016210
00016220
00016230
00016240
00016250
00016260
00016270
00016280
00016290
00016300
00016310
00016320
00016330
00016340
00016350
00016360
00016370
00016380
00016390
00016400
00016410
00016420
00016430
00016440
00016450
00016460

DIMENSION X(5)
C
K(1)=X1
K(2)=X2
K(3)=X3
K(4)=X4
K(5)=X5
N1=N-1
VANDT=1.
DO 200 L=1,N1
  LI=L+1
  DO 100 K=LI,N
    VANDT=VANDT*(K(K)-X(L))
  100 CONTINUE
  200 CONTINUE
C
RETURN
END
SUBROUTINE PROFILE
C
THIS SUBROUTINE SETS UP THE INITIAL PROFILES AND CALLS THE MESH
REFINEMENT ROUTINE IF REFIN=.TRUE.. THE USER MAY CHANGE THE
PROFILES BY LOADING NEW VALUES INTO THE MATRIX USTORE.
C
LOGICAL REFIN
COMMON /OPTIM/ ASUPPLY,VSUPPLY,ZSUPPLY,REFINE,USUPPLY
COMMON /UTX/ UTX(6,1)
COMMON /UT/ U(6,1)
COMMON /MESH/ H(1)
COMMON /MESH/ HKX(1)
COMMON /NET/ JMAX,JMAX,JFMX,KSAME
COMMON /PARMS/ I,J,LZ
COMMON /PARMS/ FACT,HRKS,AA,XH,YA,YB,ZA,ZH,NK
COMMON /PARMS/ KIMH,XNHR,XZM,HZ,C
COMMON /USTORE/ USTORE(6,10),1)
COMMON /ETA/ ETA(1)
DATA ISTAR/0/
VI=1.0
EI=EXP(-Y)
HLD=0.0
1 CONTINUE
DO 30 M=1,I
  Y=YA
  SUM=0.0
C
LOAD INITIAL PROFILES
C
USTORE(1,1,M)=0.0
A=1.5-.1*FLOAD(M-1)
B=(M-6).6) A=2.0-.1*FLOAD(M-6)

```

```

00016470      C=C*(1+Y1*(1-I.0))
00016480      C3=C2*E1
00016490      DO 30 L=1,J
00016500      C1A(L)=Y
00016510      IF (Y.GT.Y1) GO TO 10
00016520      C=EXP(-Y)
00016530      USTORE (2,L,M)=C3*Y.C2*(E-1.0)
00016540      USTORE (3,L,M)=C3-C2*E
00016550      GO TO 15
00016560      I0=EXP(-1.0*(Y-Y1)**2)
00016570      USTORE (2,L,M)=1.0*(A-1.0)*E
00016580      USTORE (3,L,M)=-2.0*(Y-Y1)*(A-1.0)*E
00016590      I5=USTORE (2,L,M)
00016600      IF (L.F0.1) GO TO 10
00016610      SUM=SUM+.5*U(L-1)*(I+1)
00016620      USTORE (1,L,M)=SUM
00016630      I1=I
00016640      USTORE (4,L,M)=USTORE (1,L,M)
00016650      USTORE (5,L,M)=USTORE (2,L,M)
00016660      USTORE (6,L,M)=USTORE (3,L,M)
00016670      Y=Y*U(L)
00016680      SUM=SUM
00016690      ADJNT=1
00016700      IF (ISTART.EQ.0) GO TO 099
00016710      WRITE (6,6000)
00016720      CALL OUTPT
00016730      RETURN
00016740
00016750
00016760      DO 60 L=1,I
00016770      DO 60 K=1,6
00016780      UTX(K,L)=USTORE (K,L,1)
00016790      60 UTX(K,L)=USTORE (K,L,1)
00016800      IF (.NOT.REFINE) RETURN
00016810
00016820
00016830
00016840
00016850
00016860
00016870
00016880
00016890
00016900
00016910
00016920
00016930
00016940
00016950
00016960
00016970

```

1 1 1
 099 CONTINUE
 30 CONTINUE
 ADJNT=1
 IF (ISTART.EQ.0) GO TO 099
 WRITE (6,6000)
 CALL OUTPT
 RETURN
 6000
 DO 60 L=1,I
 DO 60 K=1,6
 UTX(K,L)=USTORE (K,L,1)
 60 UTX(K,L)=USTORE (K,L,1)
 IF (.NOT.REFINE) RETURN
 C SET UP MESH REFINEMENT FOR THE ABOVE PROFILES
 C
 C
 KOUNT=2
 M=MAX(KOUNT)
 X=XA+X
 X=XA+.5*PI
 Z=0.0
 ZH=0
 MZ=0.0
 IZ=1
 CALL MESHIN
 CALL MESHOUT
 IZ=1
 GO TO 1
 6000 FORMAT(10X,'INITIAL PROFILES')
 6100 FORMAT('PROFL--NO CHANGE IN REFINEMENT')
 END

