A Numerical Model for Two-Dimensional Hydrodynamics and Energy Transport

June 1973

**Prepared for the
Nuclear Regulatory Commission**

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Nuclear Regulatory Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

PACIFIC NORTHWEST LABORATORY
operated by
BATTELLE
for the
U.S. ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION
Under Contract E(45-1)-1830

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, Virginia 22151
Price: Printed Copy \$7.75; Microfiche \$2.25

3 3679 00062 3597

BNWL-1803
NRC-1

A NUMERICAL MODEL FOR TWO-DIMENSIONAL
HYDRODYNAMICS AND ENERGY TRANSPORT

by
D. S. Trent

June 1973

BATTELLE
Pacific Northwest Laboratories
Richland, Washington 99352

CONTENTS

LIST OF FIGURES	v
LIST OF TABLES	vi
1.0 INTRODUCTION	1
2.0 SUMMARY	5
3.0 THEORETICAL DEVELOPMENT	7
3.1 CONSERVATION EQUATIONS	8
3.2 TRANSPORT OF HEAT, SALINITY AND BUOYANCY	11
3.3 THE EQUATION OF STATE FOR SEA WATER	14
3.4 VORTICITY TRANSPORT	15
3.5 NONDIMENSIONAL FORM OF THE EQUATIONS OF MOTION	18
3.6 TWO-DIMENSIONAL TRANSPORT EQUATIONS IN RECTANGULAR COORDINATES	21
3.6.1 Two-Dimensional Transport Equations, x-z Geometry	21
3.6.2 Two-Dimensional Transport Equations in x-y Coordinates	26
3.6.3 Simplified Momentum Equation in x-y Coordinate	29
3.6.4 Vorticity - Stream Function Equation for Lateral Flow	30
3.6.5 Irrotational Flows	32
3.6.6 Flows with Frozen Vorticity Profiles	33
4.0 FINITE-DIFFERENCE MODELS	35
4.1 FLOW SYSTEMS	35
4.2 FINITE-DIFFERENCE GRID SYSTEM	36
4.3 UPSTREAM DIFFERENCING FOR CONSERVATIVE FORMS	39
4.4 DIFFERENCE EQUATIONS FOR x-z COORDINATES	42

CONTENTS (Cont'd)

4.5	DIFFERENCE EQUATIONS FOR x-y COORDINATES	48
4.6	BOUNDARY CONDITIONS	52
5.0	CODE DESCRIPTION AND ORGANIZATION	55
5.1	COMPUTATIONAL PROCEDURE	55
5.2	EXECUTIVE PROGRAM AND SUBROUTINE DESCRIPTION	56
6.0	COMPUTER CODE TEST RESULTS	63
6.1	LAMINAR FLOW PROOF TESTS	64
6.1.1	Channel Flow	64
6.1.2	Laminar Channel Flow with Sudden Enlargement	69
6.1.3	Flow to a Shoreline Intake	71
6.1.4	Computations Using Frozen Vorticity	79
REFERENCES		89
APPENDIX A		A-1

LIST OF FIGURES

4-1	Typical x-z Coordinate Flow Problem-- Intake System Withdrawal of Stratified Ambient Water	35
4-2	Typical x-y Coordinate Flow Problem-- Intake/Discharge System	36
4-3	Computational Grid for Difference Equations	37
4-4	Typical Finite-Difference Cell Illustrating Indices for ψ , Ω , Γ , U and V	38
4-5	Convective Γ Flux for an Infinitesimal x-z Volume Element	40
4-6	x-z Finite-Difference Cell with the Four Immediate Neighbor Cells	41
6-1	Laminar Channel Flow Proof Test Cases	65
6-2	Comparison of Centerline Velocity with Other Computational Techniques	66
6-3	Centerline Velocity for a Laminar Channel at Various Reynolds Number	66
6-4	Computed Channel Axis Centerline Velocity Distribution	67
6-5	Computed Velocity Profiles for Channel Flow at Various Downstream Distance, y/L	67
6-6	Comparison Between Computed and Theoretical Velocity Profile at Near-Fully Developed Laminar Channel Flow	69
6-7	Lateral Vorticity Distribution for Fully Established Flow	70
6-8	Centerline Axial Velocity for Various Reynolds Numbers	70
6-9	Lateral Velocity Profiles at Various Axial Locations	71
6-10	Streamlines for Inlet $Re = 16.25$ Sudden Expansion	72
6-11	Vorticity for Inlet $Re = 16.25$ Sudden Expansion .	72

LIST OF FIGURES (Cont'd)

6-12	Streamlines for Inlet $Re = 65$	73
6-13	Vorticity for Inlet $Re = 65$	73
6-14	Streamlines for Inlet $Re = 162.5$ Sudden Expansion	74
6-15	Vorticity for Inlet $Re = 162.5$ Sudden Expansion .	74
6-16	Schematic Diagram of River Shoreline Intake System for Cases I-1 Through I-4 and IB-1 Through IB-4	75
6-17	Flow Patterns for Intake Cases I-1 Through I-4 .	77
6-18	Flow Patterns for Intake Cases IB-1 Through IB-4.	78
6-19	Example Case 1: Small Cooling Pond Flow . .	80
6-20	Example Case 2: Intake/Discharge to Embayment .	80
6-21	Small Pond Streamlines (Potential Flow) . . .	82
6-22	Small Pond Streamlines (Computed Vorticity) . .	83
6-23	Small Pond Streamlines (Frozen Vorticity) . .	84
6-24	Embayment Streamlines	86
6-25	Embayment Streamlines (Frozen Vorticity) . .	87
6-26	Embayment Isotherms (Frozen Vorticity) . . .	88

LIST OF TABLES

3-1	Differential Equations Required for Velocity- Pressure and Vector Potential-Vorticity Methods in Two and Three Dimensions	22
6-1	Intake Flow Case Description	76

A NUMERICAL MODEL FOR TWO-DIMENSIONAL HYDRODYNAMICS AND ENERGY TRANSPORT

D. S. Trent

1.0 INTRODUCTION

This document describes the theoretical basis and computational procedure of the VECTRA computer program. VECTRA (Vorticity-Energy Code for Transport Analysis) is designed for applying numerical simulation to a broad range of intake/discharge flows in conjunction with power plant hydrological evaluation. The code computational procedure is based on finite-difference approximation of the vorticity-stream function partial differential equations which govern steady flow momentum transport of two-dimensional, incompressible, viscous fluids in conjunction with the transport of heat and other constituents.

To aid in power plant hydrological evaluation, there is obvious need for a computational tool which is reliable and site-independent, or at least can be applied to a broad range of intake/discharge configurations with minimum modification. Methods based on simplified theory are convenient to use but are most often inadequate. Reliable numerical simulation codes require considerable development effort and, when developed by the utility, are usually site-specific. It is impractical for regulatory agencies to launch a detailed numerical simulation development effort for each site evaluation; hence, the need for a reliable, adaptable, easy to use computational package is apparent. Development of the VECTRA code is a first step toward this goal. However, the ultimate objective of this development effort is not to produce a three-dimensional, time-dependent package for all-inclusive use although there is a definite need for such a tool.

Hydrodynamic and thermodynamic behavior of thermal effluents and natural flows are inherently three-dimensional and non-linear. They are further complicated by such phenomena as turbulence, bouyancy, crosscurrents and interchange of heat and momentum at the air-water interface. Because of these and other complications, the possibility of resolving quantities such as temperature and velocity in detail is rather remote without the aid of three-dimensional computational techniques. Recent developments in large core, high speed computer systems make the numerical simulation of three-dimensional hydrodynamic phenomena feasible⁽¹⁻³⁾ and set the stage for broad practical application. However, from a practical viewpoint, such capability is not presently state-of-the-art for generalized hydrologic assessment. In addition, the computer time required to obtain meaningful resolution of quantities is substantially greater than that for comparable two-dimensional methods.

A working tool for hydrologic assessments should be as simple as possible yet capable of satisfactorily resolving the phenomena in question. Within this framework many hydrologic analyses can be adequately performed through use of two-dimensional, steady or pseudo-steady flow techniques. In view of the inherent numerical complexities and economics associated with three-dimensional simulation, it seems prudent to attain the greatest practical proficiency in dealing with two-dimensional techniques and to exploit such methods to the greatest possible degree.

Plume flow may be qualitatively defined in terms of three flow regimes:

- near field (initial mixing zone)
- intermediate field (transition zone)
- far field (drift or passive flow zone).

Within the near field, flow dynamics and dispersion are dominated by initial momentum and plume-generated turbulence. The intermediate zone represents a region of transition from flow dominated by the discharge behavior to a regime dominated by ambient momentum, turbulence and atmospheric heat transfer (far field). Depending on shoreline geometry, outfall design, and ambient and tidal currents, all flow regimes may interact with one another.

Plume analysis state-of-the-art has advanced along the path of treating each of these flow regimes independently.⁽⁴⁾ With this approach, simplified theory such as similarity analysis may be applied to the initial mixing zone and simplistic diffusion theory to the far field.⁽⁵⁾ The assumption of independent flow regimes cannot be applied to flows with confining boundaries. Such assumptions can lead to gross errors and misleading results unless considerable information is already established concerning the flow dynamics.

Since different mechanisms dominate each flow regime, it is possible to properly model the entire flow field only through use of a single model that considers the complete equations of motion and energy transport. Full numerical methods for treating energy dispersion in the far field⁽⁶⁾ have been in use for some time, but only recently were attempts made to model the complete hydrodynamic equations with interacting heat transport in the near field.^(3,7)

The VECTRA code described in this report is designed to numerically simulate the complete flow field for general two-dimensional planar systems. Philosophy of the code development is to construct a base code which is readily adaptable to various site geometrics and can be incrementally improved and modified as additional capabilities are required.

Section 3 of this report discusses the theory used to arrive at the governing partial differential equation; Section 4 illustrates a simplified version of the difference equations used by VECTRA. A description of the code is given in Section 5. Section 6 illustrates results of code proof-testing for laminar channel flow as well as flow to a shoreline intake and presents generic examples using frozen vorticity. Discussion of the theory deals not only with capabilities of the base version but also with ultimate objectives of the code. A listing of the foundation version of the code is found in Appendix A.

2.0 SUMMARY

The mathematical theory and computational schemes for numerical simulation of power plant intake/discharge are developed and implemented in a computer program named VECTRA. This code is designed to be a site-adaptable computational tool which solves the complete equations of steady motion for two-dimensional turbulent flow and heat transport associated with the dynamic behavior of thermal effluents or intake flows.

The computation scheme used by the code is based on finite-difference approximation of the steady, viscous, incompressible flow equations cast in vorticity-stream function form. The code is capable of simulating the transport momentum and heat along with other constituents from the point of discharge to, and including, the far field. Lateral flow boundaries may have a general shape and multiple boundary flows (intake or discharge) may be specified. Because the complete steady flow hydrodynamic equations are solved, the code can predict phenomena beyond the capability of similarity solutions, such as recirculation and multiple discharge interaction.

The philosophy underlying development of the VECTRA code is construction of a foundation computational tool which can be readily modified and incrementally improved to include capabilities for analysis of a broad range of discharge/intake configurations. This report is concerned with development of this base code, although portions of the mathematical theory described are related to ultimate goals of the program. Features of the base code include:

- planar x-y coordinates
- arbitrary geometry
- steady flow hydrodynamic and heat transport
- uniform depth
- constant momentum and heat diffusivity coefficients
- no bottom or surface shear forces.

Proof-testing of the VECTRA code base version was carried out for laminar channel flow. Agreement between the VECTRA results and calculations presented in the literature for similar flow was found to be excellent. The computation scheme is adjudged to be sound and accurate based upon the results of these numerical experiments.

The base code will be incrementally improved as applications demand. Potential features include:

- pseudo-transient flow capability to accommodate uniformly fluctuating water levels
- approximate methods for the third dimension
- buoyancy coupling in x-z coordinate
- variable depth
- bottom drag and wind shear
- variable eddy transport coefficients
- improved model for plume-generated turbulence
- multilayered capability
- time-dependent heat transfer.

3.0 THEORETICAL DEVELOPMENT

In this section, the differential equations governing momentum and heat transport appropriate for three-dimensional flow in natural water bodies are developed. These equations may be derived by consideration of three physical laws:

- conservation of mass (continuity)
- Newton's second law (conservation of momentum)
- the first law of thermodynamics (conservation of energy).

In addition, an appropriate equation of state is required to relate water density in terms of temperature and, if appropriate, salinity.

The assumptions which are made at this stage of the development are:

- Flow is incompressible.
- Flow is turbulent.
- Reynolds stresses may be related to mean flow in terms of eddy diffusivities for momentum.
- Eddy diffusivities include molecular effects and, in the absence of turbulence, collapse to the appropriate value of molecular kinematic viscosity.
- Local change in density, ρ , is small compared to absolute value of density (i.e., the Boussinesq approximation, $|\Delta\rho|/\rho \ll 1$).⁽⁸⁾
- Turbulent heat and mass transfer coefficients are related to eddy diffusivities by an eddy Prandtl number or Schmidt number.

3.1 CONSERVATION EQUATIONS

The governing equations for three-dimensional flow are presented in Cartesian tensorial form by the following:

Continuity:

$$\frac{\partial u_i}{\partial x_i} = 0 \quad (3.1)$$

Momentum:

$$\frac{Du_i}{Dt} + 2e_{ijk}\Omega_j^* u_k = \frac{-\partial P^o}{\partial x_i} + \frac{\rho_\infty - \rho}{\rho_r} g_{i3} + \frac{\partial}{\partial x_j} \left(\varepsilon_{ij} \frac{\partial u_i}{\partial x_j} \right) \quad (a) \quad (3.2)$$

Energy (or other constituents):

$$\frac{D\Gamma}{Dt} = \frac{\partial}{\partial x_i} \left(\varepsilon_{\gamma i} \frac{\partial \Gamma}{\partial x_i} \right) + \dot{\phi}_\gamma \quad (3.3)$$

State:

$$\rho = f(T, S) \quad (3.4)$$

Symbols in the above equations are:

- u_i = velocity along the i^{th} space coordinate
- x_i = i^{th} space coordinate
- t = time
- g = gravitational constant
- ρ = density
- ρ_∞ = ambient reference density

(a) Einsteinian notation is used where repeated indices imply summation over all three index values ($i = 1, 2, \text{ and } 3$). The summation convention does not apply for underscored indices.

ρ_r = outfall reference density
 P = pressure
 Ω_j^* = j^{th} component of planetary angular velocity
 ϵ_{ij} = eddy diffusivity tensor
 Γ = transferable constituent (heat, concentration, etc.)
 $\epsilon_{\gamma i}$ = transport coefficient for constituent γ along i^{th} space coordinate
 $\dot{\phi}_\gamma$ = represents a source or sink of the constituent, γ .
 T = temperature
 S = salinity (PPT)
 δ_{i3} = Kronecker delta function
 e_{ijk} = permutation tensor

The operator D/Dt is the substantial derivative and has the usual meaning,

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + u_j \frac{\partial}{\partial x_j} .$$

Pressure is given in terms of the "deviatoric" pressure, P^0 , defined by

$$P^0 = \frac{P - P_\infty}{\rho_r} = \frac{P}{\rho_r} - g \int_{x_3}^0 \left(\frac{\rho_\infty}{\rho_r} \right) dx . \quad (3.5)$$

An orientation of the coordinate system is assumed such that x_3 is vertical upwards and x_1 and x_2 lie in the horizontal plane. Equation (3.5) does not account for water surface fluctuations which are a function of space and time.

The momentum equation (3.2) contains the unknowns u_1 , u_2 , u_3 , ρ and P^0 ; equations (3.3) and (3.4) may be used to solve for ρ and are coupled to the momentum equation in this fashion.

The vector equation (3.2) consists of only three scalar equations, whereas four are needed to solve for the remaining variables. A fourth equation may be written to solve for p^0 by taking the divergence of equation (3.2) to obtain

$$\begin{aligned} \frac{\partial}{\partial t} \left(\frac{\partial u_i}{\partial x_i} \right) + \left(\frac{\partial u_j}{\partial x_i} \right) \left(\frac{\partial u_i}{\partial x_j} \right) + u_j \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_i} \right) + 2e_{ijk}\Omega^*{}_j \frac{\partial u_k}{\partial x_i} \\ + \frac{\partial^2 p^0}{\partial x_i \partial x_i} + \frac{g}{\rho_r} \frac{\partial (\rho_\infty - \rho)}{\partial x_3} - \frac{\partial}{\partial x_i} \cdot \left[\frac{\partial}{\partial x_j} \left(\epsilon_{ij} \frac{\partial u_i}{\partial x_j} \right) \right] = 0 \quad (3.6) \end{aligned}$$

By continuity

$$\frac{\partial u_i}{\partial x_i} = 0 \quad ,$$

so that equation (3.6) may be reduced to

$$\begin{aligned} \frac{\partial^2 p^0}{\partial x_i \partial x_i} = - \left(\frac{\partial u_j}{\partial x_i} \right) \left(\frac{\partial u_i}{\partial x_j} \right) - 2e_{ijk}\Omega^*{}_j \frac{\partial u_k}{\partial x_i} + \frac{\partial B}{\partial x_3} \\ + \frac{\partial}{\partial x_j} \left(\frac{\partial \epsilon_{ij}}{\partial x_i} \cdot \frac{\partial u_i}{\partial x_j} \right) \quad , \quad (3.7) \end{aligned}$$

where B is the buoyancy parameter, defined by

$$B = \frac{g}{\rho_r} (\rho_\infty - \rho) \quad (3.8)$$

For the case where coriolis forces are neglected and quantities involving derivatives of eddy diffusivity are small compared to other terms, the pressure equation is

$$\frac{\partial^2 p^0}{\partial x_i \partial x_i} = - \left(\frac{\partial u_i}{\partial x_j} \right) \left(\frac{\partial u_j}{\partial x_i} \right) + \frac{\partial B}{\partial x_3} . \quad (3.9)$$

3.2 TRANSPORT OF HEAT, SALINITY AND BUOYANCY

Letting $\Gamma = T$ in equation (3.3), where T is temperature, the heat transport equation is

$$\frac{DT}{Dt} = \frac{\partial}{\partial x_j} \left(\epsilon_{Hj} \frac{\partial T}{\partial x_j} \right) + \dot{\Phi} . \quad (3.10)$$

In this case, $\dot{\Phi}$ corresponds to heat sources and sinks and/or viscous dissipation. Since none of these effects is significant in plume analysis, $\dot{\Phi}$ is neglected.^(a) The quantity ϵ_{Hj} is the turbulent heat diffusion coefficient and is assumed to include molecular effects. For salt transport, $\Gamma = S$ when S is salinity; hence,

$$\frac{DS}{Dt} = \frac{\partial}{\partial x_j} \left(\epsilon_{Sj} \frac{\partial S}{\partial x_j} \right) . \quad (3.11)$$

(a) The term $\dot{\Phi}$ is sometimes included to account for surface heat transfer.

Since salinity is a conservative quantity, $\dot{\phi}$ is omitted. The quantity ϵ_{sj} is the combined molecular and turbulent mass diffusion coefficient tensor.

The equations for heat and salinity transport are coupled to the equations of motion (3.2) through the buoyancy term $(\rho_\infty - \rho)/\rho_r$. Any Γ constituent which causes density variations to occur is coupled in the same fashion. Thus, it is not the absolute value of temperature, salinity, etc., which is important to the system dynamics, but the resulting density variations in a lateral plane caused by the transport of these quantities. For this reason, it is necessary only to deal with the transport of buoyancy in analyzing the dynamic behavior of the system. However, the equation for heat or salinity transport (in a system where differences of salinity and temperature are the causes of density variations) must be solved to establish the magnitude of temperature and salt content and to treat certain boundary conditions. Once the distribution of these quantities is known, salinity or temperature may be calculated from the equation of state (3.4).

A "density transport" equation may be derived by combining equations (3.10) and (3.11) after the independent variables T and S have been expressed in terms of ρ [assuming $\rho = \rho(S, T)$]. Hence,

$$\frac{D\rho}{Dt} = \frac{\partial}{\partial x_j} (\epsilon_{\rho j} \frac{\partial \rho}{\partial x_j}) + \epsilon_{\rho j} \frac{1}{\rho} \left(\frac{\partial \zeta}{\partial x_j} \right) \cdot \left(\frac{\partial \rho}{\partial x_j} \right), \quad (3.12)$$

where

$$\zeta = \frac{\partial T}{\partial \rho} + \frac{\partial S}{\partial \rho}. \quad (3.13)$$

A buoyancy parameter may be defined as

$$\Delta = \frac{\rho_r - \rho}{\rho_r - \rho_o},$$

and the appropriate transport equation for Δ is

$$\frac{D\Delta}{Dt} = \frac{\partial}{\partial x_j} (\epsilon_{\rho j} \frac{\partial \Delta}{\partial x_j}) + \epsilon_{\rho j} \left(\frac{\partial \zeta}{\partial x_j} \right) \cdot \left(\frac{\partial \Delta}{\partial x_j} \right). \quad (3.14)$$

If density is a linear function of both temperature and salinity, that is,

$$\rho - \rho_0 = -a (T - T_0) + b(S - S_0),$$

then

$$\zeta = \text{constant and } \partial \zeta / \partial x_j = 0.$$

Equations (3.12) and (3.14) become

$$\frac{D\rho}{Dt} = \frac{\partial}{\partial x_j} (\epsilon_{\rho j} \frac{\partial \rho}{\partial x_j}) \quad (3.15)$$

and

$$\frac{D\Delta}{Dt} = \frac{\partial}{\partial x_j} (\epsilon_{\rho j} \frac{\partial \Delta}{\partial x_j}), \quad (3.16)$$

respectively.

The quantity ζ is seen to be a correction term, which accounts for nonlinearities in the equation of state, $\rho = \rho(S, T)$. In the remainder of this report, Δ will be referred to as

$$\Delta = \frac{\rho_r - \rho}{\rho_r - \rho_0}, \quad \text{buoyancy parameter.}$$

For consideration of salinity transport, a quantity, Γ_s , is defined as

$$\Gamma_s = \frac{S_r - S}{S_r - S_0}, \quad (3.17)$$

where S_r and S_0 are the reference and source effluent salinities, respectively.

3.3 THE EQUATION OF STATE FOR SEA WATER

In the absence of other pollutants, the density of sea water is a function of pressure, temperature and salinity. Hence, the equation of state has the form

$$\rho = \rho(P, S, T) \quad . \quad (3.18)$$

Since only rather shallow water on an oceanographic scale is considered here, pressure effects are negligible; therefore,

$$\rho = \rho(S, T) \quad . \quad (3.19)$$

If other contaminants having concentration C are present,

$$\rho = \rho(S, T, C) \quad . \quad (3.20)$$

In this work, only equation (3.19) will be dealt with.

Since density variations are small in the sea, oceanographers deal with a modified density called sigma-t, defined as

$$\sigma_t = (\rho - 1) \times 1000,$$

which has cgs units and is a measure of the deviation in density from 1.0 gm/ml. The equation of state in general use by oceanographers may be found in U.S. Navy Hydrographic Office publication number 615⁽⁹⁾ or in Hill⁽¹⁰⁾ and has the form

$$\sigma_t = \Sigma_t + (\sigma_o + .1324) [1 - A_t + B_t (\sigma_o - .1324)] \quad (3.21)$$

where

$$\begin{aligned}\Sigma_t &= \frac{(T - 3.98)^2}{503.370} \times \frac{T + 283}{T + 67.26}, \\ A_t &= 10^{-3} T(4.7867 - 0.098185T + 0.0010843T^2), \\ B_t &= 10^{-6} T(18.030 - 0.8164T + 0.01667T^2), \\ \sigma_o &= -0.093 + 0.8149S - 0.000482S^2 + 0.0000068S^2.\end{aligned}$$

For fresh water, σ_o takes the value

$$\sigma_o = -0.1324 \quad (3.22)$$

and σ_t as defined by equation (3.21) collapses to Σ_t .

In the above equations, T is in degrees Celsius and salinity in parts per thousand. The quantity σ_o is the density of sea water (or fresh water) in sigma-t units at zero pressure and temperature.

3.4 VORTICITY TRANSPORT

In geophysical fluid dynamic problems, it is frequently very difficult to set realistic boundary conditions required for the solution of pressure equation (3.7). Pressure, and consequently associated boundary conditions, may be eliminated entirely from consideration by introducing the quantity, vorticity.

A brief summary of the general theory will be presented here for a homogeneous, isotropic turbulent flow field (i.e., $\epsilon_{ij} = \epsilon = \text{constant}$) in three dimension. Additional information concerning vorticity transport may be found in Batchelor.⁽¹¹⁾

As demonstrated by Batchelor, a conservative fluid velocity field may be defined by vector addition of an irrotational contribution, u_I , and a solenoidal contribution, u_S , or

$$\vec{u} = \vec{u}_I + \vec{u}_S. \quad (3.23)$$

The solenoidal part satisfies

$$\nabla \times \vec{u}_S = 0,$$

whereas the irrotational part satisfies

$$\nabla \cdot \vec{u}_I = 0.$$

In addition, the irrotational part of the velocity field, u_I , may be described in terms of a scalar potential ϕ so that

$$\vec{u}_I = \nabla \phi$$

and the solenoidal part in terms of a vector potential, $\vec{\psi}$, or

$$\vec{u}_S = \nabla \times \vec{\psi}.$$

Hence, the total velocity field is described by the vector and scalar potential as

$$\vec{u} = \nabla \phi + \nabla \times \vec{\psi}. \quad (3.24)$$

Vorticity, $\vec{\omega}$, is defined as

$$\vec{\omega} = \nabla \times \vec{u}.$$

Taking the curl of equation (3.24) and using the above expression for vorticity yields

$$\vec{\omega} = \nabla \times (\nabla \times \vec{\psi}). \quad (3.25)$$

However, by vector identity

$$\nabla \times (\nabla \times \vec{\psi}) = \nabla (\nabla \cdot \vec{\psi}) - \nabla^2 \vec{\psi},$$

which, for an incompressible flow, gives

$$\nabla^2 \vec{\psi} = -\vec{\omega} \quad (3.26)$$

since $\nabla \cdot \vec{\psi} = 0$.

Equation (3.26) is a Poisson-type partial differential equation relating the vector potential to the distribution of vorticity in the flow field.

The divergence of equation (3.24) gives

$$\nabla^2 \phi = \nabla \cdot \vec{u}$$

which, in view of the incompressibility condition

$$\nabla \cdot \vec{u} = 0,$$

reduces to

$$\nabla^2 \phi = 0, \tag{3.27}$$

which is Laplace's equation. Hence, the velocity field may be established through solution of equations (3.24), (3.26) and (3.27).

Hirasaki and Hellums⁽¹²⁾ demonstrated that equation (3.27) is extremely useful for prescribing inflow-outflow boundary conditions in a three-dimensional velocity field. They demonstrated that the flux boundary condition may be prescribed entirely by the scalar potential ϕ (velocity potential) or u_I . Hence, tangential components of $\vec{\psi} = 0$ and the normal derivative of $\vec{\psi} = 0$ may be set at all boundaries. The utility of this theory lies in the fact that vector potential boundary conditions may be intractable without consideration of the scalar potential ϕ . One exception is flow in a closed system where the boundary conditions on $\vec{\psi}$ remain as described above; since there is no boundary mass flux, $\nabla \phi = 0$ everywhere [cf. Aziz⁽¹⁾].

An equation for vorticity transport may be derived by taking the curl of the equation of motion (3.2) (after setting $\varepsilon_{ij} = \varepsilon$). This operation yields

$$\frac{D\vec{\omega}}{Dt} = (\vec{\Omega} + \vec{\omega}) \cdot \nabla \vec{u} + \nabla \times B\hat{e}_3 + \epsilon \nabla^2 \vec{\omega} \quad (3.28)$$

where $\hat{e}_3$ is a unit vector in the vertical direction.

The vorticity transport equation is simplified appreciably by assuming a homogeneous, isotropic turbulence field which eliminates numerous terms concerning the gradient of ϵ_{ij} .^(a) The two-dimensional counterpart to equation (3.28) is

$$\frac{D\vec{\omega}_3}{Dt} = \nabla \times B\hat{e}_3 + \epsilon \nabla^2 \vec{\omega}_3, \quad (3.29)$$

where one coordinate is vertical (x_3) and the other lies in the lateral plane.

3.5 NONDIMENSIONAL FORM OF THE EQUATIONS OF MOTION

A nondimensional formulation of the equations of motion permits investigation of the magnitude of the various forces exerted on a fluid element in terms of similarity parameters. The importance of the various parameters may then be analyzed on an order-of-magnitude basis and the results used to justify simplification of the governing equations under certain flow conditions. To this end, the following dimensionless variables are defined:

$$\begin{aligned} U_i &= u_i/v_o, \\ p^* &= p^o \rho_r / \Delta P_o, \\ \Omega_j^{**} &= 2\Omega_j^* / f_o, \\ t^* &= tv_o / D, \\ X_i &= x_i / D, \\ \epsilon_{ij}^* &= \epsilon_{ij} / \epsilon_o. \end{aligned} \quad (3.30)$$

(a) These terms will be investigated in Section 3.6, which covers two-dimensional flow.

In the above,

v_o = reference velocity,

ΔP_o = reference dynamic pressure (may be taken as $\frac{1}{2} \rho_o v_o^2$),

f_o = characteristic coriolis parameter,

D = characteristic length,

ϵ_o = characteristic eddy diffusion coefficient for momentum
(may be set to $Cv_o D$, where C is a constant).

Substituting the equation set (3.30) into equation (3.18) yields

$$\begin{aligned} \frac{\partial U_i}{\partial t^*} + \frac{\partial U_j U_i}{\partial X_j} + e_{ijk} \left(\frac{f_o D}{v_o} \right) \Omega_j^{**} U_k = - \left(\frac{\Delta P_o}{\rho_r v_o^2} \right) \frac{\partial P^*}{\partial X_i} \\ + \left(\frac{\rho_\infty - \rho}{\rho_r} \right) \frac{gD}{v_o^2} \delta_{i3} + \left(\frac{\epsilon_o}{v_o D} \right) \frac{\partial}{\partial X_j} \left(\epsilon_{ij}^* \frac{\partial U_i}{\partial X_j} \right) \end{aligned} \quad (3.31)$$

The dimensionless groups in equation (3.31) are

$$\frac{v_o}{f_o D} = R_o, \text{ Rossby number (ratio of inertial forces to coriolis forces),}$$

$$\frac{\rho_r v_o^2}{\Delta P_o} = Eu, \text{ Euler number (ratio of inertial forces to pressure forces),}$$

$$\frac{v_o^2}{\left(\frac{\rho_\infty - \rho}{\rho_r} \right) gD} = F, \text{ densimetric Froude number (ratio of inertial forces to buoyant forces),}$$

$$\frac{v_o D}{\epsilon_o} = Re_T, \text{ turbulent Reynolds number (ratio of inertial forces to turbulent shear forces).}$$

In terms of the preceding similarity parameters, equation (3.31) becomes

$$\begin{aligned} \frac{\partial U_i}{\partial t^*} + \frac{\partial U_j U_i}{\partial X_j} + \frac{1}{Ro} e_{ijk} \Omega_j^{**} U_k = \\ - \frac{1}{Eu} \frac{\partial P^*}{\partial X_i} + \frac{1}{F} \delta_{i3} + \frac{1}{Re_T} \frac{\partial}{\partial X_j} \epsilon_{ij}^* \frac{\partial U_i}{\partial X_j}. \end{aligned} \quad (3.32)$$

Equation (3.32) represents a gross nondimensionalization. Ideally, each component of momentum should be treated separately and length scales which correspond to the particular coordinates should be used. However, for purposes here the form of equation (3.32) is sufficient.

At middle latitudes, the characteristic coriolis parameter, f_0 , is approximately equal to 10^{-4} . For a thermal plume and the scales of motion to be considered in rivers and smaller lakes, the coriolis term is sufficiently small to neglect because of the apparent size of the Rossby number. However, for simulating the flow in larger lakes or portions of the ocean, coriolis terms cannot be omitted. In considerations to follow, the equations of motion are dealt with in the general form of

$$\begin{aligned} \frac{\partial U_i}{\partial t^*} + \frac{\partial U_j U_i}{\partial X_j} = - \frac{1}{Eu} \frac{\partial P^*}{\partial X_i} + \frac{1}{F} \delta_{i3} \\ + \frac{1}{Re_T} \frac{\partial}{\partial X_j} \left(\epsilon_{ij}^* \frac{\partial U_i}{\partial X_j} \right) \end{aligned} \quad (3.33)$$

and dimensional variations of the same equation.

3.6 TWO-DIMENSIONAL TRANSPORT EQUATIONS IN RECTANGULAR COORDINATES

Ideally, the transport equations should be solved in three-space, since the nature of many flows is distinctly three-dimensional. However, computational requirements necessary to obtain proper resolution of desired quantities in three dimensions are prohibitive from a practical standpoint in view of present computer hardware and economics. Two-dimensional considerations demand significantly less computation time and computer capacity. Hopefully, computation economics will permit practical, three-dimensional engineering calculations in the near future, thus avoiding certain restrictions inherent with two-dimensional approximations. Table 3-1 summarizes general requirements for two- and three-dimensional forms of the velocity-pressure and vorticity-vector potential equations.

This study considers the two-dimensional governing equations in both x-z (lateral-vertical) and x-y (lateral-lateral) coordinates. In the x-z coordinate system, two-dimensional flows such as the density driven surface spread in the arm of a cooling lake or selective withdrawal of stratified ambient water may be studied. When using x-y coordinates (the lateral plane), the vertically integrated transport equations are employed. This set of equations cannot accommodate buoyant forces or predict phenomena such as bottom counter-currents, but the solution can give useful information concerning gross lateral flow patterns. A multi-leveled lateral transport model, which would yield more detail related to the vertical variation, can be constructed, but is not discussed in this study.

3.6.1 Two-Dimensional Transport Equations, x-z Geometry

The two-dimensional rectangular coordinate system to be considered is defined as a plane normal to the geopotential surface. The two coordinates are defined as x and z, where x lies in the

TABLE 3-1. Differential Equations Required for Velocity-Pressure and Vector Potential-Vorticity Methods in Two and Three Dimensions

	Velocity-Pressure		Vector Potential-Vorticity	
	Equation Set		Equation Set	
	3-Dim. <u>i=1,2,3</u>	2-Dim. <u>i=1,2</u>	3-Dim. <u>i=1,2,3</u>	2-Dim. <u>i=1,2</u>
U_1 Parabolic	X	X		
U_2 Parabolic	X	X		
U_3 Parabolic	X			
P^0 Elliptic	X	X		
Γ Parabolic (1 or more)	X	X	X	X
ω_1 Parabolic			X	
ω_2 Parabolic			X	
ω_3 Parabolic			X	X
ψ_1 Elliptic			X	
ψ_2 Elliptic			X	
ψ_3 Elliptic			X	X
Φ Elliptic			(X) ^(a)	
Total of Required Equations (minimum)	5	4	7(8)	3

(a) Used in case of open boundaries.

x_1 , x_2 plane with no particular orientation, and z is aligned with the vertical x_3 axis. Corresponding velocity components, u and v , are in the x and z directions, respectively.

The velocity-pressure and constituent transport equations are:

Continuity

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial z} = 0 \quad . \quad (3.34)$$

Momentum Transport (x-direction)

$$\frac{Du}{Dt} = -\frac{1}{\rho} \frac{\partial P^O}{\partial x} + \frac{\partial}{\partial x} \left(\epsilon_x \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial z} \left(\epsilon_z \frac{\partial u}{\partial z} \right) , \quad (3.35)$$

$$\frac{Dv}{Dt} = -\frac{1}{\rho} \frac{\partial P^O}{\partial x} + B + \frac{\partial}{\partial x} \left(\epsilon_x \frac{\partial v}{\partial x} \right) + \frac{\partial}{\partial z} \left(\epsilon_z \frac{\partial v}{\partial z} \right) . \quad (3.36)$$

In the above momentum transport equations, ϵ_x is the lateral eddy diffusivity coefficient and ϵ_z is the corresponding vertical value. The substantial derivative is given in two dimensions as

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + u \frac{\partial}{\partial x} + v \frac{\partial}{\partial z} \quad .$$

Constituent Transport

$$\frac{D\Gamma}{Dt} = \frac{\partial}{\partial x} \left(\epsilon_{\gamma x} \frac{\partial \Gamma}{\partial x} \right) + \frac{\partial}{\partial z} \left(\epsilon_{\gamma z} \frac{\partial \Gamma}{\partial z} \right) . \quad (3.37)$$

Equations for the transport of specific constituents such as Δ , T, S, etc. will be developed where appropriate.

Pressure

The appropriate pressure equation may be obtained from equation (3.7) by letting $i = 2, 3$ and $j = 2, 3$. Hence,

$$\begin{aligned} \nabla^2 P^O = & - \left\{ \left(\frac{\partial u}{\partial x} \right)^2 + 2 \left(\frac{\partial v}{\partial x} \right) \left(\frac{\partial u}{\partial z} \right) + \left(\frac{\partial v}{\partial z} \right)^2 \right\} + \frac{\partial B}{\partial z} \\ & + \frac{\partial}{\partial x} \left(\frac{\partial \epsilon_x}{\partial x} \cdot \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial z} \left(\frac{\partial \epsilon_z}{\partial x} \cdot \frac{\partial u}{\partial z} \right) \\ & + \frac{\partial}{\partial x} \left(\frac{\partial \epsilon_x}{\partial z} \cdot \frac{\partial v}{\partial x} \right) + \frac{\partial}{\partial z} \left(\frac{\partial \epsilon_z}{\partial z} \cdot \frac{\partial v}{\partial z} \right) , \quad (3.38) \end{aligned}$$

where

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial z^2}.$$

If turbulent contributions are neglected,

$$\nabla^2 p^0 = - \left\{ \left(\frac{\partial u}{\partial x} \right)^2 + 2 \left(\frac{\partial v}{\partial x} \right) \left(\frac{\partial u}{\partial z} \right) + \left(\frac{\partial v}{\partial z} \right)^2 \right\} + \frac{\partial B}{\partial z}. \quad (3.39)$$

Recall that, by equation (3.5),

$$p^0 = P/\rho_r - g \int_z^0 \rho_\infty/\rho_r dz.$$

The most notable pioneering work in obtaining numerical solution to the laminar form of the velocity-pressure equations just given was performed at the Los Alamos Scientific Laboratory by Welch and colleagues.⁽¹³⁾

The stream function and vorticity equations are described in the paragraphs which follow.

Stream Function

An expression for the stream function in x-z coordinates may be obtained by considering only the x_3 component of the vector equation (3.26), or

$$\nabla^2 \psi = -\omega, \quad (3.40)$$

where ψ (stream function) = ψ_3 and $\omega = \omega_3$.

Velocity

Velocity relationships are obtained by using only the ψ_3 component of equation (3.24),

$$\vec{u} = \nabla x (\psi \hat{e}_z), \quad (3.41)$$

which yields

$$u = - \frac{\partial \psi}{\partial z} \quad (3.42)$$

$$v = \frac{\partial \psi}{\partial x}. \quad (3.43)$$

Vorticity Transport

If the eddy diffusivity ϵ_{ij} is constant, equation (3.29) may be used to obtain

$$\frac{D\omega}{Dt} = - \frac{\partial B}{\partial x} + \epsilon \nabla^2 \omega \quad (3.44)$$

where again $\omega = \omega_3$.

In general, the two anisotropic, nonhomogeneous components ϵ_x and ϵ_z must be considered, involving numerous derivatives of ϵ_x and ϵ_z . The vorticity equations are derived by cross differentiating equations (3.35) and (3.36), then subtracting the latter result from the former to obtain

$$\begin{aligned} \frac{D\omega}{Dt} = & - \frac{\partial B}{\partial x} + \epsilon_x \frac{\partial^2 \omega}{\partial x^2} + \epsilon_z \frac{\partial^2 \omega}{\partial z^2} \\ & + \frac{\partial \epsilon_x}{\partial z} \cdot \frac{\partial^2 u}{\partial x^2} + \frac{\partial \epsilon_z}{\partial z} \cdot \frac{\partial^2 u}{\partial z^2} - \frac{\partial \epsilon_x}{\partial x} \cdot \frac{\partial^2 v}{\partial x^2} - \frac{\partial \epsilon_z}{\partial x} \cdot \frac{\partial^2 v}{\partial z^2} \\ & + \frac{\partial}{\partial z} \left(\frac{\partial \epsilon_x}{\partial x} \cdot \frac{\partial u}{\partial x} \right) + \frac{\partial \epsilon_z}{\partial z} \cdot \frac{\partial u}{\partial z} \\ & - \frac{\partial}{\partial x} \left(\frac{\partial \epsilon_x}{\partial x} \cdot \frac{\partial v}{\partial x} \right) + \frac{\partial \epsilon_z}{\partial z} \cdot \frac{\partial v}{\partial z} . \end{aligned} \quad (3.45)$$

If the structure of the turbulent field is homogeneous and isotropic, equation (3.45) simplifies to

$$\frac{D\omega}{Dt} = - \frac{\partial B}{\partial x} + \epsilon_x \frac{\partial^2 \omega}{\partial x^2} + \epsilon_z \frac{\partial^2 \omega}{\partial z^2} . \quad (3.46)$$

Stream function-vorticity transport solution methods were employed for a number of years by oceanographers in computing such geophysical phenomena as western boundary currents [e.g.,

the Kuro Shio and the Gulf Stream, cf. Neumann and Pierson⁽¹⁴⁾]. This technique became popular in engineering application only in the past few years, due in part to the recognition that the method is extremely well adapted to problems involving natural convection. Solutions to the laminar form of the stream function-vorticity equations given previously were carried out by a number of researchers.⁽¹⁵⁻¹⁸⁾ The most notable research work on the turbulent form of the equations is being conducted at the Imperial College by Spalding and colleagues.⁽¹⁹⁻²⁴⁾

3.6.2 Two-Dimensional Transport Equations in x-y Coordinates

The x-y coordinate system lies in the lateral plane with x parallel to the east-west direction. The governing equations apply to a column of water of depth H where

$$H(x,y,t) = h(x,y) + \xi(x,y,t). \quad (3.47)$$

The height of the water column is a function of space and time and is composed of two parts: h , the mean water depth, which is a function of the bottom topography alone; and ξ , the fluctuating component, which relates the shape of the water surface as a function of time. Leenderste^(25,26) and Pritchard in Ward⁽²⁷⁾ approached the governing equations for this type of flow by considering vertically integrated or vertically averaged quantities. Vertical averaging over the water column may be expressed as

$$\bar{q} = \frac{1}{h+\xi} \int_{-h}^{\xi} q(z) dz \quad (3.48)$$

where q indicates any space-dependent physical quantity. Application of equation (3.48) is an attempt to account for known vertical variations which cannot be resolved by use of the lateral governing equations. The quantity q may be expressed as

$$q(z) = \bar{q}[1 + f_q(z)] \quad (3.49)$$

where $f(z)$ is the deviation from the average value which must integrate to zero over the water column depth. The distribution relationship, $f_q(z)$, must be based on a theoretical profile or estimated empirically.

The appropriate conservation equations are continuity, conservation of momentum, and conservation of energy.

Continuity

The continuity equation for a vertical water column of height $h + \xi$ and lateral dimension Δx and Δy may be expressed in the limit ($\Delta x, \Delta y \rightarrow 0$) as

$$\frac{\partial \xi}{\partial t} + \frac{\partial H\bar{u}}{\partial x} + \frac{\partial H\bar{v}}{\partial y} = 0. \quad (3.50)$$

Conservation of Momentum

The governing equations for lateral momentum are fashioned after Leenderste's assumptions.⁽²⁵⁾ This method again utilizes vertically integrated quantities and, except for employment of the vertical distribution function, $f_q(z)$, basically involves the same assumptions as several other approaches which deal with long wave hydromechanics and related problems.

The equation set used by Leenderste is based on inviscid flow theory where inertia and coriolis effects are balanced against the hydraulic head, bottom drag and wind stress. Bottom drag and wind stress, the result of viscous shear, are incorporated into the inviscid momentum balance as body forces and computed by empirical relationships. These assumptions are apparently valid if the resulting equations are applied to appropriate flows such as large-scale motion in shallow seas, estuaries and bays.

With thermal plumes, small-scale motions are most frequently of interest where viscous effects are important in computing such

phenomena as re-entrainment and circulation patterns. Thus lateral eddy momentum diffusion is important and must be incorporated into the lateral equations of motion. Using the velocity distribution functions

$$u = \bar{u} \{1 + f_u(z)\}$$

$$v = \bar{v} \{1 + f_v(z)\}$$

and incorporating lateral viscous terms, the equations of motion are expressed as

x-direction:

$$\begin{aligned} \frac{\partial H\bar{u}}{\partial t} + \frac{\partial}{\partial x} [\bar{u} \cdot \bar{u} <1 + f_u^2(z)>] + \frac{\partial}{\partial y} [\bar{u} \cdot \bar{v} <1 + f_u(z) \cdot f_v(z)>] \\ - fH\bar{v} = -gH \frac{\partial \xi}{\partial x} + \frac{\partial}{\partial x} [H\bar{\epsilon}_x \frac{\partial \bar{u}}{\partial x}] + \frac{\partial}{\partial y} [H\bar{\epsilon}_x \frac{\partial \bar{u}}{\partial y}] - \frac{1}{\rho} (\tau_x^s - \tau_x^b) \end{aligned} \quad (3.51)$$

y-direction:

$$\begin{aligned} \frac{\partial H\bar{v}}{\partial t} + \frac{\partial}{\partial x} [\bar{v} \cdot \bar{u} <1 + f_u(z) \cdot f_v(z)>] + \frac{\partial}{\partial y} [\bar{v} \cdot \bar{v} <1 + f_v^2(z)>] \\ + fH\bar{u} = -gH \frac{\partial \xi}{\partial y} + \frac{\partial}{\partial x} [H\bar{\epsilon}_x \frac{\partial \bar{v}}{\partial x}] + \frac{\partial}{\partial y} [H\bar{\epsilon}_y \frac{\partial \bar{v}}{\partial y}] - \frac{1}{\rho} (\tau_y^s - \tau_y^b) \end{aligned} \quad (3.52)$$

where the notation

$$< > = \int_{-h}^{\xi} () dz .$$

In the preceding equations, the assumption is made that turbulent shear may be related to the mean vertically integrated motion by the mean effective eddy diffusivity coefficients, $\bar{\epsilon}_x$ and $\bar{\epsilon}_y$. Although the effective diffusivities vary over the

depth, no distribution function is used in this connection since such a procedure would only complicate the analysis by attempting to refine the use of the quantities ϵ_x and ϵ_y which are at best poorly understood. Note that vertical integration of the cross products of the distribution functions for velocity yields non-zero results.