```

00016988
00016990
00017000
00017010
00017020
00017030
00017040
00017050
00017060
00017070
00017080
00017090
00017100
00017110
00017120
00017130
00017140
00017150
00017160
00017170
00017180
00017190
00017200
00017210
00017220
00017230
00017240
00017250
00017260
00017270
00017280
00017290
00017300
00017310
00017320
00017330
00017340
00017350
00017360
00017370
00017380
00017390
00017400
00017410
00017420
00017430
00017440
00017450

SUBROUTINE PREP6,2
C
C PREVIOUS STATION FOR ATTACHMENT LINE EQUATION
C
COMMON /GEX/ UX(6,1) /G/ G(6,1) /MESH/ MY(1)
COMMON /PANH1/ N.MP,NO
COMMON /PARM2/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10
COMMON /PANH/ I,J,L7
COMMON /PARM6/ K(MM1,MN,MX,X,Z,N,H,Z)
DIMENSION UM(6)
C
C COMPUTE P COEFFICIENTS
C
CALL PREPP(XN-MX,Z,8)
ALPHAN=P10/MX
DO 100 L=2,J
L1=L-1
M=MY(L1)
DO 10 K=1,M
10 UM(K)=-.5*(UX(K,L)+UX(K,L1))
G(3,L)=G(1,L)*UX(3,L)-G(1,L1)*UX(3,L1)-M*( (-ALPHAN*P2)*UM(2)+UM(3)-P2)
X
G(4,L)=G(5,L)*UX(6,L)-G(5,L1)*UX(6,L1)-M*( (ALPHAN-P4)*UM(2)+UM(5)
X
* (-ALPHAN*P1)*UM(1)+UM(6)+P3*(1.0-UM(5)+UM(5)*UM(6))
X
* P4)
100 CONTINUE
RETURN
END
SUBROUTINE PREP6
C
C PREVIOUS STATION FOR FUEL J-D RALL JET EQUATION
C
COMMON /MESH/ MY(1) /UX/ UX(6,1) /G/ G(6,1)
COMMON /JSTORE/ JS(G,10),1)
COMMON /PANH1/ N.MP,NO
COMMON /PARM2/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10
COMMON /PANH/ I,J,L7
COMMON /PARM6/ K(MM1,MN,MX,X,Z,N,H,Z)
DIMENSION UM(6),UMAX(6),J1(6)
L(M)=17-1
C
C COMPUTE P COEFFICIENTS
C
CALL PREPP(X,Z,1)
TEMP=2.0*P10
PM(N1)=TEMP/MX
PM(N2)=TEMP*P7/H7
DO 100 L=2,J
L1=L-1
M=MY(L1)
DO 10 K=1,M

```

[illegible]

[illegible]

```

DO 40 I=1,N
DO 40 K=1,N
40 A(J(K,I))=0.
DO 50 K=1,N
WH(K)=U(K,1)
50 WHIN(K)=U(K,J)

```

```
CALL HC
DO 90 K=1,NP
DO 80 L=1,N
80 A(K,L,1)=A(JA(K,L)
90 F(K,1)=-F(K)
DO 200 K=1,MF
DO 100 L=1,N
100 A(K+MF,L,1)=A(JA(K
200 F(K+MF,1)=-F(K+MF
```