Conservation of Energy (or Other Constituents)

Assuming that a distribution function for constituents may be expressed by

$$\Gamma(z) = \bar{\gamma} \{1 + f_\gamma(z)\},$$

the vertically integrated transport equations may be written as

$$\begin{aligned} \frac{\partial H\bar{\gamma}}{\partial t} + \frac{\partial}{\partial x} [<1 + f_u(z) \cdot f_\gamma(z)> \bar{\gamma}] + \frac{\partial}{\partial y} [<1 + f_v(z) \cdot f_\gamma(z)> \bar{\gamma}] \\ + \dot{\gamma} = \frac{\partial}{\partial x} [H\bar{\epsilon}_{\gamma x} \frac{\partial \bar{\gamma}}{\partial x}] + \frac{\partial}{\partial y} [H\bar{\epsilon}_{\gamma y} \frac{\partial \bar{\gamma}}{\partial y}] \end{aligned} \quad (3.53)$$

where $\dot{\gamma}$ is the production or dissipation rate of the constituent (surface heat flux in the case of energy) and $\bar{\epsilon}_\gamma$ is the effective lateral eddy diffusion coefficient for $\bar{\gamma}$.

3.6.3 Simplified Momentum Equation in x-y Coordinate

The equation of motion may be simplified by considering the products of the distribution function as small compared to unity. That is,

$$1 \gg f_q(z) \cdot f_q(z).$$

Using this approximation and the continuity equation (3.50), the momentum equations may be written as

x-direction:

$$\begin{aligned} \frac{\partial \bar{u}}{\partial t} + \bar{u} \frac{\partial \bar{u}}{\partial x} + \bar{v} \frac{\partial \bar{u}}{\partial y} - f\bar{v} = & -g \frac{\partial \xi}{\partial x} + \frac{1}{H} \frac{\partial}{\partial x} [H\bar{\epsilon}_x \frac{\partial \bar{u}}{\partial x}] \\ & + \frac{\partial}{\partial y} [H\bar{\epsilon}_y \frac{\partial \bar{u}}{\partial y}] - (\tau_x^{*s} - \tau_x^{*b}) \end{aligned} \quad (3.54)$$

y-direction:

$$\begin{aligned} \frac{\partial \bar{v}}{\partial t} + \bar{u} \frac{\partial \bar{v}}{\partial x} + \bar{v} \frac{\partial \bar{v}}{\partial y} + f\bar{u} = & -g \frac{\partial \xi}{\partial y} + \frac{1}{H} \left\{ \frac{\partial}{\partial x} [H\bar{\epsilon}_y \frac{\partial \bar{v}}{\partial x}] \right. \\ & \left. + \frac{\partial}{\partial y} [H\bar{\epsilon}_y \frac{\partial \bar{v}}{\partial y}] \right\} - (\tau_y^{*s} - \tau_y^{*b}) . \end{aligned} \quad (3.55)$$

Further simplification is possible if H is constant or $\partial H/\partial x$ and $\partial H/\partial y$ are small compared to H , and the effective diffusivities are constant in space. With these restrictions, the momentum equations are simplified to

x-direction:

$$\frac{\partial \bar{u}}{\partial t} + \bar{u} \frac{\partial \bar{u}}{\partial x} + \bar{v} \frac{\partial \bar{u}}{\partial y} - f\bar{v} = -g \frac{\partial \xi}{\partial x} + \bar{\epsilon}_\ell \nabla^2 \bar{u} - (\tau_x^{*s} - \tau_x^{*b}) \quad (3.56)$$

y-direction:

$$\frac{\partial \bar{v}}{\partial t} + \bar{u} \frac{\partial \bar{v}}{\partial x} + \bar{v} \frac{\partial \bar{v}}{\partial y} + f\bar{u} = -g \frac{\partial \xi}{\partial y} + \bar{\epsilon}_\ell \nabla^2 \bar{v} - (\tau_y^{*s} - \tau_y^{*b}) . \quad (3.57)$$

3.6.4 Vorticity - Stream Function Equation for Lateral Flow

Consider vertically integrated vorticity to be defined by

$$\bar{\omega} = \frac{\partial \bar{u}}{\partial y} - \frac{\partial \bar{v}}{\partial x} . \quad (3.58)$$

Stream function $\bar{\psi}$ and scalar potential $\bar{\phi}$ are defined by

$$\bar{u}_H = - \frac{\partial \bar{\psi}}{\partial y} + \frac{\partial \bar{\phi}}{\partial x} \quad (3.59)$$

and

$$\bar{v}_H = \frac{\partial \bar{\psi}}{\partial x} + \frac{\partial \bar{\phi}}{\partial y} \quad (3.60)$$

Substituting equations (3.59) and (3.60) into equation (3.58) yields

$$\frac{\partial^2 \bar{\psi}}{\partial x^2} + \frac{\partial^2 \bar{\psi}}{\partial y^2} - \frac{1}{H} \left\{ \frac{\partial H}{\partial y} \left(\frac{\partial \bar{\psi}}{\partial y} - \frac{\partial \bar{\phi}}{\partial x} \right) + \frac{\partial H}{\partial x} \left(\frac{\partial \bar{\psi}}{\partial x} + \frac{\partial \bar{\phi}}{\partial y} \right) \right\} = - \bar{\omega}_H \quad (3.61)$$

Substituting equations (3.59) and (3.60) into the continuity equation (3.50) results in

$$\frac{\partial \xi}{\partial t} + \frac{\partial}{\partial x} \left(- \frac{\partial \bar{\psi}}{\partial y} + \frac{\partial \bar{\phi}}{\partial x} \right) + \frac{\partial}{\partial y} \left(\frac{\partial \bar{\psi}}{\partial x} + \frac{\partial \bar{\phi}}{\partial y} \right) = 0$$

which reduces to

$$\frac{\partial^2 \bar{\phi}}{\partial x^2} + \frac{\partial^2 \bar{\phi}}{\partial y^2} = - \frac{\partial \xi}{\partial t} \quad (3.62)$$

Mean velocities may be obtained by solving for $\bar{\psi}(x,y,t)$ and $\bar{\phi}(x,y,t)$ for application in equations (3.59) and (3.60). However, the solutions for $\bar{\psi}$ and $\bar{\phi}$ require resolution of the generating functions $\bar{\omega}$ and $\partial \xi / \partial t$, respectively.

A transport equation for $\bar{\omega}$ may be obtained by cross-differentiating the x and y components of the momentum equations (3.54) and (3.55), respectively, followed by subtraction of the latter results from the former. To reduce the mathematical procedure, these operations are applied to the simplified equations (3.56) and (3.57) to obtain

$$\begin{aligned} \frac{\partial \bar{\omega}}{\partial t} + \frac{\partial \bar{u}\bar{\omega}}{\partial x} + \frac{\partial \bar{v}\bar{\omega}}{\partial y} - \nu \frac{\partial^2 \bar{\omega}}{\partial y^2} &= \epsilon_\ell \left(\frac{\partial^2 \bar{\omega}}{\partial x^2} + \frac{\partial^2 \bar{\omega}}{\partial y^2} \right) - \frac{\partial}{\partial y} (\tau_x^{*s} - \tau_x^{*b}) \\ &+ \frac{\partial}{\partial x} (\tau_y^{*s} - \tau_y^{*b}) \quad (3.63) \end{aligned}$$

Although operations leading to equation (3.63) eliminated terms involving the hydraulic gradient, $\xi(x,y,t)$ must be established in some fashion to solve equation (3.62). For thermal plume analysis, usual concern is given only to local flow fields and local changes that occur by virtue of intake/discharge flows. Thus, for solution to equation (3.62), a valid assumption is that $\xi = f(t)$ alone. That is, spatial variations of ξ over the modeled flow field can be ignored, since the region is small and the gradient of ξ is small compared to $|\xi|$ or $|\nabla \xi| \ll |\xi|$ at any particular time, t . Therefore, in a tidal field, $\xi(t)$ is a function of the tidal stage and, for the purposes of continuity, is assumed spatially constant. Since the analysis is confined to local regions, the planetary vorticity contribution, $v \frac{\partial f}{\partial y}$, may also be eliminated.

3.6.5 Irrotational Flows

Irrotational flow theory requires that fluid vorticity is zero everywhere (i.e., $\bar{\omega} \equiv 0$). The flow field may be resolved by solving

$$\frac{\partial^2 \bar{\psi}}{\partial x^2} + \frac{\partial^2 \bar{\psi}}{\partial y^2} - \frac{1}{H} \left\{ \frac{\partial H}{\partial y} \left(\frac{\partial \bar{\psi}}{\partial y} - \frac{\partial \bar{\phi}}{\partial x} \right) + \frac{\partial H}{\partial x} \left(\frac{\partial \bar{\psi}}{\partial x} + \frac{\partial \bar{\phi}}{\partial y} \right) \right\} = 0 \quad (3.64)$$

along with equation (3.62).

If time fluctuations of the water level are negligible, the stream function distribution may be established by solving

$$\frac{\partial^2 \bar{\psi}}{\partial x^2} + \frac{\partial^2 \bar{\psi}}{\partial y^2} - \frac{1}{H} \left\{ \frac{\partial H}{\partial y} \cdot \frac{\partial \bar{\psi}}{\partial y} + \frac{\partial H}{\partial x} \cdot \frac{\partial \bar{\psi}}{\partial x} \right\} = 0 \quad (3.65)$$

For constant depth, equation (3.65) reduces to

$$\frac{\partial^2 \bar{\psi}}{\partial x^2} + \frac{\partial^2 \bar{\psi}}{\partial y^2} = 0 \quad (3.66)$$

3.6.6 Flows with Frozen Vorticity Profiles

Solution to the hydrodynamic equations can be a very difficult task for complex flow fields. Thus, the researcher is usually forced to accept approximate methods to establish velocity fields based on simplified theory or field data. For instance, solution to equation (3.64), irrotational flow, is not difficult, even for extension to three dimensions. However, the theory may be oversimplified to the point where the resulting velocity field is not realistic. A case in point is river flow where velocity profiles are important to the solution of the energy transport equation. For this problem, irrotational flow theory always leads to a more or less uniform cross-stream velocity distribution.

The other alternative is the use of field data. Such information usually consists of velocity estimates at various points in the fluid stream, but with insufficient data to use in the numerical energy equation without further resolution. Obtaining the proper velocity resolution for practical use requires that grid point extrapolations be made to perhaps 90-95% of the flow field. The extrapolated information must satisfy conservation of mass at each grid point (which may be a difficult task) or errors will result in the solution to the energy equation.

An alternative to the extrapolation procedure, ensuring satisfactory mass conservation, is to approximate the fluid vorticity at each point and solve

$$\frac{\partial^2 \bar{\psi}}{\partial x^2} + \frac{\partial^2 \bar{\psi}}{\partial y^2} - \frac{1}{H} \left\{ \frac{\partial H}{\partial y} \left(\frac{\partial \bar{\psi}}{\partial y} - \frac{\partial \bar{\phi}}{\partial x} \right) + \frac{\partial H}{\partial x} \left(\frac{\partial \bar{\psi}}{\partial x} - \frac{\partial \bar{\phi}}{\partial y} \right) \right\} = - \tilde{\omega} H \quad (3.67)$$

where $\tilde{\omega}$ is the estimated vorticity distribution based on field data. In this equation, $\tilde{\omega}$ tends to preserve velocity distributions so that by freezing the vorticity, velocity profiles are forced into the flow field. Although profiles established from data

may be approximate, the advantage of this procedure is that a realistic velocity field, conservative and directly usable in the energy analysis, is obtained.

To demonstrate this point, consideration can be made for fully developed laminar flow in a narrow rectangular channel of constant but great depth. The velocity profile is known to be parabolic across the channel everywhere along its length. Irrotational flow theory is used to solve for the flow field, so that

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = 0 \quad (3.68)$$

using a parabolic velocity profile as the upstream boundary condition. Solution shows that ψ very quickly assumes a linear cross-channel distribution which yields a uniform cross-channel profile. Next,

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = - \tilde{\omega} \quad (3.69)$$

where $\tilde{\omega}$ is established from parabolic velocity distribution. The solution will result in preservation of the parabolic distributed velocity. Further examples are given in Section 6.1.4.

4.0 FINITE-DIFFERENCE MODELS

In this section, two-dimensional finite-difference models are developed for the governing momentum and energy transport equations. These models, developed for steady and pseudo-steady turbulent flow in both x-z and x-y coordinates utilizing the vector potential-vorticity approach, form the basis of the computer code described in Section 5.

4.1 FLOW SYSTEMS

Typical flow systems to be modeled are schematically illustrated in Figures 4-1 and 4-2 for x-z and x-y coordinates, respectively. Figure 4-1 illustrates the flow of stratified ambient water to a submerged intake. The finite-differences model is created to accommodate a variety of configurations of these types, including two-dimensional discharges, intake skimmers and buoyant spread in general. Density variations caused by salinity and/or temperature distribution are to be considered. A general lateral flow situation is shown in Figure 4-2. The

FIGURE 4-1. Typical x-z Coordinate Flow Problem--
Intake System Withdrawal of Stratified
Ambient Water

FIGURE 4-2. Typical x-y Coordinate Flow Problem--
Intake/Discharge System

computer model for this case will include the effects of effluent momentum and recirculation, and will accommodate surface heat transfer and changing water level.

4.2 FINITE-DIFFERENCE GRID SYSTEM

The finite-difference grid layout in either x-y or x-z coordinates consists of two grid systems. One grid is used to calculate the stream function, ψ , which provides information to compute velocity components U and V . This system coincides with the physical boundaries and is illustrated by the wider lines on Figure 4-3. The stream function is calculated at the interior intersection points designated by the solid round symbols. Solid box symbols represent boundary points.

Velocities are not calculated at these points. U components are computed at vertical (or z coordinate) midpoints designated by open circle symbols. The V components are computed at horizontal (or x coordinate) midpoints and designated by open box symbols. Thus, the stream function grid layout defines a system of cells with the stream function, ψ , computed at each corner point (or set by boundary conditions) and velocities defined at the center of the cell face (see Figure 4-4).

● - Ψ; X - L, S; ○ - U; □ - V

■ - Ψ ; $\diamond$ - Δ , $\square$

FIGURE 4-3. Computational Grid for Difference Equations

FIGURE 4-4. Typical Finite-Difference Cell Illustrating Indices for ψ , Ω , Γ , U and V

The second grid system, used to calculate vorticity, Ω , and the buoyancy parameter, Δ , is illustrated in Figures 4-3 and 4-4 by the narrow lines. This layout completely overlaps the grid (and physical system) with interior intersection points centered in the cells defined by the ψ grid system. These interior grid points are indicated by crosses with boundary values at cross-and-box points.

The staggered grid system is used for computational convenience in treating boundary conditions and to permit convective transport terms to be evaluated at cell faces.

In Figure 4-3, the ψ grid system is sized by NJ and NK grid points in the x direction and vertical (or y) direction, respectively. The Ω, Δ system has size NJ + 1 and NK + 1 in the respective directions. Points on the ψ grid are indicated by j, k, whereas points on the Ω, Δ grid are indicated by p, q. In Figure 4-3, Z_h defines the water surface for x-z coordinates. Grid spacing in either coordinate system may be variable. Figure 4-4 also illustrates indices, computed quantities, cell size and radial distances for a typical interior cell.

4.3 UPSTREAM DIFFERENCING FOR CONSERVATIVE FORMS

The finite-difference methods employed in this work are based on central differencing techniques except for the transport equation convection terms which have the conservative form $\partial(U_j \Gamma) / \partial X_j$. The conservative form is a direct result of a Γ balance in terms of infinitesimal quantities and has the correct form for proper conservation of a transported quantity in numerical analysis.

Convective balance of Γ in x, z coordinates is illustrated in Figure 4-5.

The steady flow convective balance equation for volume element p is given by

$$\begin{aligned} \int_{A_T} \Gamma (\vec{v} \cdot \hat{n}) dA &= \int_{A_1} \Gamma (\vec{v} \cdot \hat{n}) dA + \int_{A_2} \Gamma (\vec{v} \cdot \hat{n}) dA \\ &+ \int_{A_3} \Gamma (\vec{v} \cdot \hat{n}) dA + \int_{A_4} \Gamma (\vec{v} \cdot \hat{n}) dA = 0. \end{aligned} \quad (4.1)$$

FIGURE 4-5. Convective Γ Flux for an Infinitesimal x-z Volume Element

In equation (4.1) and Figure 4-5, A_1 , A_2 , etc., are element areas corresponding to side 1, 2, etc., and $\hat{n}$ is a unit normal vector, with outward being the positive sense and inward, the negative. Like directional sense is used for the boundary velocity vector $\vec{v}$.

The grid system shown in Figure 4-6 has constant Δx and Δz , and velocities u and v are specified at the cell face, whereas Γ is cell-centered at point p . For the donor cell difference scheme based on equation (4.1), the value of Γ at the upstream neighbor is convected into cell p , and the value of Γ at point p is convected out of cell p . Thus, the value of Γ to be used in equation (4.1) is given by

$$\Gamma = \begin{cases} \Gamma_p, & \text{for } |\vec{v} \cdot \hat{n}| = \vec{v} \cdot \hat{n} \\ \text{value at upstream neighbor} & \text{for } |\vec{v} \cdot \hat{n}| \neq \vec{v} \cdot \hat{n}. \end{cases} \quad (4.2)$$

FIGURE 4-6. x-z Finite-Difference Cell with the Four Immediate Neighbor Cells

Unlike typical difference schemes, equation (4.1) provides flexibility of convecting into or out of any cell face. For the infinitesimal volume, the convective balance equation is

$$\begin{aligned} \Delta z \Gamma(\vec{v} \cdot \hat{n}) \Big|_{x-\frac{\Delta x}{2}} + \Delta z \Gamma(\vec{v} \cdot \hat{n}) \Big|_{x+\frac{\Delta x}{2}} + \Delta x \Gamma(\vec{v} \cdot \hat{n}) \Big|_{z-\frac{\Delta z}{2}} \\ + \Delta x \Gamma(\vec{v} \cdot \hat{n}) \Big|_{z+\frac{\Delta z}{2}} = 0. \end{aligned}$$

Dividing by volume ($\Delta x \Delta z$) having unit depth yields

$$\frac{\Gamma(\vec{v} \cdot \hat{n}) \Big|_{x-\frac{\Delta x}{2}}}{\Delta x} + \frac{\Gamma(\vec{v} \cdot \hat{n}) \Big|_{x+\frac{\Delta x}{2}}}{\Delta x} + \frac{\Gamma(\vec{v} \cdot \hat{n}) \Big|_{z-\frac{\Delta z}{2}}}{\Delta z} + \frac{\Gamma(\vec{v} \cdot \hat{n}) \Big|_{z+\frac{\Delta z}{2}}}{\Delta z} = 0.$$

(4.3)

Using the ideas expressed by equation (4.2), equation (4.3) is expressed in finite-difference form as

$$\begin{aligned}
& \frac{1}{2} \left\{ \frac{\Gamma_p (|u_{j-1/2}| - u_{j-1/2}) - \Gamma_{j-1} (|u_{j-1/2}| + u_{j-1/2})}{\Delta x} \right. \\
& + \frac{1}{2} \left\{ \frac{\Gamma_p (|u_{j+1/2}| + u_{j+1/2}) - \Gamma_{j+1} (|u_{j+1/2}| - u_{j+1/2})}{\Delta x} \right. \\
& + \frac{1}{2} \left\{ \frac{\Gamma_p (|v_{k-1/2}| - v_{k-1/2}) - \Gamma_{k-1} (|v_{k-1/2}| + v_{k-1/2})}{\Delta z} \right. \\
& + \left. \frac{1}{2} \left\{ \frac{\Gamma_p (|v_{k+1/2}| + v_{k+1/2}) - \Gamma_{k+1} (|v_{k+1/2}| - v_{k+1/2})}{\Delta z} \right\} \right. \\
& \Leftrightarrow \left. \frac{\partial (u\Gamma)}{\partial x} + \frac{\partial (v\Gamma)}{\partial z} \right|_p. \tag{4.4}
\end{aligned}$$

The above convective-difference form is used throughout this work.

4.4 DIFFERENCE EQUATIONS FOR x-z COORDINATES

Finite-difference models for the x-z coordinate system are based on equations (3.37), (3.40) and (3.46), along with the auxiliary equations (3.42) and (3.43) needed for computing velocity components. In this discussion, only the steady flow condition will be considered with constant eddy transport coefficients. The governing partial differential equations are summarized here.

Stream Function

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega \tag{4.5}$$

Vorticity

$$\frac{\partial u \omega}{\partial x} + \frac{\partial v \omega}{\partial y} = - \frac{\partial B}{\partial x} + \epsilon_x \frac{\partial^2 \omega}{\partial x^2} + \epsilon_z \frac{\partial^2 \omega}{\partial z^2} \quad (4.6)$$

Buoyancy Parameter

$$\frac{\partial u \Delta}{\partial x} + \frac{\partial v \Delta}{\partial y} = \epsilon_{\rho x} \frac{\partial^2 \Delta}{\partial x^2} + \epsilon_{\rho z} \frac{\partial^2 \Delta}{\partial y^2} \quad (4.7)$$

Conservative forms are employed for the convective terms in transport equations (4.6) and (4.7).

Velocity

$$u = - \frac{\partial \psi}{\partial y} \quad (4.8)$$

$$v = \frac{\partial \psi}{\partial x} \quad (4.9)$$

The following dimensionless variables are defined:

$$X = x/L,$$

$$Y = y/L,$$

$$U = u/v_o,$$

$$V = v/v_o,$$

$$\Psi = \psi/(Lv_o),$$

$$\Omega = \omega L/v_o,$$

where v_o and L are reference velocity and length, respectively.

Dimensionless parameters are:

$$RE_x = \frac{v_o L}{\epsilon_x}, \text{ horizontal reference Reynolds number,}$$

$$RE_z = \frac{v_o L}{\epsilon_z}, \text{ vertical reference Reynolds number,}$$

$$PR_x = \frac{\epsilon_x}{k_x} , \text{ horizontal reference Prandtl number,}$$

$$PR_z = \frac{\epsilon_z}{k_z} , \text{ vertical reference Prandtl number,}$$

$$F_o = \frac{v_o^2}{\left(\frac{\rho_r - \rho_o}{\rho_o}\right)gL} , \text{ reference densimetric Froude number.}$$

Based on the preceding dimensionless variables and parameters, the governing equations are

- Stream Function:

$$\frac{\partial^2 \Psi}{\partial X^2} + \frac{\partial^2 \Psi}{\partial Z^2} = -\Omega . \quad (4.10)$$

- Vorticity:

$$\frac{\partial}{\partial X} (U\Omega) + \frac{\partial}{\partial Z} (V\Omega) = -\frac{1}{F_o} \frac{\partial \Delta}{\partial X} + \frac{1}{RE_x} \frac{\partial^2 \Omega}{\partial X^2} + \frac{1}{RE_z} \frac{\partial^2 \Omega}{\partial Z^2} \quad (4.11)$$

- Buoyancy Parameter:

$$\frac{\partial}{\partial X} (U\Delta) + \frac{\partial}{\partial Z} (V\Delta) = \frac{1}{RE_x PR_x} \frac{\partial^2 \Delta}{\partial X^2} + \frac{1}{RE_z PR_z} \frac{\partial^2 \Delta}{\partial Z^2} \quad (4.12)$$

- Velocity:

$$U = -\frac{\partial \Psi}{\partial Z} \quad (4.13)$$

$$V = \frac{\partial \Psi}{\partial X} . \quad (4.14)$$

The finite-difference representations of these equations are given below. For simplicity, the grid is uniform with spacing ΔX and ΔZ in the horizontal and vertical direction, respectively.

- Stream Function:

$$\begin{aligned} \psi_{j,k} = & \frac{\psi_{j-1,k} + \psi_{j+1,k}}{2\left(\frac{1}{\Delta X^2} + \frac{1}{\Delta Z^2}\right) \Delta X^2} + \frac{\psi_{j,k-1} + \psi_{j,k+1}}{2\left(\frac{1}{\Delta X^2} + \frac{1}{\Delta Z^2}\right) \Delta Z^2} \\ & + \frac{\bar{\Omega}_{j,k}}{2\left(\frac{1}{\Delta X^2} + \frac{1}{\Delta Z^2}\right)}. \end{aligned} \quad (4.15)$$

Vorticity, $\bar{\Omega}_{j,k}$, is the average value for the four surrounding cells (see Figure 4-4) and given as

$$\bar{\Omega}_{j,k} = \frac{1}{4} (\Omega_{p,q} + \Omega_{p,q+1} + \Omega_{p+1,q} + \Omega_{p+1,q+1}) \quad (4.16)$$

- Velocity:

$$U_{j,k} = -\frac{1}{\Delta Z} (\psi_{j,k} - \psi_{j,k-1}) \quad (4.17)$$

$$V_{j,k} = \frac{1}{\Delta X} (\psi_{j,k} - \psi_{j-1,k}) \quad (4.18)$$

- Vorticity:

$$\begin{aligned} & \left[\frac{1}{2\Delta X} \left(|U_{j,k}| + U_{j,k} + |U_{j-1,k}| - U_{j-1,k} \right) \right. \\ & + \frac{1}{2\Delta Z} \left(|V_{j,k}| + V_{j,k} + |V_{j,k-1}| - V_{j,k-1} \right) \\ & \left. + \frac{2}{RE_x \Delta X^2} + \frac{2}{RE_z \Delta Z^2} \right] \Omega_{p,q} \\ & = \left[\frac{1}{2\Delta X} \left(|U_{j-1,k}| + U_{j-1,k} \right) + \frac{1}{RE_x \Delta X^2} \right] \Omega_{p-1,q} \end{aligned}$$

$$\begin{aligned}
& + \left[\frac{1}{2\Delta X} \left(|U_{j,k}| - U_{j,k} \right) + \frac{1}{RE_X \Delta X^2} \right] \Omega_{p+1,q} \\
& + \left[\frac{1}{2\Delta Z} \left(|V_{j,k-1}| + V_{j,k-1} \right) + \frac{1}{RE_Z \Delta Z^2} \right] \Omega_{p,q-1} \\
& + \left[\frac{1}{2\Delta Z} \left(|V_{j,k}| - V_{j,k} \right) + \frac{1}{RE_Z \Delta Z^2} \right] \Omega_{p,q+1} \\
& - \frac{1}{2F_0 \Delta X} \left(\Delta_{p+1,q} - \Delta_{p-1,q} \right). \tag{4.19}
\end{aligned}$$

- Buoyancy parameter, Δ :

$$\begin{aligned}
& \left[\frac{1}{2\Delta X} \left(|U_{j,k}| + U_{j,k} + |U_{j-1,k}| - U_{j-1,k} \right) \right. \\
& + \frac{1}{2\Delta Z} \left(|V_{j,k}| + V_{j,k} + |V_{j,k-1}| - V_{j,k-1} \right) \\
& \left. + \frac{2}{RE_X PR_X \Delta X^2} + \frac{2}{RE_Z PR_Z \Delta Z^2} \right] \Delta_{p,q} \\
& = \left[\frac{1}{2\Delta X} \left(|U_{j-1,k}| + U_{j-1,k} \right) + \frac{1}{RE_X PR_X \Delta X^2} \right] \Delta_{p-1,q} \\
& + \left[\frac{1}{2\Delta X} \left(|U_{j,k}| - U_{j,k} \right) + \frac{1}{RE_X PR_X \Delta X^2} \right] \Delta_{p+1,q} \\
& + \left[\frac{1}{2\Delta Z} \left(|V_{j,k-1}| + V_{j,k-1} \right) + \frac{1}{RE_Z PR_Z \Delta Z^2} \right] \Delta_{p,q-1} \\
& + \left[\frac{1}{2\Delta Z} \left(|V_{j,k}| - V_{j,k} \right) + \frac{1}{RE_Z PR_Z \Delta Z^2} \right] \Delta_{p,q+1}. \tag{4.20}
\end{aligned}$$

In reference to Figure 4-1, boundary conditions are

- Free-surface ($Z=Z_h$):

$$\Psi = \text{constant} = \Psi_1$$

$$\Omega = 0 \text{ (free slip condition)}$$

$$\frac{\partial \Delta}{\partial Z} = 0 \text{ (adiabatic condition, for present work)}$$

- Plume or intake boundary: Discharge/intake velocity V_b and Δ are assumed known from data or empirical relationships.

$$\Psi = \Psi_1 + \int_0^Z V_b dz$$

$$\Omega = \frac{\partial U}{\partial Z} - \frac{\partial V}{\partial X}$$

$$\Delta = \Delta_b$$

- Bottom boundary:

$$\Psi = \Psi_1 + \int_0^{Z_h} V_b dz = \text{constant} = \Psi_3$$

$$\Omega = \frac{\partial U}{\partial Z} - \frac{\partial V}{\partial X} \text{ (no-slip condition)}$$

$$\frac{\partial \Delta}{\partial Z} = 0$$

- Inflow-outflow boundary ($X=X_\infty$):

$\frac{\partial \Psi}{\partial X} = \text{constant}$, that is, streamlines do not change slope as they cross this boundary.

$\Omega = 0$, flow into system is irrotational

$$\Delta = \frac{\rho_r - \rho_\infty}{\rho_r - \rho_o} .$$

4.5 DIFFERENCE EQUATIONS FOR x-y COORDINATES

The finite-difference models for lateral flow are based on equations (3.61), (3.63) and (3.53), along with the auxiliary equations (3.59) and (3.60) needed to compute velocities. As in the case of x-z coordinates, only steady flow, which is governed by the following dimensionless equations, is considered.

- Stream Function:

$$\frac{\partial^2 \psi}{\partial X^2} + \frac{\partial^2 \psi}{\partial Y^2} - \frac{1}{H^*} \left\{ \frac{\partial H^*}{\partial Y} \left(\frac{\partial \psi}{\partial Y} - \frac{\partial \phi}{\partial X} \right) + \frac{\partial H^*}{\partial X} \left(\frac{\partial \psi}{\partial X} + \frac{\partial \phi}{\partial Y} \right) \right\} = - \bar{\Omega} H^* \quad (4.21)$$

- Velocity Potential:

$$\frac{\partial^2 \phi}{\partial X^2} + \frac{\partial^2 \phi}{\partial Y^2} = - D \quad (4.22)$$

where D is flow divergence (source or sink in this case)

- Vorticity:

$$\frac{\partial U \Omega}{\partial X} + \frac{\partial V \Omega}{\partial Y} = \frac{1}{RE_\ell} \left(\frac{\partial^2 \Omega}{\partial X^2} + \frac{\partial^2 \Omega}{\partial Y^2} \right) + \frac{\partial \tau_{X^{**}b}}{\partial Y} + \frac{\partial \tau_{Y^{**}b}}{\partial X} \quad (4.23)$$

Comparing equation (4.23) to (3.63) reveals that both coriolis and wind stress are omitted. Also, complications involving changes of depth and variable transport coefficient are ignored in the lateral viscous terms.

- Energy:

$$\frac{\partial H^* U T}{\partial X} + \frac{\partial H^* V T}{\partial Y} + \dot{Q} = \frac{\partial}{\partial X} \left[\frac{H^*}{RE_x PR_x} \frac{\partial T}{\partial X} \right] + \frac{\partial}{\partial Y} \left[\frac{H^*}{RE_y PR_y} \frac{\partial T}{\partial Y} \right] \quad (4.24)$$

Reference to equation (3.53) shows that cross-products of the vertical distribution functions are ignored.

- Velocity:

$$U H^* = - \frac{\partial \psi}{\partial Y} + \frac{\partial \phi}{\partial X} \quad (4.25)$$

$$VH^* = \frac{\partial \Psi}{\partial X} + \frac{\partial \Phi}{\partial Y} . \quad (4.26)$$

In equations (4.21) through (4.26), dimensionless quantities are defined:

$$\begin{aligned} X &= x/L, \\ Y &= y/L, \\ H^* &= H/L, \\ U &= u/v_o, \\ V &= v/v_o, \\ \Psi &= \bar{\Psi}/(Lv_o), \\ \Omega &= \bar{\omega}L/v_o, \\ \Phi &= \bar{\phi}/(Lv_o), \\ \dot{Q} &= q/(\rho c_p v_o), \\ \tau^{**b} &= \tau^b/(H^* v_o^2). \end{aligned}$$

Dimensionless parameters are

$$RE_x = \frac{v_o L}{\epsilon_x}$$

$$RE_y = \frac{v_o L}{\epsilon_y}$$

$$PR_x = \frac{\epsilon_x}{k_x}$$

$$PR_y = \frac{\epsilon_y}{k_y}$$

The difference equations given below are based on a regular grid for simplicity, with spacing ΔX and ΔY .

- Stream Function:

$$\begin{aligned} \psi_{j,k} &= \frac{G}{\Delta X^2} (\psi_{j-1,k} + \psi_{j+1,k}) + \frac{G}{\Delta Y^2} (\psi_{j,k-1} + \psi_{j,k+1}) \\ &+ G \cdot f(\psi, \phi) + G \cdot (\bar{\Omega} H^*)_{j,k} \end{aligned} \quad (4.27)$$

where

$$\begin{aligned} G &= \frac{1}{2 \left(\frac{1}{\Delta X^2} + \frac{1}{\Delta Y^2} \right)} \\ \bar{\varepsilon}(\psi, \phi) &= \frac{1}{H_{j,k}^*} \left[\left(\frac{H_{j,k+1}^*}{2\Delta Y} - H_{j,k-1}^* \right) \left(\frac{\psi_{j,k+1}}{2\Delta Y} - \psi_{j,k-1} \right) \right. \\ &\quad \left. - \frac{\phi_{j+1,h} - \phi_{j-1,k}}{2\Delta X} \right] + \left(\frac{H_{j+1,k}^*}{2\Delta X} - H_{j-1,k}^* \right) \\ &\quad \left(\frac{\psi_{j+1,k} - \psi_{j-1,k}}{2\Delta X} + \frac{\phi_{j,k+1} - \phi_{j,k-1}}{2\Delta Y} \right) \end{aligned} ,$$

for constant H^* , $f = 0$.

Vorticity, $\bar{\Omega}_{j,k}$, is the average value for the four surrounding cells (see Figure 4-4) and is given as

$$\bar{\Omega}_{j,k} = 1/4 (\Omega_{p,q} + \Omega_{p,q+1} + \Omega_{p+1,q} + \Omega_{p+1,q+1}) \quad (4.28)$$

- Velocity:

$$(UH^*)_{j,k} = -\frac{1}{\Delta Y} (\psi_{j,k} - \psi_{j,k-1})$$

$$(VH^*)_{j,k} = \frac{1}{\Delta X} (\psi_{j,k} - \psi_{j-1,k}) \quad (4.29)$$

- Velocity Potential:

$$\begin{aligned} \phi_{j,k} &= \frac{G}{\Delta X^2} (\phi_{j-1,k} + \phi_{j+1,k}) + \frac{G}{\Delta Y^2} (\phi_{j,k-1} + \phi_{j,k+1}) \\ &+ G D_{jk} \end{aligned} \quad (4.30)$$

- Vorticity:

$$C_{\Omega p,q}^{\Omega} = C_1^{\Omega} p-1,q + C_2^{\Omega} p+1,q + C_3^{\Omega} p,q-1 + C_4^{\Omega} p,q+1 + f(\tau)_{p,q} \quad (4.31)$$

where

$$\begin{aligned} C &= \left[\frac{1}{2\Delta X} (|U_{j,k}| + U_{j,k} + |U_{j-1,k}| - U_{j-1,k}) \right. \\ &\quad + \frac{1}{2\Delta Y} (|V_{j,k}| + V_{j,k} + |V_{j,k-1}| - V_{j,k-1}) \\ &\quad \left. + \frac{2}{RE_{\ell} \Delta X^2} + \frac{2}{RE_{\ell} \Delta Y^2} \right] \\ C_1 &= \left[\frac{1}{2\Delta X} (|U_{j-1,k}| + U_{j-1,k}) + \frac{1}{RE_{\ell} \Delta X^2} \right] \\ C_2 &= \left[\frac{1}{2\Delta X} (|U_{j,k}| - U_{j,k}) + \frac{1}{RE_{\ell} \Delta X^2} \right] \\ C_3 &= \left[\frac{1}{2\Delta Y} (|V_{j,k-1}| + V_{j,k-1}) + \frac{1}{RE_{\ell} \Delta Y^2} \right] \\ C_4 &= \left[\frac{1}{2\Delta Y} (|V_{j,k}| - V_{j,k}) + \frac{1}{RE_{\ell} \Delta Y^2} \right] \\ f(\tau)_{p,q} &= \frac{1}{2\Delta Y} \left[\left(\frac{\tau_x^{*b}}{H^*} \right)_{p+1,q} - \left(\frac{\tau_x^{*b}}{H^*} \right)_{p-1,q} \right] \\ &\quad - \frac{1}{2\Delta X} \left[\left(\frac{\tau_y^{*b}}{H^*} \right)_{p,q+1} - \left(\frac{\tau_y^{*b}}{H^*} \right)_{p,q-1} \right] \end{aligned}$$

- Energy:

$$\begin{aligned} C_T^T p,q &= C_{T1}^T p-1,q + C_{T2}^T p+1,q + C_{T3}^T p,q-1 \\ &\quad + C_{T4}^T p,q+1 + C_{T5}^T f(T_{p,q}) \end{aligned} \quad (4.32)$$

where

$$C_T = \left[\frac{1}{2\Delta X} \left(|UH_{j,k}^*| + UH_{j,k}^* + |UH_{j-1,k}^*| - UH_{j-1,k}^* \right) \right. \\ \left. + \frac{1}{2\Delta Y} \left(|VH_{j,k}^*| + VH_{j,k}^* + |VH_{j,k-1}^*| - VH_{j,k-1}^* \right) \right. \\ \left. + \frac{2}{RE_\ell PR_\ell \Delta X^2} + \frac{2}{RE_\ell PR_\ell \Delta Y^2} \right]$$

$$C_{T1} = \left[\frac{1}{2\Delta X} \left(|UH_{j-1,k}^*| + UH_{j-1,k}^* \right) + \frac{1}{RE_\ell PR_\ell \Delta X^2} \right]$$

$$C_{T2} = \left[\frac{1}{2\Delta X} \left(|UH_{j,k}^*| - UH_{j,k}^* \right) + \frac{1}{RE_\ell PR_\ell \Delta X^2} \right]$$

$$C_{T3} = \left[\frac{1}{2\Delta Y} \left(|VH_{j,k-1}^*| + VH_{j,k-1}^* \right) + \frac{1}{RE_\ell PR_\ell \Delta Y^2} \right]$$

$$C_{T4} = \left[\frac{1}{2\Delta Y} \left(|VH_{j,k}^*| - VH_{j,k}^* \right) + \frac{1}{RE_\ell PR_\ell \Delta Y^2} \right]$$

- Constituents:

$$C_\Gamma \Gamma_{p,q} = C_{\Gamma 1} \Gamma_{p-1,q} + C_{\Gamma 2} \Gamma_{p+1,q} + C_{\Gamma 3} \Gamma_{p,q-1} + C_{\Gamma 4} \Gamma_{p,q+1}$$

(4.33)

where the C_Γ 's are identical to the C_T 's when the eddy Prandtl number, PR , is replaced by the eddy Schmidt number, SC .

4.6 BOUNDARY CONDITIONS

Boundary conditions for x-y coordinates must be more general with respect to geometry than for x-z coordinates. The three types of boundary conditions needed are:

- no-slip
- inflow-outflow (distribution specified)
- outflow (distribution not specified)

Let N and M be the directions normal and tangential to a physical boundary.

- No-slip

$$\Psi = \Psi_0 = \text{constant}$$

$$\text{That is, } \frac{\partial \Psi}{\partial M} = 0.$$

$$\frac{\partial \phi}{\partial N} = 0$$

$$V_M, V_N = 0.$$

$$\Omega = \frac{\partial U}{\partial Z} - \frac{\partial V}{\partial X}$$

Where the Ω boundary condition is always calculated at the center of the adjacent fluid node,

$$\frac{\partial T}{\partial N} = 0 \text{ (adiabatic condition).}$$

- In $\frac{\partial \Gamma}{\partial N} = 0$ flow-outflow (distribution specified),

$$\Psi = \Psi_c + \int_{M_1}^{M_2} (\pm) V_b(M) dM$$

where M_1 and M_2 are the positions which define the edges of the inflow or outflow stream, $V_b(M)$ is the boundary velocity distribution, and Ψ_c is the value of the stream function at the beginning of integration across the stream. The sign $(\pm)$ depends on the velocity component being integrated [see equations (4.25) and (4.26)].

$$\phi = \phi_b$$

$$\Omega = \frac{\partial U}{\partial Z} - \frac{\partial V}{\partial X}$$

$$T = T_b$$

$$\Gamma = \Gamma_b$$

- Outflow (distribution computed)

$$\frac{\partial \Psi}{\partial N} = \text{constant}$$

$$\frac{\partial \Phi}{\partial N} = \text{constant}$$

$$\frac{\partial \Omega}{\partial N} = \text{constant}$$

$$\frac{\partial T}{\partial N} = \text{constant}$$

$$\frac{\partial \Gamma}{\partial N} = \text{constant}$$

The outflow boundary conditions stipulate that the normal distributions of Ψ , Ω , T and Γ do not change slope across an outflow boundary.

5.0 CODE DESCRIPTION AND ORGANIZATION

The computer program described in this section is designed to obtain the solution of the difference equations (4.15), (4.17), (4.18), (4.19), and (4.20) for the quantities Ψ , U , V , Ω , and Δ , (or Γ), respectively. The program consists of subroutines and/or functions which are managed in part by an executive routine called "VECTRA" (Vorticity-Energy Code For Transport Analysis). The code has been adapted to both the Univac 1108 and Control Data Corporation 6600 (CYBER 74).

5.1 COMPUTATIONAL PROCEDURE

The task of the computer program is the simultaneous solution of one elliptic partial differential equation (4.15) for the stream function, ψ , and parabolic transport equations (4.19) and (4.20) for the vorticity, Ω , and the buoyancy parameter, Δ . Equations for U (4.17) and V (4.18) are auxiliary but are also solved simultaneously during iteration. For neutral buoyancy, the buoyancy parameter equation need not be solved simultaneously with the vorticity and stream function equation. The Gauss-Seidel iterative technique is used for all quantities defined by second order partial differential equations. Liebmann acceleration is employed with the alternatives of both under- and over-relaxation.

Assuming all boundary conditions are set and pertinent variables are initialized, the procedure for solving the buoyancy coupled equations follows the steps below.

1. Compute $\Delta_{p,q}$ and $\Gamma_{p,q}$ using equation (4.20) based on previously calculated values of $U_{j,k}$, $V_{j,k}$ and appropriate transport coefficients.
2. Compute $\Omega_{p,q}$ using equation (4.19) and the previously computed values of $U_{j,k}$, $V_{j,k}$, $\Delta_{p,q}$ and appropriate transport coefficients.

3. Update necessary boundary values for Δ , Γ , and Ω .
4. Use the newly computed values of Ω to compute the stream function distribution from equation (4.15). One or more iterations may be required to arrive at a satisfactory solution for Ψ . Compute a new velocity field $V_{j,k}$ and $U_{j,k}$ from the newly calculated Ψ distribution.
5. If the eddy transport terms are not constant, compute multipliers FR and FZ from new velocity field (not implemented in present code version).
6. Repeat Steps 1 through 5 until a preset convergence criterion is satisfied or a specific number of iterations has been completed.

5.2 EXECUTIVE PROGRAM AND SUBROUTINE DESCRIPTION

The computer code consists of an executive routine called VECTRA and 18 subroutines and/or functions. A list of the primary duties served by each follows.

VECTRA Executive Routine

- Reads case header and case set-up information.
- Reads alphanumeric data for line printer output array option, plot file options, and program control.
- Calls subroutines for data input, problem set-up and initialization, and program execution. The subroutines called in sequence are

INPUT

TYPE

READY

SEMBLE (for transport solution)

PLABAK

STREAM (for inviscid flow solution)

SSCOMP

- Performs other miscellaneous tasks such as clock initialization, presetting variables, etc.

Subroutines

Primary duties of subroutines are given below.

Subroutine INPUT

General data input routine:

- reads restart file if required
- reads remaining input data from cards
- converts portions of input data to appropriate quantities and units

INPUT is called once during execution.

Subroutine TYPE

Sets cell corner and cell center node types:

- sets type 0 and type 10 fluid cell corner nodes
- sets type 0 and type 10 fluid cell center nodes
- sets slip boundary bookkeeping indices

TYPE is called once during execution.

Subroutine READY

Problem set-up routine:

- sets all computed constants
- sets constant boundary conditions
- computes boundary coordinates for plotting

READY is called once during execution.

Subroutine SEMBLE

Computes intercell conductances for heat and constituent transport.

Subroutine PLABAK

General information and debug output:

- writes to line printer various computed and input supplied variables and the operation modes of current case
- writes to line printer constant arrays used in the difference equation computations
- writes to line printer cell-corner and cell-center node type arrangement
- writes debug arrays

PLABAK is called once during execution.

Subroutine STREAM (IT, NSKIP)

Solves for stream function, Ψ :

- computes the viscous or inviscid stream function by Gauss-Siedel iteration. When called, this subroutine iterates on Ψ "IT" times.
- Upon completion of "IT" iterations, the velocity components $U_{j,k}$ and $V_{j,k}$ are computed by the auxiliary equations (4.17) and (4.18).

Subroutine STREAM constitutes what is referred to in this report as the "inner iteration loop" (subroutine SSCOMP constitutes the "outer iteration loop") and is called at least once for each "outer iteration."

Subroutine SSCOMP

Calls subroutines for steady flow solution of all transport equations:

- solves transport equations for Ω
- calls subroutine TRSPRT
- calls subroutine ROTATE. Calls subroutine BOUSET

- calls subroutine STREAM to compute velocity field
- writes out monitor node values
- calls subroutine OUTPUT for either interim or final array output
- generates plot data file

This subroutine is referred to as the "outer iteration loop" and is called but once during a case execution. The code spends the majority of the execution time in this routine.

Subroutine EDDY (M)

Computes eddy transport multiplier FR and FZ; this subroutine is not implemented for this version.

Subroutine TRSPRT

Computes quantities Δ and Γ .

Subroutine ROTATE

Solves vorticity transport equation.

Subroutine BOUSET (JUMP)

Sets vorticity boundary conditions.

Subroutine OUTPUT (MODE)

Primary line printer output call routine.

- The primary purpose of this routine is to selectively call the output array writer subroutine AROUT, based on the alpha input read in through the executive routine. The arrays and array header Holleriths are aligned in the call list of AROUT. This subroutine may be called selectively for array writing through the input Fortran variable NOUT.

- The secondary purpose of subroutine OUTPUT is to selectively write out the convergence rate information computed in subroutine SSCOMP (that is, maximum changes in Ψ and Ω , and the nodal location of these changes, during successive iterations). The iteration numbers selected for output are specified by the input Fortran variable NTTY, in the exact manner that NOUT is used in the primary purpose described above.

Subroutine AROUT (list)

A general array writer, this subroutine is used to write out all computed arrays specified for printing. The appropriate array, header and grid coordinates are aligned in the call list at subroutine OUTPUT. Miscellaneous computations are also performed here as necessary. For instance, if normalized arrays are desired, these are normalized in AROUT, and if temperature arrays are required, the buoyancy parameter(Δ) array is converted to a temperature array through successive calls to function TEMP.