```
DO 210 L=1,N
  UD 210 A=1,N
  210 AJA(K,L)=0.
```

```

1/M)=1/2-1
DO 900 K=2, J
  M=M-1
  IF (KASE.EQ.2) GO TO 295
  DO 290 N=1, N
    UM(K)=.5*(UM(K)+1) * UM(N)
    UM(N)=.5*(UM(N)+1) * UM(K)
    UM(NK)=.5*(UM(K)+1) * USSTORE(K, M, IZ)
    UM(NK)=.5*(USSTORE(N, M, IZ)+1) * UM(N)
  CONTINUE
  GO TO 310
295 CONTINUE
DO 300 K=1, N
  UM(NK)=UM(K)+1) * UM(N) / 2.

```



```

1200 CONTINUE
  DO 1400 M=1,J
    DO 1300 L=1,N
      T=A(NP+KSWITCH+L,M)
      A(NP+KSWITCH+L,M)=B(K-NQ+L,M+1)
      B(K-NQ+L,M+1)=T
      T=C(KSWITCH+L,M)
      C(KSWITCH+L,M)=A(K-NQ+L,M+1)
      A(K-NQ+L,M+1)=T
1300 CONTINUE
      T=F(NP+KSWITCH+M)
      F(NP+KSWITCH+M)=F(K-NQ,M+1)
      F(K-NQ,M+1)=T
1400 CONTINUE
1500 CONTINUE
C
      RETURN
      END
      SUBROUTINE BLOCK1
C THIS SUBROUTINE DECOMPOSES THE BLOCK TRIAGONAL MATRIX INTO LU-FORM
C
      COMMON /A/ A(6,6,1) /H/ H(6,6,1) /C/ C(2,6,1)
      COMMON /NH/ NH(6,1) /NC/ NC(6,1) /NCR/ NCR(6,1)
      COMMON /PARM1/ N,NP,NQ
      COMMON /PARM2/ I,J,I2
      DO 100 L=1,J
        DO 100 K=1,N
          NH(K,L)=K
          NC(K,L)=K
100 NCR(K,L)=0
      CALL LU50R.V(1)
C
      DO 600 M=2,J
        M1=M-1
        KM=M
      CALL HETASV(KM)
C
      C SOLVE SCALAR MATRIX ALPHA
C
      DO 500 N=1,NP
        DO 400 L=1,N
          SUM=0.
          DO 200 KK=1,NQ
            SUM=SUM+H(K,K+NP,M)*C(KK+L,M)
          A(K,L,M)=A(K,L,M)+SUM
400 CONTINUE
500 CONTINUE
C
      CALL LU50R.V(KM)

```

```

C
C
C      400 CONTINUE
C
C      RETURN
C      END
C      SUBROUTINE LUSOLV(KM)
C THIS SUBROUTINE DECOMPOSES A SCALAR MATRIX INTO LU FORM USING
C A MIXED-PIVOTING STRATEGY
C
C      COMMON /A/ A(6,6,1)
C      COMMON /NR/ NR(6,1) /NC/ NC(6,1) /NCH/ NCH(6,1)
C      COMMON /PARM1/ N,ND,NQ
C
C      DO 600 M=2,N
C      M1=M-1
C      C SEARCH FOR OPTIMAL PIVOT
C
C      KH=M
C      KC=M
C      NCH=NR(M,KM)
C      NCH=NC(M,KM)
C      C PIVOT=A(NRM,NCH,KM)
C      RPIVOT=CPIVOT
C      DO 200 K=M,N
C      NKR=NR(K,KM)
C      NCK=NC(K,KM)
C      IF (ABS(NRPIVOT).GE.ABS(A(NRM,NCH,KM))) GO TO 100
C
C      KH=K
C      KC=M
C      RPIVOT=A(NRM,NCH,KM)
C      C PIVOT=A(NRM,NCH,KM)
C      100 CONTINUE
C      IF (ABS(CPIVOT).GE.ABS(A(NRM,NCH,KM))) GO TO 200
C      KC=K
C      C PIVOT=A(NRM,NCH,KM)
C      200 CONTINUE
C      IF (ABS(NRPIVOT).GE.ABS(CPIVOT)) GO TO 400
C
C      C PIVOT BY INTERCHANGING COLUMN
C
C      IF (ABS(CPIVOT).LT.1.E-10) WHIF(6,6000) KM,M1,CPIVOT
C      NCR(M,KM)=1
C      KI=NC(KC,KM)
C      IF (KC.NE.M) KSIGN=KSIGN+1
C      NC(KC,KM)=NC(M,KM)
C      NC(M,KM)=KI
C
C      C GAUSSIAN ELIMINATION
C

```