Function SFLUX (list)

Computes surface heat transfer from table data.

Subroutine PCHCRD (list)

Creates punch card file of cell-centered velocities.

Subroutine MAPPER (list)

Prints cell corner and cell center node type identification to line printer.

Function SIGMAT (SAL, T, N)

Given the salinity, SAL, and temperature, T, this function computes sigma-t based on algebraic equations given in Reference 9 or in Reference 10.

Function TEMP (SALT, SIGMA)

Given the salinity, SALT, and the density in sigma-t units, SIGMA, this function solves the equations referenced above for the temperature in degrees centigrade by the Newton-Raphson method. The function SIGMAT (SAL, T, N) is repeatedly called during the iteration process.

6.0 COMPUTER CODE TEST RESULTS

Development of the VECTRA code has proceeded by laying out a foundational computational scheme for arbitrary two-dimensional geometry which solves the complete equations for steady, incompressible turbulent flow having a minimum of other complexities. The philosophy underlying this approach is to provide a sound base code which can be improved incrementally to eventually provide a broad range computational capability. The current code version features:

- steady flow solution by the vorticity-stream function approach;
- x-y coordinate system;
- uniform bottom geometry;
- no bottom or surface shear forces;
- constant eddy transport coefficient; and
- heat and constituent transport.

Results presented in this document were computed using the foundation version of VECTRA which has the base capabilities listed above.

The anticipated final version of the VECTRA computer code will be designed to treat two-dimensional flows in either x-y or x-z coordinates having arbitrary geometry, with these possible improved features:

- steady or pseudo-transient flow (uniformly fluctuating water level);
- buoyancy coupling in x-z coordinates;
- approximate method for the third dimension;
- variable bottom geometry;

- multi-layered capability;
- bottom drag and wind shear;
- time-dependent heat transfer;
- heat and constituent transport;
- variable eddy transport coefficients; and
- k- ϵ turbulence model⁽²³⁾

6.1 LAMINAR FLOW PROOF TESTS

The simplest viscous fluid systems to analyze are those in which the flow is laminar. Although such flows are of little importance as far as environmental systems are concerned, they provide a class of flows for which much analytical and experimental information is available. This information can be compared with computed results to ensure that the basic computation scheme is sound. If the code predicts various details of laminar flow that agree well with experimental data and/or other computational methods, then the code is correctly solving the governing partial differential equations and the boundary conditions are being handled properly. For this reason, the two laminar flows illustrated in Figure 6-1 were investigated in some detail using the VECTRA code.

6.1.1 Channel Flow

The first test problem investigated was the hydrodynamic entry between parallel flat plates. Calculations were carried out for channel Reynolds numbers of 250 and 500, where the Reynolds number is defined as

$$Re = \frac{Lv_o}{\nu} , \quad (6.1)$$

where $2L$ is the channel width and ν is kinematic viscosity. In both cases, the entrance velocity, v_o , was assumed uniform across

FIGURE 6-1. Laminar Channel Flow Proof Test Cases

the channel. The entrance vorticity boundary condition was set at the first cell-centered fluid node and calculated using the vorticity definition,

$$\Omega = \frac{\partial U}{\partial Y} - \frac{\partial V}{\partial X} . \quad (6.2)$$

Results of these tests are illustrated in Figures 6-2 through 6-5.

Figure 6-2 compares the computed centerline velocity to results obtained by other researchers. The irrotational entrance solutions require $\Omega = 0$ at the entrance of the channel.

FIGURE 6-2. Comparison of Centerline Velocity with Other Computational Techniques

FIGURE 6-3. Centerline Velocity for a Laminar Channel at Various Reynolds Number

FIGURE 6-4. Computed Channel Axis Centerline Velocity Distribution

FIGURE 6-5. Computed Velocity Profiles for Channel Flow at Various Downstream Distance, y/L

McDonald, et al.,⁽²⁸⁾ also investigated the condition $\Omega = \partial V / \partial X$ at that boundary. The boundary condition for inlet vorticity used by VECTRA [equation (6.2)] gives results which lie slightly below McDonald's computations using $\Omega = -\partial V / \partial X$ (Figures 6-3 and 6-4). The condition $\Omega = 0$ is a very "soft" boundary condition and gives results considerably different than $\Omega = \partial V / \partial X$, which is a very "stiff" condition. Equation (6.2) is a compromise between these conditions but yields results quite close to those obtained using the "stiff" condition.

Entrance velocity profiles at various downstream positions are shown in Figure 6-5. The theoretical velocity profile for fully developed flow is parabolic, yielding a centerline value of $v_{CL} = 1.5v_o$. Figure 6-6 compares the computed profile for $Re = 250$ to the parabolic distribution. The computed profile is not quite fully established near the centerline ($v_{CL} = 1.49v_o$), but otherwise matches the parabolic profile extremely well. An additional check on the solution scheme is provided by the cross channel vorticity distribution in the fully developed region. Theoretically,

$$V(X) = 1.5(1-X^2), \quad (-1 \leq X \leq 1). \quad (6.3)$$

The vorticity associated with this profile is found by differentiation, which yields

$$\Omega = \frac{-\partial V}{\partial X} = 3X \quad (-1 \leq X \leq 1). \quad (6.4)$$

In Figure 6-7, equation (6.4) is compared with near fully developed flow as computed by VECTRA. Again, there is excellent agreement between the computed results and the actual distribution.

FIGURE 6-6. Comparison Between Computed and Theoretical Velocity Profile at Near-Fully Developed Laminar Channel Flow

6.1.2 Laminar Channel Flow with Sudden Enlargement

Further proof-testing of the VECTRA code was conducted by performing computation of laminar channel flow with a sudden enlargement (See Figure 6-1). The Reynolds numbers were based on the channel entrance half-width, L ; values of 16.5, 65.0 and 162.5 were used for the computation.

Computation centerline velocity distributions for these three cases are shown in Figure 6-8; Figure 6-9 illustrates profiles at various downstream cross sections for $Re = 162.5$.

FIGURE 6-7. Lateral Vorticity Distribution for Fully Established Flow

FIGURE 6-8. Centerline Axial Velocity for Various Reynolds Numbers

FIGURE 6-9. Lateral Velocity Profiles at Various Axial Locations

Contour plots of streamline and vorticity level lines are illustrated by Figures 6-10 through 6-15 for the three Reynolds numbers mentioned above. At this writing, these results have not been compared to experimental values.

6.1.3 Flow to a Shoreline Intake

Figure 6-16 illustrates a possible configuration for a rivershore cooling water intake system. This particular configuration was selected to demonstrate the capability of the code to predict eddy formation caused by either the diverging river channel shape or the presence of the downstream berm.

FIGURE 6-10. Streamlines for
Inlet $Re = 16.25$
Sudden Expansion

FIGURE 6-11. Vorticity for
Inlet $Re = 16.25$
Sudden Expansion

FIGURE 6-12. Streamlines for
Inlet Re = 65

FIGURE 6-13. Vorticity for
Inlet Re = 65

FIGURE 6-14. Streamlines for
Inlet $Re = 162.5$
Sudden Expansion

FIGURE 6-15. Vorticity for
Inlet $Re = 162.5$
Sudden Expansion

FIGURE 6-16. Schematic Diagram of River Shoreline Intake System for Cases I-1 Through I-4 and IB-1 Through IB-4

Eddy formation in front of the intake is, of course, an undesirable consequence because of siltation and possible promotion of a fish trap effect.

Referring to Figure 6-16, two geometric situations are illustrated: Cases I-1 through I-4 have no downstream berm, while Cases IB-1 through IB-4 include the berm.

For all cases the river is assumed to be 700 ft wide with a volumetric flow rate of 14,000 cfs. The river is assumed to have a constant 20-ft depth, a parabolic upstream velocity distribution and average velocity of 1 fps. The four cases examined for each of the two geometries involve four intake flow rates as shown in Table 6-1.

Streamlines of steady flow patterns for the four no-berm cases are shown in Figure 6-17. Similar information is illustrated in Figure 6-18 for the four berm cases. Case I-1 shows that an eddy will develop in front of this intake structure and is caused by the ramp shape of the shoreline. Intake causes the eddy to move to the upstream side of the intake structure and the eddy grows increasingly weaker with increasing intake rates.

TABLE 6-1. Intake Flow Case Description

	<u>Intake Flow Ratio (%)</u> <u>(Intake Flow/River Flow)</u>
No Berm	
I-1	-0-
I-2	1.125
I-3	2.81
I-4	5.62
Berm	
IB-1	-0-
IB-2	1.125
IB-3	2.81
IB-4	5.62

The especially large area of intake influence associated with low velocity water might attract fish and prolong their exposure to the intake flow.

In the case of the berm, two eddies are created, one in front of the intake and the other downstream of the berm. As was the case without the berm, increasing the intake rate causes the eddy to weaken and to move upstream. In this particular case, the berm decreases the volume of low velocity water by eliminating possible downstream intake. However, if the berm were to penetrate the mainstream flow, the volume of water circulating and exposed to the intake would be increased.

These examples of intake flow demonstrate the capability of VECTRA to simulate formation of eddies which possibly promote the adverse effects of siltation and fish trapping. The computational procedure in these cases could be significantly improved by modifying the code to accommodate depth variations and bottom drag.

FIGURE 6-17. Flow Patterns for Intake Cases I-1 Through I-4

FIGURE 6-18. Flow Patterns for Intake Cases IB-1 Through IB-4

6.1.4 Computations Using Frozen Vorticity

The concept of using frozen vorticity for establishing mass-conserving velocity fields was introduced in Section 3.6.6. This idea can be used in conjunction with both field velocity data and analytical solutions. Two generic examples are illustrated to demonstrate the utility of the approach and how it can improve the flow solution in certain cases, while reducing computer time and code complexity.

Figures 6-19 and 6-20 illustrate these examples defined by:

Case 1. A small cooling pond.

Case 2. An embayment with a power plant intake/
discharge.

Both cases involve a shoreline intake/discharge system and constant receiving water depth is assumed.

For illustrative purposes, it is assumed that the discharge flow may be analytically described by a similarity solution for a planar, turbulent flow momentum jet. Abraham⁽²⁹⁾ gives the mainstream velocity distribution for the plane jet as

$$U(x,y) = \frac{2.3 B_o^{1/2} U_o}{x^{1/2}} e^{-k(\frac{y}{x})^2} \quad (6.5)$$

where

x = coordinate along the jet axis;

y = coordinate normal to the jet axis;

U_o = average velocity at discharge point;

k = experimentally determined entrainment parameter;

B_o = initial width of the discharge.

FIGURE 6-19. Example Case 1: Small Cooling Pond Flow

FIGURE 6-20. Example Case 2: Intake/Discharge to Embayment

The approximate vorticity field induced by the jet discharge is obtained by differentiation of equation (6.5) with respect to y . Thus,

$$\omega = \frac{du}{dy} = - \frac{4.6 B_o^{1/2} U_o k y}{x^{5/2}} e^{-k(\frac{y}{x})^2}. \quad (6.6)$$

Once ω is established by equation (6.6), the complete velocity field may be obtained by solution of

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega. \quad (6.7)$$

The flow field for the small cooling pond test Case 1 was computed using three different conditions:

1. $\omega = 0$, potential flow.
2. ω computed by complete viscous flow solution.
3. ω computed by equation (6.6) (frozen jet vorticity).

Results of these computations are illustrated in Figures 6-21, 6-22, and 6-23. Figure 6-21 indicates typical potential flow streamlines, a result which is totally unacceptable in the vicinity of the discharge where recirculation normally occurs. The flow field shown in Figure 6-22 reveals expected behavior near the discharge flow in that jet-induced recirculation patterns are present. However, a constant value of lateral momentum diffusivity was used in this case with no-slip boundary conditions. It is expected that the jet would spread less rapidly and circulation would increase if the turbulence were properly modeled.

Results for frozen jet vorticity are illustrated in Figure 6-23. This computation preserves the characteristics of the momentum jet and indicates considerable circulation. Except for the jet flow, the flow field is potential as may be noted by comparing the streamlines of the intake side with those shown in Figure 6-21.

FIGURE 6-21. Small Pond Streamlines
(Potential Flow)

FIGURE 6-22. Small Pond Streamlines
(Computed Vorticity)

FIGURE 6-23. Small Pond Streamlines
(Frozen Vorticity)

Results for the embayment (Case 2) are shown in Figures 6-24, 6-25, and 6-26. The viscous flow solution streamlines are given by Figure 6-24. Again, a constant value of lateral momentum diffusivity was used which results in unrealistic recirculation to the intake. Results for the frozen vorticity simulation are shown in Figures 6-25 (streamlines) and 6-26 (isotherms). In this example, the offshore flow vorticity field was computed numerically from the profile in Figure 6-20.

The use of frozen vorticity demonstrated by the preceding generic examples has several significant aspects:

1. The method is convenient for establishing mass-conserving velocity distributions appropriate for transport modeling.
2. The use of near field similarity solutions cannot account for recirculation. Hence, thermal plume solutions are inaccurate where reentrainment of heated water occurs. Frozen vorticity provides a convenient method for simulating the approximate interaction between the near and far field.
3. Because equation (6.7) is rather easily solved, the flow solution is significantly simplified compared to the full viscous flow momentum equations.
4. Although the approach is approximate, the results may very well be more accurate than provided by more elaborate numerical approaches because it allows the systematic combining of analytical solutions, field data and the researchers judgement.

FIGURE 6-24. Embayment Streamlines

FIGURE 6-25. Embayment Streamlines
(Frozen Vorticity)

FIGURE 6-26. Embayment Isotherms
(Frozen Vorticity)

REFERENCES

1. K. Aziz, A Numerical Solution of Cellular Convection. Ph.D. dissertation, Rice University, Houston, TX, 1966.
2. B. W. Nichols and C. W. Hirt, Transient Three-Dimensional Fluid Flow in the Vicinity of Large Structures. LA-DC-72-711, Los Alamos Scientific Laboratory, Los Alamos, NM, 1972.
3. D. K. Brady and John C. Geyer, Development of a General Computer Model for Simulating Discharges in Three Dimensions. Report No. 7 to the Edison Electric Institute, Cooling Water Discharge Project RP-49 by the Johns Hopkins University, Baltimore, MD, 1972.
4. D. J. Baumgartner and D. S. Trent, Ocean Outfall Design, Part 1: Literature Review and Theoretical Development. Federal Water Quality Administration, Environmental Protection Agency, Corvallis, OR, 1970.
5. D. R. Smith, Comparison of Theoretical Ocean Diffusion Models. USNRDL-TR-67-50, U.S. Naval Radiological Defense Laboratory, San Francisco, CA, 1967.
6. D. C. Kolesar and J. C. Sonnichsen, Jr., TOPLYR-II: A Two-Dimensional Thermal-Energy Transport Code. HEDL-TME-72-46, Hanford Engineering Development Laboratory, Richland, WA, March 1972.
7. D. S. Trent and J. R. Welty, Numerical Thermal Plume Model for Vertical Outfall in Shallow Water. EPA-R2-73-162, U.S. Environmental Protection Agency, Corvallis, OR, 1973.
8. O. M. Phillips, The Dynamics of the Upper Ocean. Cambridge University Press, London, England, 1966.
9. Tables for Sea Water Density. Publication No. 615, U.S. Navy Hydrographic Office, Washington, DC, 1952.
10. M. N. Hill, ed., The Sea, vol 1. Interscience Publishers, New York, NY, 1962.
11. G. K. Batchelor, An Introduction to Fluid Dynamics. Cambridge University Press, London, England, 1967.

12. G. J. Hirasaki and J. D. Hellums, "Boundary Conditions on the Vector and Scalar Potentials in Viscous Three-Dimensional Hydrodynamics." Quarterly of Applied Mathematics. 28 (2):293-296, 1970.
13. J. E. Welch, F. H. Harlow, J. P. Shannon and B. J. Daly, The MAC Method - A Computing Technique for Solving Viscous, Incompressible, Transient Fluid-Flow Problems Involving Free Surfaces. LA-3425, Los Alamos Scientific Laboratory, Los Alamos, NM, 1965.
14. G. Neumann and W. J. Pierson, Jr., Principles of Physical Oceanography. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1966.
15. J. O. Wilkes and S. W. Churchill, "The Finite-Difference Computation of Natural Convection in a Rectangular Enclosure." American Institute of Chemical Engineers Journal. 12(1):161-166, 1966.
16. H. Baraket, Transient Natural Convection Flows in Closed Containers. Ph.D. dissertation, University of Michigan, Ann Arbor, MI, 1965.
17. K. E. Torrance and J. A. Rockett, "Numerical Study of Natural Convection in an Enclosure with Localized Heating from Below Creeping Flow to the Onset of Laminar Instability." Journal of Fluid Mechanics. 36(1):33-54, 1969.
18. J. H. VanSant, Laminar Free Convection of Heat Generating Fluids in Horizontal Pipes. UCRL-71569, Lawrence Radiation Laboratory, Livermore, CA, 1969.
19. D. B. Spaulding, Numerical Methods in Fluid Mechanics and Heat Transfer. Report No. EF/TN/A/34, Department of Mechanical Engineering, Imperial College, London, England, 1970.
20. D. B. Spaulding, Heat and Mass Transfer in Recirculating Flows. Report No. EF/TN/A/8, Department of Mechanical Engineering, Imperial College, London, England, 1968.
21. D. B. Spaulding, Mathematical Models of Turbulent Flames. Report No. BL/TN/B/22, Department of Mechanical Engineering, Imperial College, London, England, 1969.

22. A. D. Gosman, et al., Heat and Mass Transfer in Recirculating Flows. Academic Press, London, England, 1969.
23. B. E. Launder and D. B. Spaulding, Mathematical Models of Turbulence. Academic Press, London, England, 1972.
24. S. V. Patankar and D. B. Spaulding, "A Finite-Difference Procedure for Solving the Equations fo the Two-Dimensional Boundary Layer." International Journal of Heat and Mass Transfer. 10(10):1389-1412, 1967.
25. J. J. Leenderste, Aspects of a Computational Model for Long Period Water-Wave Propagation. RM-5294-PR, The Rand Corporation, Santa Monica, CA, 1967.
26. J. J. Leenderste, The Behavior of the Water Quality Simulation Model. P-4810, The Rand Corporation, Santa Monica, CA, 1972.
27. G. H. Ward, ed., Estuarine Modeling: An Assessment. Doc. No. 16070DZV, Environmental Protection Agency, Washington, DC.
28. J. W. McDonald, V. E. Denny and A. F. Mills, "Numerical Solutions of the Navier-Stokes Equations in Inlet Regions." Transactions of the ASME, Journal of Applied Mechanics, No. 72-APM-DD, 1972.
29. G. Abraham, Jet Diffusion in Stagnant Ambient Fluid. Publication No. 29, Delft Hydraulics Laboratory, Delft, Netherlands, July 1963.

APPENDIX A

COMPUTER CODE LISTING

```

PROGRAM VECTRA (PUNCH,INPUT,OUTPUT,INTAPE,OUTAPE,TAPE7=INTAPE,TAPE
*8=OUTAPE,TAPE5=INPUT,TAPE6=OUTPUT)
C *****
C CDC CYBER 7V VERSION
C *****

C ***** PARTIAL LIST OF PROGRAM VARIABLES *****
C
C GAM(J,K) CONSERVATIVE CONSTITUENT CONCENTRATION
C OMEG(J,K) VORTICITY
C PSI(J,K) STREAM FUNCTION
C SC(J,M) RADIAL PROGRAMMING CONSTANTS
C SZ(L,K) VERTICAL PROGRAMMING CONSTANTS
C UX (J,K) DIMENSIONLESS VELOCITY AT POINT (X,Z), X-COMPONENT
C UZ (J,K) DIMENSIONLESS VELOCITY AT POINT (X,Z), Z-COMPONENT
C FR(J,K) X-DIRECTION EDDY MULTIPLICATION FACTORS
C FZ(J,K) Z-DIRECTION EDDY MULTIPLICATION FACTORS
C DX WIDTH OF NODE, X-DIRECTION,
C DZ WIDTH OF NODE, Z-DIRECTION
C RER X-DIRECTION REFERENCE TURBULENT REYNOLDS NUMBER
C REZ Z-DIRECTION REFERENCE TURBULENT REYNOLDS NUMBER
C RRP X-DIRECTION REFERENCE PRANDTL NUMBER
C ZRP Z-DIRECTION REFERENCE PRANDTL NUMBER
C
C IPMAX MAXIMUM NUMBER OF ITERATIONS FOR PSI ITERATION
C ITMAX TOTAL NUMBER OF ITERATIONS
C KASE CASE NUMBER
C NJ NUMBER OF NODES, RADIAL DIRECTION
C NJJ NJ-1
C NK NUMBER OF NODES, VERTICAL DIRECTION
C NKK NK-1
C NOX(J) NUMBERING FOR OUTPUT HEADING, SET IN MAIN PROGRAM
C NOUT NUMBER OF ITERATIONS FOR LINE PRINTER OUTPUT
C NITY NUMBER OF ITERATIONS FOR CALCULATION MONITORING OUTPUT
C NPI NUMBER OF ITERATIONS ON STREAM FUNCTION IN MAIN COMP
C NWRITE(J) SIGNAL TO CALL OUTPUT OF SPECIFIC DATA
C OLIST(J) CHARACTER DATA INPUT SIGNAL OUTPUT ARRAYS DESIRED
C OPLIST(J) MATCHES IMBEDDED DATA OLIST(J)
C TO SET VALUE OF NWRITE(J)
C OLIST(1) = PSIP, WRITE POTENTIAL FLOW STREAM FUNCT
C OLIST(2) = PSIV, WRITE VISCOUS FLOW STREAM FUNCT
C OLIST(3) = DELT, WRITE DENSITY DISPARITY
C OLIST(4) = OMEG, WRITE VORTICITY
C OLIST(5) = VELV, WRITE Z-DIRECTION VELOCITY
C OLIST(6) = VELR, WRITE X-DIRECTION VELOCITY
C OLIST(7) = GAM4, WRITE GAMMA CONSTITUENT
C OLIST(8) = TEMP, WRITE TEMPERATURES
C OLIST(9) = QSUR, WRITE SURFACE HEAT FLUX
C OLIST(10) = NVEL, WRITE NORMALIZED VERT. VELOCITY
C OLIST(11) = NTEM, WRITE NORMALIZED TEMPERATURE
C OLIST(12) = RFAC, WRITE RADIAL EDDY FACTORS
C OLIST(13) = VFAC, WRITE VERTICAL EDDY FACTORS
C OLIST(14) = RICH, WRITE RICHARDSON NUMBERS

```

```

C          OLIST(15) = BLANK AT PRESENT
C          CONTRL(1) LOGICAL DATA FOR PROGRAM CONTROL
C              CONTRL(1) = .TRUE. IF DIRECT(1) = CYLN
C              CONTRL(2) = .TRUE. IF DIRECT(1) = TRAN
C              CONTRL(3) = .TRUE. IF DIRECT(1) = BUOY
C              CONTRL(4) = .TRUE. IF DIRECT(1) = GCON
C              CONTRL(5) = .TRUE. IF DIRECT(1) = MONT
C              CONTRL(6) = .TRUE. IF DIRECT(1) = SAVE
C              CONTRL(7) = .TRUE. IF DIRECT(1) = PCHC
C              CONTRL(8) = .TRUE. IF DIRECT(1) = DBUG
C              CONTRL(9) = .TRUE. IF DIRECT(1) = TAPE
C              CONTRL(10) = .TRUE. IF DIRECT(1) = INVS
C              CONTRL(11) = .TRUE. IF DIRECT(1) = HOMO
C              CONTRL(13) = .TRUE. IF DIRECT(1) = INTR
C              CONTRL(14) = .TRUE. IF DIRECT(1) = QSUR
C          OUT LINE PRINTER OUTPUT LUN

DIMENSION OLIST(15),OLIST(15),CLIST(15),ELIST(5),RLIST(5)
DIMENSION OPTION(3),DATA(15),DIRECT(15)
LOGICAL CONTRL
INTEGER OUT,SIND
EQUIVALENCE(VREF,V0)
COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1 GAM(62,32),G(62,32),G(62,32),
2 S2(32,15),Z(32),UZ(32),SC(62,15),R(62),RC(62),
3 T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4 T6(62,32),QS(150),PSTART(5),
5 R05(32),TEM(10),ZC(32),N3OPT(5),
6 OMYR,TIM,TLABEL(8),UB(300),VB(300),
7 CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
COMMON /2/ NOX(42),NWRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1 MAP(2,4),JINDU(300),KINDU(300),KINOV(300),JINDV(300),
2 INDEX(50,8),NSTART(10),SIND(70,4),NPEN(300)
COMMON /3/ DX,DX2,DZC,R0,V0,HER,REZ,RRP,ZRP,EXR,EXS,EXT,
1 OPMAX,DOMAX,DDMAX,TINT,START,F0,
2 TREF,TZERO,GREF,GZERO,DMULT,HZERO,DELTR,DZERO,
3 DREF,VBOMB,EZ0,ER0
COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1 ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2 ITEMP,NEO,ITNOQ,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
COMMON /5/ INDU,INDV,IN08,NREF,NSPSI,ISLIP,LINES
DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
DATA(OLIST(I),I=1,15)/SHPSIV,,SHPSIV,,SHOELT,,SHOMEG,,SHVELV,,
1 SHVELR,,SHGAMA,,SHTEMP,,SHQSUR,,SHNVEL,,
1 SHNTEM,,SHRFAC,,SHVFAC,,SHRICH,,SH /
DATA(ELIST(I),I=1,5)/SHPSIV,,SHOELT,,SHGAMA,,SHTEMP,,SHOMEG,/
DATA(OPTION(I),I=1,3)/SHAOUT,,SHPLOT,,SHCONT,/
DATA(DIRECT(I),I=1,15)/SHCYLN,,SHTRAN,,SHBUOY,,SHGCON,,SHMONT,,
1 SHSAVE,,SHPCHC,,SHDEBUG,,SHTAPE,,SHINVS,,SHHOMO,,
2 SHTURB,,SHINTR,,SHQSUR,,SH /
CALL TIME(TIM)
CALL DATE(OMYR)

```

```

OUT      = 6
IN       = 5
DO 3 I = 1,15
CONTRL(I) = .FALSE.
3 CONTINUE
EXT      = 1,6
DO 4 J = 1,4
N        = (-1)**J
N1       = (J+1)/4
N2       = J/5
MAP(1,J) = N*(1-N1)
MAP(2,J) = N*(N1-N2)
4 CONTINUE
5 READ(IN,1002) TLABEL
READ(IN,1002) KASE,NJ,NK,IPMAX,NPI,NPSTRT,NTEMP
IF(NPI .EQ. 0) NPI = 1
IF(NPI .GT. 10) NPI = 0
IF(IPMAX .EQ. 0) IPMAX = 100
IF(NPSTRT .EQ. 0) NPSTRT = 1
IF(NTEMP .EQ. 0) NTEMP = 70
IF(KASE .EQ. 0) STOP
WRITE(OUT,1004) TLABEL,DMYR,TIM
READ(IN,1000) NOUT,NTTY,ITMAX,NEDDY,NED
6 READ(IN,1001) TLIST,DATA
IF(TLIST.EQ.OPTION(1)) GO TO 7
IF(TLIST.EQ.OPTION(2)) GO TO 9
IF(TLIST.EQ.OPTION(3)) GO TO 11
GO TO 14
7 DO 8 I = 1,15
8 OLIST(I) = DATA(I)
GO TO 6
9 DO 10 I = 1,5
KPLDT      = 1
10 RLIST(I) = DATA(I)
GO TO 6
11 DO 12 I = 1,15
12 CLIST(I) = DATA(I)
GO TO 6
14 NJJ      = NJ-1
NKK        = NK-1
NH         = NK+1
NL         = NJ+1
ITERS      = IPMAX
ITN00      = 0
ITNO       = 0
DO 17 I = 1,5
PSTART(I)  = NPSTRT
17 CONTINUE
DO 18 M = 1,NL
18 NOX(M)   = M
DO 20 J = 1,NJ
DO 20 K = 1,NK
OMEG(J,K) = 0.
T1(J,K)   = 0.

```

```

T2(J,K) = 0.
T3(J,K) = 0.
T4(J,K) = 0.
T5(J,K) = 0.
T6(J,K) = 0.
G(J,K) = 0.
Q(J,K) = 0.
UZ(J,K) = 0.
UX(J,K) = 0.
PSI(J,K) = 0.
TEMP(J,K) = NTEMP
20 CONTINUE
DO 25 J = 1,NL
DO 25 K = 1,NH
NC(J,K) = 50
NT(J,K) = 50
25 CONTINUE
DO 30 N = 1,3
K = (N-1)*10
MON(K+1) = 2
MON(K+2) = NK/3
MON(K+3) = 2
MON(K+4) = 2*MON(K+2)
MON(K+5) = 2
MON(K+6) = NK
MON(K+7) = NJ/2
MON(K+8) = NK
MON(K+9) = NJ-1
MON(K+10) = NK
30 CONTINUE
DO 100 I = 1,15
DO 100 J = 1,15
IF(OLIST(I).EQ.OLIST(J)) WRITE(J) = I
100 CONTINUE
DO 105 I = 1,5
DO 105 J = 1,5
IF(ELIST(I).EQ.RLIST(J)) N3DPT(J) = I
105 CONTINUE
DO 110 I = 1,15
DO 110 J = 1,15
IF(DIRECT(I).EQ.CLIST(J)) CONTRL(I) = .TRUE.
110 CONTINUE
CALL INPUT
CALL TYPE
CALL READY
IF(CONTRL(3).OR.CONTRL(4)) CALL SEMBLE
CALL PLABAK
CALL TIME(START)
WRITE(OUT,1003) START
IF( CONTRL(10)) CALL STREAM(ITERS,0)
IF(ITMAX.EQ.0) GO TO 150
EXT = EXS
CALL OUTPUT(5)
CALL SSCOMP

```



```

      GO TO 100
C**  READ FLOW BOUNDARY NODE TYPES (OPTION 2)      (IT = 2)
120  CONTINUE
      JR      = 0
      DO 125 L = 1,10
      NUMBER  = NDATA(L)
      JO      = 5
      DO 125 NO = 1,JO
      M      = JO-NO+1
      JR      = JR+1
      NOTE    = NUMBER/10**(M-1)
      NT(JR,IM) = NOTE*10
      NUMBER  = NUMBER -NOTE*10**(M-1)
      IF(JR.EQ.NJ) GO TO 126
125  CONTINUE
126  GO TO 100
C**  READ SPECIAL NODE TYPES (FOR OPTION 1)      (IT = 3)
130  NT(IM,JM) = NDATA(1)
      GO TO 100
C**  READ NODE INDEX PAIRS (JS,KS) FOR START OF STREAM FUNCTION
C**  BOUNDARY LINE INTEGRATION, 5 PAIR MAXIMUM (5 DISCONNECTED
C**  FLOW REGIONS)                                (IT = 4)
140  DO 145 I = 1,10
      NSTART(I) = NDATA(I)
145  CONTINUE
      NSPSI    = IM
      GO TO 100
150  CONTINUE
      GO TO 100
160  CONTINUE
      GO TO 100
C**
C**  READ REAL DATA
C**
200  READ (IN,1000) DATA,JI,KI,NI
      NI      = NI+1
      GO TO(400,210,220,230,240,250,260,270,280,290,300,320,330),NI
C**  READ REFERENCE QUANTITIES                      (NI = 1)
210  REFD      = DATA(1)
      VREF     = DATA(3)
      HZERO    = DATA(4)/REFD
      TREF     = DATA(5)
      YZERO    = DATA(6)
      GREF     = DATA(7)
      GZERO    = DATA(8)
      VBOMB    = DATA(9)
      IF(VBOMB.EQ.0.0) VBOMB = 10.*VREF
      GO TO 200
C**  READ PARAMETERS                                (NI = 2)
220  OX      = DATA(1)
      OZC    = DATA(2)
      ER0    = DATA(3)*1.E-2
      EZ0    = DATA(4)*1.E-2
      IF(DATA(5).EQ.0.) OATA(5) = 1.

```



```

      IF(DATA(6).EQ.0.) DATA(6) = 1.
      RRP      = 1./DATA(5)
      ZRP      = 1./DATA(6)
      EXS      = DATA(7)
      EXR      = DATA(6)
      IF(EXS.EQ.0.) EXS = .999
      IF(EXR.EQ.0.) EXR = .999
      GO TO 200
C** READ VARIABLE DZ(K) (NI = 3)
230 DO 235 N = JI,KI
      KIN      = N-JI+1
      DZ(N)    = DATA(KIN)
235 CONTINUE
      GO TO 200
C** READ VARIABLE DR(J) (NI = 4)
240 DO 245 N = JI,KI
      KIN      = N-JI+1
      DR(N)    = DATA(KIN)
245 CONTINUE
      GO TO 200
250 CONTINUE
C** READ SURFACE HEAT TRANSFER TABLE (NI = 5)
DO 255 N = JI,KI
      KIN      = N-JI+1
      QS(N)    = DATA(KIN)
255 CONTINUE
      GO TO 200
C** READ BOUNDARY VELOCITY COMPONENTS (NI = 6)
260 CONTINUE
      I1      = DATA(1)
      I2      = DATA(2)
      J1      = DATA(3)
      J2      = DATA(4)
      DO 265 L = I1,I2
      DO 265 M = J1,J2
      UX(L,M) = DATA(5)
      UZ(L,M) = DATA(6)
265 CONTINUE
C** READ SPECIAL BOUNDARY NODE VELOCITIES (NI = 7)
270 CONTINUE
      UX(JI,KI) = DATA(1)
      UZ(JI,KI) = DATA(2)
      GO TO 200
C** READ BOUNDARY TEMPERATURES AND GAMA (NI = 8)
280 CONTINUE
      I1      = DATA(1)
      I2      = DATA(2)
      J1      = DATA(3)
      J2      = DATA(4)
      DO 285 L = I1,I2
      DO 285 M = J1,J2
      TEMP(L,M) = DATA(5)
      GAM (L,M) = DATA(6)
      Q(L,M)   = DATA(7)

```

```

      G(L,M) = DATA(8)
285 CONTINUE
      GO TO 200
C** READ SPECIAL BOUNDARY NODE TEMP, AND GAMA (NI = 9)
290 TEMP(JI,KI) = DATA(1)
      GAM (JI,KI) = DATA(2)
      Q(JI,KI) = DATA(3)
      G(JI,KI) = DATA(4)
      GO TO 200
C** READ MONITOR NODE INDEX PAIRS (NI = 10)
300 DO 315 N = JI,KI
      KIN = N-JI+1
      MON(N) = DATA(KIN)
315 CONTINUE
      GO TO 200
C* READ STREAM FUNCTION BOUNDRY VALUES (NI = 11)
320 CONTINUE
      DO 325 I = 1,5
      PSTART(I) = DATA(I)
325 CONTINUE
      GO TO 200
330 CONTINUE
      GO TO 200
400 CONTINUE
      NSPSI = 2*NSPSI
      R0 = REFO
      RER = VREF*R0/ER0
      REZ = VREF*R0/EZ0
500 FORMAT(14I5)
1000 FORMAT(10F5.0,3I5)
      RETURN
      END
      SUBROUTINE READY
      LOGICAL CONTRL
      INTEGER OUT,SIND
      EQUIVALENCE(VREF,V0)
      LOGICAL DONE
      COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1 GAM(62,32),Q(62,32),G(62,32),
2 SZ(32,15),Z(32),DZ(32),SC(62,15),R(62),RC(62),
3 T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4 T6(62,32),QS(150),PSTART(5),
5 R05(32),TEM(10),ZC(32),N3OPT(5),
6 DMYR,TIM,TLABEL(8),UB(300),VB(300),
7 CONTRL(15),DR(62),XPLOT(300),ZPLOT(300)
      COMMON /2/ NOX(62),NRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1 MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINOV(300),
2 INDEX( 50,8),NSTART(10),SIND(70,4),NPEN(300)
      COMMON /3/ OX,OX2,OZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1 OPMAX,OOMAX,DOMAX,TINT,START,F0,
2 TREF,TZERO,GREF,GZERO,DMULT,HZERO,DELTR,DZERO,
3 DREF,VBOMB,EZ0,ER0
      COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1 ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,

```

```

2          ITEMP,NED,ITNOO,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
COMMON /5/ INOU,INDV,INDB,NREF,NPSI,ISLIP,LINES
DATA LJ,LK,LBY,L40,L20/62,32,300,50,70/

C**
C**  SET UP COORDINATES
      IF(DZC.EQ.0.) GO TO 11
      DO 10 K = 1,NH
10  DZ(K) = DZC
11  Z(1) = 0.
      DZ(NH) = DZ(NK)
      ZC(1) = Z(1)-.5*DZ(1)
      DO 15 K = 2,NH
      Z(K) = Z(K-1)+ DZ(K)
      ZC(K) = Z(K)-.5*DZ(K)
15  CONTINUE
      IF(DX.EQ.0.) GO TO 21
      DO 20 J = 1,NL
20  DR(J) = DX
21  CONTINUE
      DR(NL) = DR(NJ)
      R(1) = 0.0
      DO 25 J = 2,NL
      RC(1) = R(1)-.5*DR(1)
      R(J) = R(J-1)+DR(J)
      RC(J) = R(J)-.5*DR(J)
25  CONTINUE

C
C
C  SET-UP FOR Z-DIRECTION CONSTANTS

      DO 70 K = 2,NK
      SZ(K,1) = 2./REZ*(1./(DZ(K+1)+DZ(K))+1./(DZ(K-1)+DZ(K)))/DZ(K)
      SZ(K,2) = 2./REZ*(1./(DZ(K)+DZ(K-1)))/DZ(K)
      SZ(K,3) = 2./REZ*(1./(DZ(K+1)+DZ(K)))/DZ(K)
      SZ(K,4) = 1./(2.*DZ(K))
      SZ(K,5) = 2./(DZ(K)*DZ(K+1))
      SZ(K,6) = 2./(DZ(K)*(DZ(K)+DZ(K+1)))
      SZ(K,7) = 2./(DZ(K+1)*(DZ(K)+DZ(K+1)))
      SZ(K,8) = 1./(DZ(K)+DZ(K-1))
      SZ(K,9) = 1./(DZ(K)+DZ(K+1))
      SZ(K,10) = SZ(K,8)-SZ(K,9)
70  CONTINUE

C
C
C  SET-UP FOR X-DIRECTION CONSTANTS

      DO 80 J = 2,NJ
      SC(J,1) = 2./RER*(1./(DR(J+1)+DR(J))+1./(DR(J-1)+DR(J)))/DR(J)
      SC(J,2) = 2./RER*(1./(DR(J-1)+DR(J)))/DR(J)
      SC(J,3) = 2./RER*(1./(DR(J+1)+DR(J)))/DR(J)
      SC(J,4) = 1./(2.*DR(J))
      SC(J,5) = 2./(DR(J)*DR(J+1))
      SC(J,6) = 2./(DR(J)*(DR(J)+DR(J+1)))
      SC(J,7) = 2./(DR(J+1)*(DR(J)+DR(J+1)))
      SC(J,8) = 1./(DR(J)+DR(J-1))
      SC(J,9) = 1./(DR(J)+DR(J+1))

```

```

      SC(J,10) = SC(J,8)-SC(J,9)
80  CONTINUE
      IF(.NOT.CONTROL(9)) GO TO 95
      DO 90 J = 1,NJ
      DO 90 K = 1,NK
      N      = NT(J,K)
      IF(N.EQ.10.OR.N.EQ.20.OR.N.EQ.30) PSI(J,K) = 0.
90  CONTINUE
95  CONTINUE
      KOUNT      = 0
      LINE       = 1
      INOU      = 0
      INDV      = 0
      DO 200 NG= 1,NSPSI,2
      NGG      = (NG+1)/2
      IF(NG.EQ.1) GO TO 98
      LINE     = LINE + 1
      NPEN(LINE) = 3
      XPLOT(LINE) = R(JS)
      ZPLOT(LINE) = Z(KS)
98  CONTINUE
      JS      = NSTART(NG)
      KS      = NSTART(NG+1)
      J1      = JS-1
      K1      = KS
      J2      = JS+1
      K2      = KS
      J3      = JS
      K3      = KS-1
      J4      = JS
      K4      = KS+1
      NPEN(LINE) = 3
      XPLOT(LINE) = R(JS)
      ZPLOT(LINE) = Z(KS)
100  DONE      = .FALSE.
      IF(NT(JS,KS).LT.10.OR.NT(JS,KS).GT.20) GO TO 320
      PSI(JS,KS) = PSTART(NGG)
105  JL      = JS
      KL      = KS
      IF(.NOT.DONE) GO TO 110
      LINE     = LINE + 1
      NPEN(LINE) = 3
      XPLOT(LINE) = R(JS)
      ZPLOT(LINE) = Z(KS)
110  CONTINUE
      KOUNT = KOUNT+1
      LINE  = LINE + 1
      NPEN(LINE) = 3
      IF(KOUNT.GT.LBV) GO TO 310
      DO 120 M = 1,4
      JJ      = JL
      KK      = KL
      JM      = MAP(1,M)+JL
      IF(JM.LT.1.OR.JM.GT.NJ) GO TO 120

```

```

      N      = NT(JM,KL)
      IF((N.EQ.10.OR.N.EQ.20.OR.N.EQ.30)
1      .AND.PSI(JM,KL).EQ.0.) GO TO 130
      KM      = MAP(2,M)+KL
      IF(KM.LT.1.OR.KM.GT.NK) GO TO 120
      N      = NT(JL,KM)
      IF((N.EQ.10.OR.N.EQ.20.OR.N.EQ.30)
1      .AND.PSI(JL,KM).EQ.0.) GO TO 140
120 CONTINUE
      IF(JL.EQ.J1.AND.KL.EQ.K1) GO TO 125
      IF(JL.EQ.J2.AND.KL.EQ.K2) GO TO 125
      IF(JL.EQ.J3.AND.KL.EQ.K3) GO TO 125
      IF(JL.EQ.J4.AND.KL.EQ.K4) GO TO 125
      GO TO 126
125 NPEN(LINE)      = 2
      XPLOT(LINE)    = R(JS)
      BX0W1-ORVN*    = 0-01*

      KOUNT      = KOUNT+1
      LINE      = LINE +1
      NPEN(LINE)  = 3
126 IF(DONE) GO TO 160
      DONE      = .TRUE.
      GO TO 125
130 IF(MAP(1,M).EQ.1) JJ = JM
      PSI(JM,KL) = PSI(JL,KL)+DR(JJ)*UZ(JJ,KL)*MAP(1,M)
      INDV      = INDV+1
      VB(INDV)  = UZ(JJ,KL)
      JINDV(INDV) = JJ
      KINDV(INDV) = KL
      DO 131 MM=1,4
      JMM      = MAP(1,MM) + JM
      IF(JMM.LT.1.OR.JMM.GT.NJ) GO TO 131
      NX      = NT(JMM,KL)
      IF(NX.EQ.1.OR.N.EQ.30.OR.N.EQ.40) GO TO 132
      KMM      = MAP(2,MM) + KL
      IF(KMM.LT.1.OR.KMM.GT.NK) GO TO 131
      NV      = NT(JM,KMM)
      IF(NV.EQ.1.OR.N.EQ.30.OR.N.EQ.40) GO TO 132
131 CONTINUE
      JL      = JM
      KOUNT    = KOUNT - 1
      LINE     = LINE  -1
      GO TO 110
132 NB      = NT(JL,KL)
      NPEN(LINE)      = 2
      IF(NB.EQ.30.AND.N.EQ.30) NPEN(LINE) = 0
      IF(NB.NE.30.AND.N.EQ.30.AND.MAP(1,M).EQ. 1) NPEN(LINE) = 3
      IF(NB.NE.30.AND.N.EQ.30.AND.MAP(1,M).EQ.-1) NPEN(LINE) = 2
      IF(NB.EQ.30.AND.N.NE.30.AND.MAP(1,M).EQ. 1) NPEN(LINE) = 2
      IF(NB.EQ.30.AND.N.NE.30.AND.MAP(1,M).EQ.-1) NPEN(LINE) = 3
      XPLOT(LINE) = R(JM)
      ZPLOT(LINE) = Z(KL)
      IF(NB.NE.30.AND.N.EQ.30) GO TO 133

```

```

      JL      = JM
      GO TO 110
133 KOUNT    = KOUNT + 1
      LINE    = LINE + 1
      NPEN(LINE) = 0
      XPLOT(LINE) = R(JM)
      ZPLOT(LINE) = Z(KL)
      JL      = JM
      GO TO 110
140 IF(MAP(2,M).EQ.1) KK = KM
      PSI(JL,KM) = PSI(JL,KL)-OZ(KK)*UX(JL,KK)*MAP(2,M)
      INDU      = INDU+1
      UB(INDU) = UX(JL,KK)
      JINDU(INDU) = JL
      KINDU(INDU) = KK
      DO 141 NN=1,4
      JMM      = MAP(1,NN) + JL
      IF(JMM.LT.1.OR.JMM.GT.NJ) GO TO 141
      NX      = NT(JMM,KM)
      IF(NX.EQ.1.OR.N.EQ.30.OR.N.EQ.40) GO TO 142
      KMM      = MAP(2,NN) + KM
      IF(KMM.LT.1.OR.KMM.GT.NK) GO TO 141
      NV      = NT(JL,KMM)
      IF(NV.EQ.1.OR.N.EQ.30.OR.N.EQ.40) GO TO 142
141 CONTINUE
      KL      = KM
      KOUNT    = KOUNT - 1
      LINE    = LINE - 1
      GO TO 110
142 NB      = NT(JL,KL)
      NPEN(LINE) = 2
      IF(NB.EQ.30.AND.N.EQ.30) NPEN(LINE) = 0
      IF(NB.NE.30.AND.N.EQ.30.AND.MAP(2,M).EQ. 1) NPEN(LINE) = 3
      IF(NB.NE.30.AND.N.EQ.30.AND.MAP(2,M).EQ.-1) NPEN(LINE) = 2
      IF(NB.EQ.30.AND.N.NE.30.AND.MAP(2,M).EQ. 1) NPEN(LINE) = 2
      IF(NB.EQ.30.AND.N.NE.30.AND.MAP(2,M).EQ.-1) NPEN(LINE) = 3
      XPLOT(LINE) = R(JL)
      ZPLOT(LINE) = Z(KM)
      IF(NB.NE.30.AND.N.EQ.30) GO TO 143
      KL      = KM
      GO TO 110
143 KOUNT    = KOUNT + 1
      LINE    = LINE + 1
      NPEN(LINE) = 0
      XPLOT(LINE) = R(JL)
      ZPLOT(LINE) = Z(KM)
      KL      = KM
      GO TO 110
160 CONTINUE
200 CONTINUE
      DO 250 J = 1,NL
      DO 250 K = 1,NH
      IF(NT(J,K).NE.30) GO TO 241
      JX      = J