```

00020770
00020780
00020790
00020800
00020810
00020820
00020830
00020840
00020850
00020860
00020870
00020880
00020890
00020900
00020910
00020920
00020930
00020940
00020950
00020960
00020970
00020980
00020990
00021000
00021010
00021020
00021030
00021040
00021050
00021060
00021070
00021080
00021090
00021100
00021110
00021120
00021130
00021140
00021150
00021160
00021170
00021180
00021190
00021200
00021210
00021220
00021230
00021240
00021250
00021260
00021270

DO 300 L=M,N
NCL=NC(L,KM)
A(NM,NCL,KM)=A(NM,NCL,KM)/CPIVOT
I=A(NM,NCL,KM)
DO 100 K=M,N
NRK=NR(K,KM)
300 A(NM,NCL,KM)=A(NM,NCL,KM)-I*A(NRK,KI,KM)
GO TO 680

C
C PIVOT BY INTERCHANGING ROW
C
400 CONTINUE
IF (ABS(RPIVOT).LT.1.E-10) WRITE(6,6800) KM,M1,RPIVOT
KI=NR(KI,KM)
IF (KR.NE.M1) KSIGN=1
NR(KR,KM)=NR(M1,KM)
NR(M1,KM)=KI

C
C GAUSSIAN ELIMINATION
C
DO 500 L=M,N
NKL=NR(L,KM)
A(NKL,NCM,KM)=A(NKL,NCM,KM)/RPIVOT
I=A(NKL,NCM,KM)
DO 500 K=M,N
MCK=NC(K,KM)
500 A(NKL,NCK,KM)=A(NKL,NCK,KM)-I*A(KI,NCK,KM)

600 CONTINUE
NRN=NR(N,NM)
NCN=NC(N,NM)
IF (ABS(A(NRN,NCM,KM)).LT.1.E-10) WRITE(6,6100) KM,A(NRN,NCM,KM)
6000 FORMAT(' LU SOLV - BLOCK =',I4.0 PIVOT AT EQN NO.=',I2.0 IS=',E12.5)
6100 FORMAT(' LU SOLV - BLOCK =',I4.0 LAST PIVOT IS=',E12.5)

C
C RETURN
C
END
SUBROUTINE HELASV(NM)
C THIS SUBROUTINE SOLVES FOR HELA IN THE LU-DECOMPOSITION OF THE BLOCK
C TRIANGULAR MATRIX
C
COMMON /SETUP/ UN(6,10),UP2(6),UP3(6),FF(6),AJA(6,6)
COMMON /A/ A(6,6,1) /B/ B(6,6,1) /C/ C(2,6,1)
COMMON /NR/ NR(6,1) /NC/ NC(6,1) /NCK/ NCK(6,1)
COMMON /PARAM/ N,NP,NM
C SOLVE Y IN Y + (Ueq) = H
C
M1=KM-1
DO 700 M=1,NP

```