```

```

      KZ      = K
      DO 210 M = 1,4
      JM      = MAP(1,M)+J
      KM      = MAP(2,M)+K
      N       = NT(JM,KM)
      IF(JM.LT.1.OR.JM.GT.NL) GO TO 210
      IF(KM.LT.1.OR.KM.GT.NH) GO TO 210
      IF(N.EQ.1) GO TO 230
210  CONTINUE
      DO 220 M = 1,4
      JM      = MAP(1,M)+J
      KM      = MAP(2,M)+K
      IF(JM.LT.1.OR.JM.GT.NL) GO TO 220
      IF(KM.LT.1.OR.KM.GT.NH) GO TO 220
      IF(NT(JM,KM).NE.30) GO TO 220
      JL      = -MAP(1,M)+J
      KL      = -MAP(2,M)+K
      IF(JL.LT.1.OR.JL.GT.NL) GO TO 220
      IF(KL.LT.1.OR.KL.GT.NH) GO TO 220
      IF(NT(JL,KL).EQ.10) GO TO 220
      GO TO 240
220  CONTINUE
      GO TO 250
230  DO 235 M = 1,4
      JL      = MAP(1,M)+J
      KL      = MAP(2,M)+K
      N       = NT(JL,KL)
      IF(N.NE.10) GO TO 235
      IF(M.EQ.2.OR.M.EQ.4) GO TO 235
      JM      = MAP(1,M) + JL
      KM      = MAP(2,M) + KL
      IF(JM.LT.1.OR.JM.GT.NL) GO TO 235
      IF(KM.LT.1.OR.KM.GT.NH) GO TO 235
      IF(NT(JM,KM).EQ.10) GO TO 235
      GO TO 240
235  CONTINUE
      GO TO 250
240  CONTINUE
      INDV    = INDV+1
      INDU    = INDU+1
      JINDV(INDV) = JL
      JINDU(INDU) = JL
      KINDU(INDU) = KL
      KINDV(INDV) = KL
      UB(INDU)   = -UX(JX,KZ)
      VB(INDV)   = -UZ(JX,KZ)
      GO TO 250
C*  SET TYPE 30 FICTITIOUS BOUNDARY VELOCITY (DOWN AND LEFT)
241  IF(NT(J,K).NE.10) GO TO 250
      DO 245 M = 2,4,2
      JM      = MAP(1,M)+J
      KM      = MAP(2,M)+K
      IF(JM.LT.1) GO TO 245
      IF(KM.LT.1) GO TO 245

```

```

      IF(NT(JM,KM).NE.30) GO TO 245
      IF(NC(JM,KM).EQ.10) GO TO 245
      JL      = J
      KL      = K
      JX      = JM
      KZ      = KM
      IF(M.EQ.2.AND.NT(J-1,K).EQ.10) GO TO 245
      IF(M.EQ.4.AND.NT(J,K-1).EQ.10) GO TO 245
      GO TO 240
245  CONTINUE
250  CONTINUE
      LINES   = LINE
      IF(.NOT.CONTRL(8)) RETURN
      DO 180 M = 1,LINES
180  WRITE(OUT,2000) M,NPEN(M),XPLT(M),ZPLT(M)
2000  FORMAT(2I5,2X,2F10.5)
      RETURN
310  WRITE(OUT,1001) KOUNT
      GO TO 400
320  WRITE(OUT,1000) JS,KS
      CALL MAPPER(NT,* STREAM FUNCT. NODE TYPES FOR ABORT CASE *)
      CALL MAPPER(NC,* CELL CENTERED NODE TYPES FOR ABORT CASE *)
      GO TO 400
400  STOP
1000  FORMAT(/// * INPUT ERROR*/# INDICES #I3,*,#I3,* NOT A TYPE 10 NODE
1      * /* CASE ABORTED #)
1001  FORMAT(///# DIMENSION ERROR, NUMBER OF BOUNDARY NODES EXCEEDS #I4/
2      * CASE ABORTED #)
      END
      SUBROUTINE TYPE
      LOGICAL    CONTRL
      INTEGER    OUT,SIND
      EQUIVALENCE(VREF,V0)
      COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1      GAM(62,32),Q(62,32),G(62,32),
2      SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3      T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4      T6(62,32),QS(150),PSTART(5),
5      R05(32),TEM(10),ZC(32),N3DPT(5),
6      DMYR,TIM,TLABEL(8),UB(300),VB(300),
7      CONTRL(15),DR(62),XPLT(300),ZPLT(300)
      COMMON /2/ NOX(62),NWRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1      MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2      INDEX( 50,8),NSTART(10),SIND(70,4),NPEN(300)
      COMMON /3/ OX,OX2,OZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1      DPMAX,DQMAX,DUMAX,TINT,START,F0,
2      TREF,TZERO,GREF,GZERO,CMULT,HZERO,DELTR,UZERO,
3      DREF,VBOMB,EZ0,ER0
      COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1      ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2      ITEMP,NED,ITNOO,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
      COMMON /5/ INOU,INDV,INDB,NREF,NSPSI,ISLIP,LINES
      DATA LJ,LK,LBV,L42,L20/62,32,300,50,70/
C**  SUBROUTINE SETS NODE TYPES FOR COMPUTATIONAL CONTROL

```



```

C**  NODE TYPE DESIGNATIONS=
C      TYPE 0    REGULAR
C      TYPE 1    FLUID NODE NEXT TO BOUNDARY
C      TYPE 10   NO-SLIP BOUNDARY
C      TYPE 20   SLIP BOUNDARY
C      TYPE 30   FLOW BOUNDARY
C      TYPE 50   EXTERIOR (NULL)
C**
DO 50 J = 1,NJ
DO 50 K = 1,NK
DO 45 M = 1,4
JM      = MAP(1,M)+J
KM      = MAP(2,M)+K
45 IF(NT(J,K).EQ.0.AND.NT(JM,KM).GE.50) NT(JM,KM) = 10
50 CONTINUE
C**
C**  CHANGE ALL NODES OF TYPE 0 ADJACENT TYPE 10,20,30,OR,40 NODES TO
C**  TYPE 1
C
DO 100 J = 1,NJ
DO 100 K = 1,NK
N      = NT(J,K)
IF(N.EQ.10.OR.N.EQ.20.OR.N.EQ.30.OR.N.EQ.40) GO TO 80
GO TO 100
80 DO 90 M = 1,4
JM      = MAP(1,M)+J
KM      = MAP(2,M)+K
IF(JM.LT.1.OR.JM.GT.NJ.OR.KM.LT.1.OR.KM.GT.NK) GO TO 90
IF(NT(JM,KM).EQ.0) NT(JM,KM) = 1
90 CONTINUE
100 CONTINUE
C**  SET 10 NODES WHICH ARE NOT ACTUAL FLUID BOUNDARY NODES TO TYPE 50
C
DO 120 J = 1,NJ
DO 120 K = 1,NK
MARK    = 0
IF(NT(J,K).NE.10) GO TO 120
DO 110 M = 1,4
JM      = MAP(1,M)+J
KM      = MAP(2,M)+K
N      = NT(JM,KM)
IF(N.EQ.20.OR.N.EQ.30.OR.N.EQ.40.OR.N.EQ.1) MARK = 1
110 CONTINUE
IF(MARK.EQ.0) NT(J,K) = 50
120 CONTINUE
C
C**  SET CELL-CENTERED NODE TYPES
DO 150 J = 1,NL
DO 150 K = 1,NH
IF(NT(J,K).EQ.0) NC(J,K) = 0
IF(NT(J,K).NE.1) GO TO 150
DO 130 M = 1,4
JM      = MAP(1,M)+J
KM      = MAP(2,M)+K

```

```

      N      = NT(JM,KM)
      JJ     = J+1
      KK     = K+1
      IF(N.GE.10.AND.(M.EQ.1.OR.M.EQ.3)) NC(J,K) = 10
      IF(N.GE.10.AND.(M.EQ.2.OR.M.EQ.4)) NC(JJ,KK) = 10
130  CONTINUE
150  CONTINUE
C
C**  SET ALL TYPE 50 NODES WHICH ARE NO-SLIP BOUNDARY NODES TO TYPE 10
DO 160 J = 1,NJ
DO 160 K = 1,NK
IF(NT(J,K).NE.50) GO TO 160
DO 155 M = 1,4
  MJ      = M/3+1
  MK      = M+2-(M/3)*2
  JM      = MAP(1,MJ)+J
  KM      = MAP(2,MK)+K
  IF(JM.LT.1.OR.JM.GT.NJ.OR.KM.LT.1.OR.KM.GT.NK) GO TO 155
  IF(NT(JM,KM).EQ.1) NT(J,K)=10
155  CONTINUE
160  CONTINUE
DO 210 J = 2,NJ
DO 210 K = 2,NK
IF(NC(J,K).NE.50) GO TO 210
MARK      = 0
DO 205 M = 1,4
  JM      = MAP(1,M)+J
  KM      = MAP(2,M)+K
  IF(NC(JM,KM).EQ.0) MARK = 1
205  CONTINUE
  IF(MARK.EQ.1) NC(J,K) = 0
210  CONTINUE
DO 200 J = 1,NL
DO 200 K = 1,NH
190  IF(NT(J,K).NE.20) GO TO 200
DO 195 M = 1,4
  JM      = MAP(1,M)+J
  KM      = MAP(2,M)+K
  IF(JM.LT.1.OR.JM.GT.NL) GO TO 195
  IF(KM.LT.1.OR.KM.GT.NH) GO TO 195
  IF(NT(JM,KM).EQ.50.AND.M.EQ.1) NC(J+1,K) = 20
  IF(NT(JM,KM).EQ.50.AND.M.EQ.2) NC(J,K) = 20
  IF(NT(JM,KM).EQ.50.AND.M.EQ.3) NC(J,K+1) = 20
  IF(NT(JM,KM).EQ.50.AND.M.EQ.4) NC(J,K) = 20
195  CONTINUE
200  CONTINUE
DO 201 J = 1,NL
DO 201 K = 1,NH
IF(NC(J,K).NE.20) GO TO 201
DO 302 M = 1,4
  JM      = MAP(1,M) + J
  KM      = MAP(2,M) + K
  IF(JM.LT.1.OR.JM.GT.NL) GO TO 302
  IF(KM.LT.1.OR.KM.GT.NH) GO TO 302

```

```

      N      = NC(JM,KM)
      IF(N.EQ.20.AND.J.EQ.JM) GO TO 303
      IF(N.EQ.20.AND.K.EQ.KM) GO TO 304
302 CONTINUE
303 CONTINUE
      DO 305 JR= JM,JM
      DO 305 KR= 1,NH
      IF(NC(JR,KR).EQ.50) GO TO 305
      IF(NC(JR,KR).NE.20) NC(JR,KR) = 20
305 CONTINUE
      GO TO 202
304 CONTINUE
      DO 306 JR=1,NL
      DO 306 KR= KM,KM
      IF(NC(JR,KR).EQ.50) GO TO 306
      IF(NC(JR,KR).NE.20) NC(JR,KR) = 20
306 CONTINUE
      GO TO 202
201 CONTINUE
202 CONTINUE
      DO 220 J = 2,NJ
      DO 220 K = 2,NK
      IF(NC(J,K).NE.50) GO TO 220
      IF(NC(J-1,K+1).EQ.0) NC(J,K) = 10
      IF(NC(J+1,K-1).EQ.0) NC(J,K) = 10
220 CONTINUE
      DO 230 J = 2,NL
      DO 230 K = 2,NH
      IF(NT(J,K).EQ.40) NC(J,K) = 0
      IF(NC(J,K).NE.10) GO TO 230
      MARK      = 0
      DO 225 M = 1,4
      JM      = MAP(1,M)+J
      KM      = MAP(2,M)+K
      IF(NC(JM,KM).EQ.50.OR.NC(JM,KM).EQ.20) MARK = 1
      IF(NT(JM,KM).EQ.40) MARK = 0
225 CONTINUE
      IF(MARK.EQ.0) NC(J,K) = 0
230 CONTINUE
      INDB      = 0
      DO 270 J = 1,NJ
      DO 270 K = 1,NK
      IF(NT(J,K).NE.40) GO TO 270
      INDB      = INDB+1
      IF(INDB.GT.L40) GO TO 275
      DO 260 M = 1,4
      JM      = MAP(1,M) + J
      KM      = MAP(2,M) + K
      IF(NT(JM,KM).EQ.10.AND.M.EQ.1) NC(J,K) = 10
      IF(NT(JM,KM).EQ.10.AND.M.EQ.3) NC(J,K) = 10
      IF(NT(JM,KM).NE.1) GO TO 260
      INDEX(INDB,1) = J
      INDEX(INDB,2) = K
      INDEX(INDB,3) = JM

```

```

      INDEX(INDB,4) = KM
      INDEX(INDB,5) = MAP(1,M) + JM
      INDEX(INDB,6) = MAP(2,M) + KM
      INDEX(INDB,7) = -MAP(1,M) + J
      INDEX(INDB,8) = -MAP(2,M) + K
260 CONTINUE
270 CONTINUE
      GO TO 280
275 WRITE(OUT,1020) INDB
      STOP
1000 FORMAT(/// * NUMBER OF OUTFLOW BOUNDARY NODES EXCEEDS DIMENSION SP
      * SPECIFICATION */ * RESET DIMENSIONS OF N IN INDEX(N,6) *)
280 CONTINUE
C** SET INDICES FOR SLIP BOUNDARY BOOKKEEPING
      ISLIP = 0
      DO 300 J = 1,NJ
      DO 300 K = 1,NK
      IF(NC(J,K).NE.20) GO TO 300
      DO 290 M = 1,4
      JM = MAP(1,M)+J
      KM = MAP(2,M)+K
      IF(NC(JM,KM).NE.50) GO TO 290
      SIND(ISLIP+1,1) = JM
      SIND(ISLIP+1,2) = KM
      SIND(ISLIP+1,3) = J
      SIND(ISLIP+1,4) = K
      ISLIP = ISLIP + 1
      IF(ISLIP.GT.L20) GO TO 310
290 CONTINUE
300 CONTINUE
      DO 301 J = 1,NJ
      DO 301 K = 1,NK
      IF(NC(J,K).NE.20) GO TO 301
      NC(J,K) = 0
301 CONTINUE
      GO TO 320
310 WRITE(OUT,1001) ISLIP
      STOP
1001 FORMAT(/// * NUMBER OF SLIP BOUNDARY NODES EXCEEDS DIMENSION
      * SPECIFICATION, L20 (SEE COMDECK FACTS) *)
320 CONTINUE
      RETURN
      END
      SUBROUTINE SEMBLE
      LOGICAL CONTRL
      INTEGER OUT,SIND
      EQUIVALENCE(VREF,V0)
      COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1      GAM(62,32),D(62,32),G(62,32),
2      SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3      T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4      T6(62,32),QS(150),PSTART(5),
5      R05(32),TEM(10),ZC(32),N3OPT(5),
6      DMYR,TIM,TLABEL(8),UB(300),VB(300),

```

```

7      CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
COMMON /2/ NOX(62),NRITE(15),NODE(6),MON(34),NT(62,32),NC(62,32),
1      MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2      INDEX(50,8),NSTART(10),SIND(70,4),NPEN(300)
COMMON /3/ OX,OX2,OZC,R0,V0,REF,REZ,RRP,ZRP,EXR,EXS,EXT,
1      OPMAX,DDMAX,DDMAX,TINT,START,F0,
2      TREF,TZERO,GREF,GZERO,DMULT,HZERO,DELTR,DZERO,
3      DREF,V8QMB,EZ0,ER0
COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1      ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2      ITEMP,NED,ITNGO,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
COMMON /5/ INDU,INDV,INDB,NREF,NPSPI,ISLIP,LINES
DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
DO 40 J = 1,NJ
DO 40 K = 1,NK
T1(J,K) = OR(J)/(OZ(K)*2.*R0*HZERO*ER0*RRP)
T2(J,K) = T1(J,K)
T3(J,K) = OZ(K)/(OR(J)*2.*R0*HZERO*EZ0*ZRP)
T4(J,K) = T3(J,K)
T5(J,K) = .5*OZ(K)*HZERO*R0*R0*V0
T6(J,K) = .5*OR(J)*HZERO*R0*R0*V0
Q(J,K) = OR(J)*OZ(K)*R0**2/(62.4*1.0*3600.)
10 CONTINUE
40 CONTINUE
C** CHANGE RESISTORS TO INTERCELL CONDUCTORS (CENTER-TO-CENTER)
80 DO 90 J = 1,NJ
DO 90 K = 1,NK
T2(J,K) = 1./(T2(J,K)+T1(J+1,K))
T4(J,K) = 1./(T4(J,K)+T3(J,K+1))
90 CONTINUE
DO 100 J = 2,NJ
DO 100 K = 2,NK
T1(J,K) = T2(J-1,K)
T3(J,K) = T4(J,K-1)
100 CONTINUE
C** CORRECT INTERCELL CONDUCTORS TO ACCOUNT FOR ADIABATIC BOUNDARIES
DO 150 J = 1,NJ
DO 150 K = 1,NK
IF(NC(J,K).EQ.0) GO TO 150
IF(NC(J,K).EQ.50) GO TO 140
DO 110 M = 1,4
JM = MAP(1,M)+J
KM = MAP(2,M)+K
N = NC(JM,KM)
IF(N.EQ.50.AND.M.EQ.1) T1(J,K) = 0.
IF(N.EQ.50.AND.M.EQ.2) T2(J,K) = 0.
IF(N.EQ.50.AND.M.EQ.3) T3(J,K) = 0.
IF(N.EQ.50.AND.M.EQ.4) T4(J,K) = 0.
110 CONTINUE
GO TO 150
140 T1(J,K) = 0.
T2(J,K) = 0.
T3(J,K) = 0.
T4(J,K) = 0.

```

```

150 CONTINUE
300 RETURN
END
SUBROUTINE PLABAK
LOGICAL CONTRL
INTEGER OUT,SIND
EQUIVALENCE(VREF,V0)
COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1 GAM(62,32),J(62,32),G(62,32),
2 SZ(32,15),Z(32),DZ(32),SC(62,15),R(62),RC(62),
3 T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4 T6(62,32),QS(150),PSTART(5),
5 R0S(32),TEM(10),ZC(32),N3OPT(5),
6 DMYR,TIM,TLABEL(8),UB(300),VB(300),
7 CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
COMMON /2/ NOX(62),NRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1 MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2 INDEX(50,8),NSTART(10),SIND(70,4),NPEN(300)
COMMON /3/ DX,DX2,DZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1 DPMAX,DOMAX,DDMAX,TINT,START,F0,
2 TREF,TZERO,GREF,GZERO,OMULT,HZERO,DELTR,DZERO,
3 DREF,V0DNB,EZ0,ER0
COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1 ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2 ITEMP,NED,ITNOO,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
COMMON /5/ INDU,INDV,IND8,NREF,NSPSI,ISLIP,LINES
DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
EPR = 1./RRP
EPZ = 1./ZRP
WRITE(OUT,1001) KASE
WRITE(OUT,1002) NJ,NK,DX,DZC,F0,R0,V0,RER,REZ,TZERO,TREF,EPR
WRITE(OUT,1002) EPZ,GZERO,GREF,EXR,EXT
5 WRITE(OUT,1010) DMYR,TIM,(NOX(K),K=1,8)
DO 140 J = 1,NJ
WRITE(OUT,1012) J,OR(J),RC(J),R(J),(SC(J,L),L=1,8)
140 CONTINUE
WRITE(OUT,1010) DMYR,TIM,(NOX(K),K=9,15)
DO 145 J = 1,NJ
WRITE(OUT,1012) J,OR(J),RC(J),R(J),(SC(J,L),L=9,15)
145 CONTINUE
WRITE(OUT,1013)
WRITE(OUT,1014) DMYR,TIM,(NOX(K),K=1,8)
DO 150 K = 1,NH
150 WRITE(OUT,1012) K,DZ(K),ZC(K),Z(K),(SZ(K,L),L=1,8)
WRITE(OUT,1014) DMYR,TIM,(NOX(K),K=9,15)
DO 155 K = 1,NH
WRITE(OUT,1012) K,DZ(K),ZC(K),Z(K),(SZ(K,L),L=9,15)
155 CONTINUE
INTER = 0
DO 10 J = 1,NJ
DO 10 K = 1,NK
IF(NT(J,K).EQ.0.OR.NT(J,K).EQ.1) INTER = INTER+1
10 CONTINUE
WRITE(OUT,1004) INTER

```

```

CALL MAPPER(NT,* STREAM FUNCTION NODE TYPES          *)
CALL MAPPER(NC,* CELL-CENTERED NODE TYPES            *)
IF(.NOT.CONTROL(8)) RETURN
CALL AROUT(1,Z,R,PSI,
1          * STREAM FUNCTION INITIAL VALUES        *)
CALL AROUT(1,Z,PC,UZ,
1          * Z-DIRECTION VELOCITY BOUNDARY VALUES  *)
CALL AROUT(1,ZC,R,UX,
1          * X-DIRECTION VELOCITY BOUNDARY VALUES  *)
WRITE(OUT,1006)
N2      = 1
200 N1      = N2+1
N2      = N1+9
IF(N2.GT.NJ) N2 = NJ
WRITE(OUT,1009)(NOX(I),I=N1,N2)
DO 210 K = 2,NK
KN      = NK+2-K
WRITE(OUT,1017)KN,(T1(J,KN),J = N1,N2)
WRITE(OUT,1018)  (T2(J,KN),J = N1,N2)
WRITE(OUT,1018)  (T3(J,KN),J = N1,N2)
WRITE(OUT,1018)  (T4(J,KN),J = N1,N2)
WRITE(OUT,1018)  (T5(J,KN),J = N1,N2)
WRITE(OUT,1018)  (T6(J,KN),J = N1,N2)
WRITE(OUT,1018)  (Q (J,KN),J = N1,N2)
WRITE(OUT,1018)  (G (J,KN),J = N1,N2)
210 CONTINUE
IF(N2.NE.NJ) GO TO 200
1001 FORMAT(/// * I N P U T   F O R   C A S E   N O.   * I3)
1002 FORMAT(///
1/55H   NUMBER OF X-DIRECTION NODES- - - - - F13.5
3/55H   NUMBER OF Z-DIRECTION NODES- - - - - F13.5
4/55H   LATERAL CELL THICKNESS (X-DIRECTION), DX - - - - F13.5
*/55H   LATERAL CELL THICKNESS (Z-DIRECTION), DZ - - - - F13.5
5/55H   REFERENCE FROUDE NUMBER - - - - - F13.5
6/55H   SYSTEM REFERENCE DIMENSION (FEET), R0- - - - - F13.5
7/55H   SYSTEM REFERENCE VELOCITY (FT/SEC), V0 - - - - - F13.5
8/55H   X-DIRECTION REFERENCE REYNOLDS NUMBER- - - - - F13.5
9/55H   Z-DIRECTION REFERENCE REYNOLDS NUMBER- - - - - F13.5
A/55H   REFERENCE TEMPERATURE (COLD), DEGREES F- - - - - F13.5
B/55H   REFERENCE TEMPERATURE (HOT ), DEGREES F- - - - - F13.5
C/55H   X-DIRECTION PRANDTL NUMBER - - - - - F13.5
0)
102 FORMAT(
0 55H   Z-DIRECTION PRANDTL NUMBER - - - - - F13.5
E/55H   G = ZERO REFERENCE (CONCENTRATION) - - - - - F13.5
F/55H   G = SOURCE REFERENCE (CONCENTRATION) - - - - - F13.5
G/55H   OUTER LOOP ACCELERATION FACTOR - - - - - F13.5
H/55H   INNER LOOP ACCELERATION FACTOR - - - - - F13.5
K)
1004 FORMAT(///* TOTAL NUMBER OF INTERIOR NODES = * I5//)
1005 FORMAT(1P6(E13.3,1X))
1006 FORMAT(I3,F7.2,2F10.3,1P5E11.3,E13.5)
1007 FORMAT(I9, 9XF4.3,4H TO   F4.0,F7.0, 4H TO   F4.0,1P5(E15.3))
1008 FORMAT(1H1//* INTERNODE CONDUCTANCE * //)