```

00021200      DO 300 L=1,N
00021210      NCL=NC(L,M)
00021220      SUM=H(M,NCL,KM)
00021230      IF (L.FO.1) GO TO 200
00021240      L=L-1
00021250      DO 100 K=1,L1
00021260      NKR=NR(K,M)
00021270      SUM=SUM-H(NKR,M)*H(NKR,MCL,M)
00021280      UNCL)=SUM
00021290      NCL=NR(L,M)
00021300      IF (NCR(L,M).EQ.0) UNCL)=UNCL)/A(NRCL,MCL,M)
00021310      300 CONTINUE
00021320
00021330      C SOLVE HETA IN HETA * A = H
00021340      C
00021350      C
00021360      DO 500 EL=2,M
00021370      L=N-EL+1
00021380      L1=L+1
00021390      SUM=H(L)
00021400      NCL=NC(L,M)
00021410      DO 400 K=1,L1
00021420      NKR=NR(K,M)
00021430      SUM=SUM-UN(K)*H(NKR,MCL,M)
00021440      UNCL)=SUM
00021450      NCL=NR(L,M)
00021460      IF (NCR(L,M).EQ.1) UNCL)=UNCL)/A(NRCL,MCL,M)
00021470      500 CONTINUE
00021480
00021490      C REARRANGE COMPONENTS THE TO MAIED PIVOTING
00021500      C
00021510      C
00021520      DO 600 L=1,M
00021530      NKR=NR(L,M)
00021540      H(M,NRCL,MCL,M)=H(L)
00021550      700 CONTINUE
00021560
00021570      RETURN
00021580      END
00021590
00021600      SUBROUTINE BLOCK2
00021610      COMMON /PARM1/ M,NO,NO
00021620      COMMON /PARM2/ L,J,LZ
00021630      C SOLVE Y IN L * Y = F
00021640      C
00021650      C
00021660      DO 300 M=2,J
00021670      M1=M-1
00021680
00021690
00021700
00021710
00021720
00021730
00021740
00021750
00021760
00021770
00021780

```

C ASSUMING THE BLOCK TRIANGULAR MATRIX IS IN FACTORIZED FORM, THIS
ROUTINE COMPUTES THE SOLUTION FOR A PARTICULAR RIGHT SIDE

COMMON /PARM1/ M,NO,NO
COMMON /PARM2/ L,J,LZ

C SOLVE Y IN L * Y = F

DO 300 M=2,J
M1=M-1

```

00021790      DO 200 K=1,NP
00021800      SUM=0.
00021810      DO 100 KK=1,M
00021820      SUM=SUM-H(K,K,M)*F(K,K,M)
00021830      F(K,M)=F(K,M)+SUM
00021840      200 CONTINUE
00021850      300 CONTINUE
00021860
00021870      C SOLVE DU IN A * DU = F
00021880      C
00021890      CALL USOLVE(J)
00021900      C
00021910      DO 600 MM=2,NJ
00021920      C UPDATE RIGHT HAND SIDE
00021930      C
00021940      C
00021950      M1=J-MM+2
00021960      M2=M1-1
00021970      DO 500 K=1,M1
00021980      SUM=0.
00021990      DO 400 L=1,M
00022000      SUM=SUM-C(K,L,M)*H(L,M)
00022010      400 F(MP,K,M)=F(MP,K,M)+SUM
00022020      500 CONTINUE
00022030      C
00022040      CALL USOLVE(M)
00022050      C
00022060      600 CONTINUE
00022070      C
00022080      RETURN
00022090      END
00022100      SUBROUTINE USOLVE(M)
00022110      C ASSUMING A SCALAR MATRIA IS IN FACTORIZED FORM, THIS ROUTINE SOLVES
00022120      C THE SOLUTION FOR A PARTICULAR RIGHT HAND SIDE
00022130      C
00022140      COMMON /A/ A(6,6) /F/ F(6,1) /DU/ DU(6,1)
00022150      COMMON /MH/ MH(6,1) /MC/ MC(6,1) /MCH/ MCH(6,1)
00022160      COMMON /PARI/ N,MP,NJ
00022170      C SOLVE Y IN L * Y = F
00022180      C
00022190      DO 300 L=1,M
00022200      MH=L*H(L,M)
00022210      SUM=F(MH,M)
00022220      IF (L.FU.1) GO TO 200
00022230      LI=L-1
00022240      DO 100 K=1,LI
00022250      MCH=MC(K,M)
00022260      SUM=SUM-A(L,MH)*MCH+M(K,M)
00022270      200 DU(L,M)=SUM
00022280      MCH=MC(L,M)
00022290

```

```

00022300
00022310
00022320
00022330
00022340
00022350
00022360
00022370
00022380
00022390
00022400
00022410
00022420
00022430
00022440
00022450
00022460
00022470
00022480
00022490
00022500
00022510
00022520
00022530
00022540
00022550
00022560
00022570
00022580
00022590

IF (NCL(L,M)+EQU.1) INCL(M)=INCL(L,M)/A(NML,NLL,M)
300 CONTINUE
NML=NN(L,M)
NCL=NC(L,M)
F(N,M)=DU(N,M)/A(NRM,NCM,M)
C
C SOLVE DU IN A * DU = F
C
DO 500 LL=2,M
  L=N-LL+1
  L1=L-1
  SUM=DU(L,M)
  NML=NN(L,M)
  DO 400 K=L1,M
    NCL=NC(K,M)
    SUM=SUM-A(NML,NCL,M)*F(K,M)
  400 SUM=SUM-A(NML,NCL,M)*F(L,M)/SU4
  F(L,M)=SUM
  NCL=NC(L,M)
  IF (NCL(L,M)+EQU.0) F(L,M)=F(L,M)/A(NML,NCL,M)
500 CONTINUE
C
C REARRANGE COMPONENTS ONE TO MIXED PIVOTING
C
DO 600 L=1,M
  NCL=NC(L,M)
  600 DU(INCL,M)=F(L,M)
  ME FURTHER
  ENI

```

INITIAL DISTRIBUTION LISTCopies

7	Commander Naval Air Development Center Warminster, PA 18974 (2) Attn: Mr. C. Mazza (Code 6053) (5) Attn: Dr. K. Green (Code 6052)
3	Commander Naval Air Systems Command Jefferson Plaza Washington, DC 20361 Attn: Mr. R. Siewert (AIR-320D)
1	Commander Naval Weapons Center China Lake, CA 93555 Attn: Mr. B. Kowalsky (Code 3183)
1	Commander Naval Ship Research and Development Center Carderock, MD 20034 Attn: Mr. R. Murphy
1	Commander Naval Air Propulsion Test Center Trenton, NJ 98628 Attn: Mr. L. Palcza
1	Chief Office of Naval Research 1800 N. Quincy Street Arlington, VA 22217 Attn: Dr. R. Whitehead
1	Superintendent Naval Postgraduate School Monterey, CA 93940 Attn: Dr. M. Platzar
12	Defense Documentation Center Building #5 Cameron Station Alexandria, VA 22314
1	NASA-Ames Research Center 40 x 80 Wind Tunnel Moffett Field, CA 94035 Attn: Mr. D. Koenig (MS-247-1)

INITIAL DISTRIBUTION LIST (Continued)

Copies

1	Director Air Force Flight Dynamics Laboratory AFDL/XFM Wright-Patterson AFB Dayton, OH 45433 Attn: Dr. K. Nagaraja
1	Office of Secretary of Defense Director of Defense Research and Engineering OAD/ET Room 3D1089 Pentagon, Washington, DC 20301 Attn: Dr. B. Osborne, Jr.
3	Science Center Rockwell International Corporation 1049 Camino Dos Rios P.O. Box 1085 Thousand Oaks, CA 91360 Attn: Dr. N. Malmuth
1	Vought Corporation Advanced Technology Center, Inc. P.O. Box 6144 Dallas, TX Attn: Dr. C. Haight
1	General Dynamics Corporation Fort Worth, TX 76101 Attn: Mr. J. Clifton (MS-2892)
1	Rockwell International Columbus, OH 43216 Attn: Dr. P. Bevilacqua
1	Flight Dynamics Research Corporation 15809 Stagg Street Van Nuys, CA 91406 Attn: Dr. M. Alperin
1	Computational Mechanics 3601A Chapman Highway Knoxville, TN 37920 Attn: Dr. J. Baker