```

```

1009 FORMAT(//@      J = @ 10(I8,4X)//)
1010 FORMAT(1H1,@      DATE @A10,@      TIME @A10/
1 60H      COMPUTED CONSTANTS FOR X-DIRECTION DIFFERENCES-SC(J,L)//
2 30H J   DX(J)   XC(J)   X(J)           ,8(I6,5X)//)
1012 FORMAT(13,F7.3,2F10.3,1P3E11.3)
1013 FORMAT(/)
1014 FORMAT(1H1,@      DATE @A10,@      TIME @A10/
1 60H      COMPUTED CONSTANTS FOR Z-DIRECTION DIFFERENCES-SZ(K,L)//
2 30H K   OZ(K)   Z/O0   Z(K)           ,8(I6,5X)//)
1017 FORMAT(/5H K = 12,2X1P10(E12.3))
1018 FORMAT(9X 1P10(E12.3))
      RETURN
      END
C      SUBROUTINE STREAM (IT,NSKIP)
      SUBROUTINE CALCULATES THE TWO DIMENSIONAL STREAM FUNCTION, PSI(J
      LOGICAL      CONTRL
      INTEGER      OUT,SIND
      EQUIVALENCE(VREF,V0)
      COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1      GAM(62,32),J(62,32),R(62,32),
2      SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3      T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4      T6(62,32),QS(150),PSTART(5),
5      R05(32),TEM(10),ZC(32),N3DPT(5),
6      DMYR,TIM,FLABEL(8),UB(302),VB(300),
7      CONTRL(15),OR(62),XPLOT(300),ZPLGT(300)
      COMMON /2/ NOX(62),NRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1      MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2      INDEX( 50,8),NSTART(10),SIND(70,4),NPEN(300)
      COMMON /3/ OX,OX2,OZC,R0,V0,RRR,RRZ,RRP,ZRP,EXR,EXS,EXT,
1      OPMAX,ODMAX,ODMAX,TINT,START,F0,
2      TREF,TZERO,GREF,GZERO,DMULT,HZERO,DELTR,DZERO,
3      DREF,VBOMB,EZ0,ER0
      COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1      ITMAX,ITNU,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2      ITEMP,NED,ITN00,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
      COMMON /5/ INDU,INDV,IN08,NREF,NSPSI,ISLIP,LINES
      DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
      INDU      = INDU
      INDV      = INDV
      OMEGA3= 0.
      IN08      = IN08
10 DO 120 I = 1,IT
      IF(IN08.EQ.0) GO TO 20
      DO 5 N = 1,IN08
      J      = INDEX(N,1)
      K      = INDEX(N,2)
      JM     = INDEX(N,3)
      KM     = INDEX(N,4)
      JM1    = INDEX(N,5)
      KM1    = INDEX(N,6)
      PSI(J,K) = 2.*PSI(JM,KM)-PSI(JM1,KM1)
      5 CONTINUE
20 CONTINUE

```



```

      DPMAX      = 0.
      DO 100 J = 1,NJ
      A3         = SC(J,13)
      DO 100 K = 1,NK
      N          = NT(J,K)
      IF(N.GT.5) GO TO 100
      CON        = SC(J,5)+SZ(K,5)
      PSIO       = PSI(J,K)
      IF(NSKIP.EQ.0) GO TO 50
      OMEGA3     = OMEG(J,K)*DZ(K)*DR(J)+OMEG(J+1,K)*DZ(K)*DR(J+1)+
1      OMEG(J,K+1)*DZ(K+1)*DR(J)+OMEG(J+1,K+1)*DZ(K+1)*DR(J+1)
      OMEGA3     = OMEGA3/(DZ(K)*DR(J)+DZ(K)*DR(J+1)+DZ(K+1)*DR(J)+
1      DZ(K+1)*DR(J+1))
50  PSI(J,K)    = (SC(J,6)*PSI(J-1,K)+SC(J,7)*PSI(J+1,K)+
1      SZ(K,6)*PSI(J,K-1)+SZ(K,7)*PSI(J,K+1)+OMEGA3)/CON
      DEL        = ABS((PSIO-PSI(J,K))/PSI(J,K))
      DEL        = DEL*PSTART(1)
      DPMAX     = AMAX1(DPMAX,DEL)
      IF(DPMAX.GT.DEL) GO TO 95
      NODE(5)    = J
      NODE(6)    = K
95  PSI(J,K)    = PSIO+EXT*(PSI(J,K)-PSIO)
100 CONTINUE
      IF(DPMAX.LE..00001) GO TO 130
120 CONTINUE
130 IF(NSKIP.EQ.1) GO TO 150
      ITNO       = I
C   CALCULATE VELOCITY FIELD
150 DO 250 J = 2,NJ
      DO 250 K = 2,NK
      IF(NC(J,K).GT.20.OR.NC(J,K).LT.0) GO TO 250
      UX(J,K)    = -(PSI(J,K)-PSI(J,K-1))/DZ(K)
      UZ(J,K)    = (PSI(J,K)-PSI(J-1,K))/DR(J)
250 CONTINUE
      DO 300 J = 1,NJ
      DO 300 K = 1,NK
      IF(NC(J,K).NE.10) GO TO 300
      DO 280 M = 1,4
      JM         = MAP(1,M)+J
      KM         = MAP(2,M)+K
      N          = NC(JM,KM)
      IF(N.EQ.0.OR.N.EQ.10) GO TO 280
      IF(M.LE.2) UZ(JM,KM) = -UZ(J,K)*OR(JM)/DR(J)
      IF(M.GT.2) UX(JM,KM) = -UX(J,K)*DZ(KM)/DZ(K)
280 CONTINUE
300 CONTINUE
      IF(INDB.EQ.0) GO TO 307
      DO 305 N = 1,INDB
      J          = INDEX(N,7)
      K          = INDEX(N,8)
      JM         = INDEX(N,1)
      KM         = INDEX(N,2)
      JM1        = INDEX(N,3)
      KM1        = INDEX(N,4)

```

```

      UX(J,K) = 2.*UX(JM,KM)-UX(JM1,KM1)
      UZ(J,K) = 2.*UZ(JM,KM)-UZ(JM1,KM1)
305 CONTINUE
307 CONTINUE
      DO 310 N = 1,INDU
      J      = JINDU(N)
      K      = KINDU(N)
      IF(NT(J,K).EQ.40) GO TO 310
      UX(J,K) = UB(N)
310 CONTINUE
      DO 320 N = 1,INOV
      J      = JINOV(N)
      K      = KINDV(N)
      IF(NT(J,K).EQ.40) GO TO 320
      UZ(J,K) = VB(N)
320 CONTINUE
      CALL TIME(TIM)
      IF(NSKIP.NE.0) RETURN
      TINT    = TIM-START
      IF(NSKIP.EQ.0) CALL OUTPUT(0)
      RETURN
1000 FORMAT(15,E12.3)
      END
      SUBROUTINE SSCOMP
      DIMENSION AMON(15)
      LOGICAL    CONTRL
      INTEGER     OUT,SIND
      EQUIVALENCE(VREF,V2)
      LOGICAL     DONE,BOMBED
      COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1      GAM(62,32),Q(62,32),G(62,32),
2      SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3      T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4      T6(62,32),QS(150),PSTART(5),
5      R05(32),TEM(10),ZC(32),N3DPT(5),
6      OMYR,TIM,TLABEL(8),UB(300),VB(300),
7      CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
      COMMON /2/ NOX(62),NRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1      MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2      INDEX( 50,8),NSTART(10),SIND(70,4),NPEN(300)
      COMMON /3/ OX,OX2,OZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1      OPMAX,DOMAX,BOMAX,TINT,START,F0,
2      TREF,TZERO,GREF,GZERO,DMULT,HZERO,DELTR,DZERO,
3      DREF,VBOMB,EZ0,ERO
      COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1      ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2      ITEMP,NED,ITNOO,MONIT,NVT,NX,NPCH,NEDUY,NOTAPE,INTER
      COMMON /5/ INDU,INOV,INDB,VREF,NSPSI,ISLIP,LINES
      DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/

C
      DONE      = .FALSE.
      BOMBED    = .FALSE.
      DOMAX     = 1.E-4
      N2        = ITMAX

```

```

ITNO      = ITNOO
ITERS     = NPI
N1        = 1
IF(.NOT.CONTRL(9)) GO TO 15
N1        = ITNO+1
ITMAX     = ITNO+ITMAX
N2        = ITMAX
NED       = NED + ITNO
IF(NED.GT.ITNO.OR,NEEDY.EQ.0) GO TO 15
CALL EDDY(NEEDY)
15 CALL TIME(START)
DO 300 L = N1,N2
ITNO      = L
IF(CONTRL(3).OR.CONTRL(4)) CALL TRSPRT
IF(DOMAX.LE.1.E-5) DONE = .TRUE.
IF(CONTRL(13)) GO TO 400
CALL ROTATE
CALL STREAM(ITERS,1)
CALL BOUSET(1)
400 CONTINUE
IF(.NOT.CONTRL(5)) GO TO 450
JK        = 0
IF(CONTRL(13)) GO TO 410
DO 405 KK = 1,10,2
L1        = MON(KK)
L2        = MON(KK+1)
L3        = MON(KK+10)
L4        = MON(KK+11)
L5        = MON(KK+20)
L6        = MON(KK+21)
JK        = JK+1
AMON(JK)  = UZ(L1,L2)
AMON(JK+5) = UX(L3,L4)
AMON(JK+10) = TEMP(L5,L6)
IF(AMON(JK).GT.VBOMB) BOMBED = .TRUE.
IF(AMON(JK+5).GT.VBOMB) BOMBED = .TRUE.
IF(AMON(JK+10).GT.10.*TZERO) BOMBED = .TRUE.
405 CONTINUE
WRITE(OUT,1000) ITNO,(AMON(KK),KK= 1,15)
GO TO 450
410 CONTINUE
DO 415 KK = 1,10,2
L5        = MON(KK+20)
L6        = MON(KK+21)
JK        = JK+1
AMON(JK)  = TEMP(L5,L6)
IF(AMON(JK).GT.10.*TZERO) BOMBED = .TRUE.
415 CONTINUE
WRITE(OUT,1001) ITNO,(AMON(KK),KK = 1,5)
450 CONTINUE
IF(BOMBED) GO TO 950
IF(.NOT.CONTRL(12)) GO TO 750
IF(NED.GE.L) GO TO 750
CALL EDDY(NEEDY)

```

```

750 IF(MOD(L,NOUT1.EQ.0.OR.DONE ) CALL OUTPUT(1)
    IF(MOD(L,NTTY).EQ.0) CALL OUTPUT (2)
    IF(DONE) GO TO 805
800 CONTINUE
805 CONTINUE
    IF(.NOT.CONTROL(14)) GO TO 808
    QTOT = 3.2
    AREA = 0.
    DO 807 J=1,NJ
    DO 807 K=1,NK
    Q(J,K) = Q(J,K)*SFLUX(TEMP(J,K),TZERO,QS)*62.4*1.0*3600.
    QTOT = QTOT+Q(J,K)
    IF(NC(J,K).EQ.50) GO TO 807
    AREA = AREA+DK(J)*DZ(K)*R0*R0
    ACRES = AREA/43560.
807 CONTINUE
    QMW = QTOT/3.413E6
    WRITE(6,1209) QTOT,QMW,AREA,ACRES
808 CONTINUE
    IF(CONTROL(7)) CALL PCHCRD(T1,T2,T3,T4)
    IF(.NOT.CONTROL(6)) RETURN
    WRITE(8) ITNO,OMEG,TEMP,UX,UZ,PSI,GAM
    ENTRY WRITEP
    IF(.NOT.CONTROL(6)) RETURN
    N3DPTS = 0
    DO 810 J = 1,5
    IF(N3DPT(J).EQ.0) GO TO 810
    N3DPTS = N3DPTS+1
810 CONTINUE
    NX = NL
    NZ = NH
    IF(N3DPTS.EQ.0) GO TO 830
    WRITE(8) KASE,OMYR,TIM,TLABEL,N3DPTS,NX,NZ,LINES,
1      (NPEN(N),N=1,LINES),
2      (XPLOT(N),N=1,LINES),
3      (ZPLOT(N),N=1,LINES)
    DO 820 J = 1,5
    L = N3DPT(J)
    IF(L.EQ.0) GO TO 820
    IF(L.EQ.1) WRITE(8) L,(R(N),N=1,NX),(Z(N),N=1,NZ),
*      ((NT(N,M),N=1,NX),M=1,NZ),
1      ((PSI(N,M),N=1,NX),M=1,NZ)
    IF(L.EQ.3) WRITE(8) L,(RC(N),N=1,NX),(ZC(N),N=1,NH),
1      ((NC(N,M),N=1,NX),M=1,NZ),
2      ((GAM(N,M),N=1,NX),M=1,NZ)
    IF(L.EQ.4) WRITE(8) L,(RC(N),N=1,NX),(ZC(N),N=1,NH),
1      ((NC(N,M),N=1,NX),M=1,NZ),
2      ((TEMP(N,M),N=1,NX),M=1,NZ)
    IF(L.EQ.5) WRITE(8) L,(RC(N),N=1,NX),(ZC(N),N=1,NH),
1      ((NC(N,M),N=1,NX),M=1,NZ),
2      ((OMEG(N,M),N=1,NX),M=1,NZ)
    IF(L.EQ.1) WRITE(OUT,1204) L
    IF(L.EQ.2) WRITE(OUT,1205) L
    IF(L.EQ.3) WRITE(OUT,1206) L

```

```

      IF(L.EQ.4) WRITE(OUT,1007) L
      IF(L.EQ.5) WRITE(OUT,1008) L
820  CONTINUE
830  CONTINUE
      WRITE(OUT,1003) N3OPTS,NX,NZ
      RETURN
950  CONTINUE
      WRITE(OUT,1010)
      CALL OUTPUT(1)
      RETURN
1000 FORMAT(I6,3X15(F7.3,1X))
1001 FORMAT(I6,63X5F7.3)
1002 FORMAT(I10,2(12XF10.1))
1003 FORMAT(///5X15, ' THREE-D PLOT RECORDS WRITTEN ON TAPE'/
1      ' SET PLOT PARAMETERS NJ = #I3, # NK = #I3)
1004 FORMAT(' STREAM FUNCTION RECORD WRITTEN TO TAPE - RECORD NO #I3)
1005 FORMAT(' BUOYANCY PARAMETER RECORD WRITTEN TO TAPE - RECORD NO #I3)
1006 FORMAT(' GAMMA-CONSTITUENT RECORD WRITTEN TO TAPE - RECORD NO #I3)
1007 FORMAT(' TEMPERATURE RECORD WRITTEN TO TAPE - RECORD NO #I3)
1008 FORMAT(' VORTICITY RECORD WRITTEN TO TAPE - RECORD NO #I3)
1009 FORMAT(///' TOTAL SURFACE HEAT FLUX'/
1      ' Q-SURFACE = #,E10.3,# BTU/HR'/
2      ' Q-SURFACE = #,F10.0,# MW(E)'/
3      ' SURFACE AREA = #,E10.3,# SQ FT'/
4      ' SURFACE AREA = #,F10.2,# ACRES'///)
1010 FORMAT(1H1///5X 'MONITOR CELLS INDICATE RUN IS UNSTABLE'
1//5X 'RUN ABORTED')
      END
      SUBROUTINE ROTATE
      LOGICAL CTRL
      INTEGER OUT,SIN
      EQUIVALENCE(VREF,V0)
      COMMON /1/ QMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1      GAM(62,32),Q(62,32),G(62,32),
2      SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3      T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4      T6(62,32),QS(150),PSTART(5),
5      R05(32),TEM(10),ZC(32),N3OPT(5),
6      DMYR,TIM,TLABEL(8),UB(300),VB(300),
7      CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
      COMMON /2/ NOX(62),WRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1      MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2      INDEX(50,8),NSTART(10),SINO(70,4),NPEN(300)
      COMMON /3/ DX,DY2,DZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1      DPMAX,DOMAX,DOMAX,TINT,START,F0,
2      TREF,TZERO,GREF,GZERO,OMULT,HZERO,DELTR,OZERO,
3      DREF,VBOMB,EZ0,ER0
      COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1      ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2      ITEM,PED,ITNOO,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
      COMMON /5/ INDU,INDV,IND8,NREF,NPSI,ISLIP,LINES
      DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
      DOMAX = 0.
      DOMEQ = 0.

```

```

DO 300 J = 2,NJ
MW 311 0 = =V0
IF(NC(J,K).NE.0) GO TO 300
DO 300 K = 2,NK
IF(NC(J,K).NE.0) GO TO 300
A1      = SC(J,1)*FR(J,K-1)
A2      = SC(J,2)*FR(J,K-1)
A3      = SC(J,3)*FR(J,K-1)
A4      = SC(J,4)
UPOS1   = ABS(UX(J-1,K)) + UX(J-1,K)
UPOS2   = ABS(UX(J,K)) + UX(J,K)
UNEG1   = ABS(UX(J-1,K)) - UX(J-1,K)
UNEG2   = ABS(UX(J,K)) - UX(J,K)
VPOS1   = ABS(UZ(J,K-1)) + UZ(J,K-1)
VPOS2   = ABS(UZ(J,K)) + UZ(J,K)
VNEG1   = ABS(UZ(J,K-1)) - UZ(J,K-1)
VNEG2   = ABS(UZ(J,K)) - UZ(J,K)
B0      = FZ(J,K)*SZ(K,1)
B1      = 1./ (B0+A1+A4*(UPOS2+UNEG1)+SZ(K,4)*(VPOS2+VNEG1))
AJ1     = (A2+A4*(UPOS1))*OMEG(J-1,K)
AJ2     = (A3+A4*(UNEG2))*OMEG(J+1,K)
AK1     = (FZ(J,K)*SZ(K,2)+SZ(K,4)*VPOS1)*OMEG(J,K-1)
AK2     = (FZ(J,K)*SZ(K,3)+SZ(K,4)*VNEG2)*OMEG(J,K+1)
OMEGO   = OMEG(J,K)
OMEG(J,K) = (AJ1+AJ2+AK1+AK2+OMEGO)*B1
IF(OMEG(J,K).EQ.0.) GO TO 295
DEL      = ABS((OMEG(J,K)-OMEGO)/OMEG(J,K))
DOMAX    = AMAX1(DOMAX,DEL)
IF(DOMAX.GT.DEL) GO TO 295
NODE(3)  = J
NODE(4)  = K
295 OMEG(J,K) = OMEGO+EXR*(OMEG(J,K)-OMEGO)
300 CONTINUE
RETURN
END
SUBROUTINE TRSPRT
LOGICAL   CONTRL
INTEGER   OUT,SIND
EQUIVALENCE(VREF,V0)
COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1          GAM(62,32),G(62,32),G(62,32),
2          SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3          T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4          T6(62,32),OS(150),PSTART(5),
5          R05(32),TEM(10),ZC(32),N3OPT(5),
6          DMYR,TIM,TLABEL(8),UB(300),VB(300),
7          CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
COMMON /2/ NOX(62),NRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1          MAP(2,4),JINDU(300),KINDU(300),KINOV(300),JINOV(300),
2          INDEX( 50,8),NSTART(10),SIND(70,4),NPEN(300)
COMMON /3/ OX,OX2,OZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1          DPMAX,DOMAX,DDMAX,TINT,START,F0,
2          TREF,TZERO,GREF,GZERO,OMULT,HZERO,DELTR,OZERO,
3          DREF,VBOMB,EZ0,ER0

```

```

COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1          ITMAX,ITNO,ITAE,INP,NTTY,NK,NKK,NPT,KS,JS,
2          ITEMP,NEO,ITNOO,MONIT,NVT,NX,NPCH,NEOY,NOTAPE,INTER
COMMON /5/ INOU,INDV,IND8,NREF,NPSI,ISLIP,LINES
DATA LJ,LK,L3V,L40,L20/62,32,300,50,70/
DDMAX      = 0.
OGMAX      = 0.
DO 200 J = 2,NJ
DO 200 K = 2,NK
IF(NC(J,K).GT.10) GO TO 200
TEMPO      = TEMP(J,K)
GAMAO      = GAM (J,K)
DIASUM     = T1(J,K)+T2(J,K)+T3(J,K)+T4(J,K)
SUML       = T1(J,K)
SUMR       = T2(J,K)
SUMU       = T4(J,K)
SUMD       = T3(J,K)
EXC        = EXR
UPOS1      = ABS(UX(J-1,K))+ UX(J-1,K)
UPOS2      = ABS(UX(J,K)) + UX(J,K)
UNEG1      = ABS(UX(J-1,K))- UX(J-1,K)
UNEG2      = ABS(UX(J,K)) - UX(J,K)
VPOS1      = ABS(UZ(J,K-1))+UZ(J,K-1)
VPOS2      = ABS(UZ(J,K)) + UZ(J,K)
VNEG1      = ABS(UZ(J,K-1))-UZ(J,K-1)
VNEG2      = ABS(UZ(J,K)) - UZ(J,K)
DIASUM     = T5(J,K)*(UPOS2+UNEG1)+
*          T6(J,K)*(VPOS2+VNEG1)+ DIASUM
SUML       = SUML+T5(J,K)*UPOS1
SUMR       = SUMR+T5(J,K)*UNEG2
SUMD       = SUMD +T6(J,K)*          VPOS1
SUMU       = SUMU +T6(J,K)*          VNEG2
10 CONTINUE
QSURF      = 0.
IF(CONTRL(14)) QSURF = Q(J,K)*SFLUX(TEMP(J,K),TZERO,QS)
TEMP(J,K) = (SUML*TEMP(J-1,K)+SUMR*TEMP(J+1,K)+
*          SUMD*TEMP(J,K-1)+SUMU*TEMP(J,K+1)+QSURF)/DIASUM
GAM (J,K) = (SUML*GAM (J-1,K)+SUMR*GAM (J+1,K)+
*          SUMD*GAM (J,K-1)+SUMU*GAM (J,K+1))/DIASUM
DEL        = ABS((TEMP(J,K)-TEMPO)/TEMP(J,K))
DDMAX      = AMAX1(DDMAX,DEL)
IF(DDMAX.GT.DEL) GO TO 50
NODE(1)    = J
NODE(2)    = K
50 TEMP(J,K)=TEMPO+EXC*(TEMP(J,K)-TEMPO)
GAM (J,K)= GAMAO+EXR*(GAM (J,K)-GAMAO)
200 CONTINUE
RETURN
END
SUBROUTINE BOUSET(JUMP)
LOGICAL     CONTRL
INTEGER     OUT,SIND
EQUIVALENCE(VREF,V0)
COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),

```

```

1      GAM(62,32),Q(62,32),G(62,32),
2      SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3      T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4      T6(62,32),QS(150),PSTART(5),
5      R05(32),TEM(10),ZC(32),N3OPT(5),
6      DMYR,TIM,TLABEL(6),UB(300),VB(300),
7      CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
COMMON /2/ NOX(62),NWRITE(15),NDOE(6),MON(30),NT(62,32),NC(62,32),
1      MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2      INDEX(50,8),NSTART(10),SIND(70,4),NPEN(300)
COMMON /3/ DX,DX2,OZC,R0,V0,RRR,RRZ,RRP,ZRP,EXR,EXS,EXT,
1      DPMAX,DOMAX,ODMAX,TINT,START,F0,
2      TREF,TZERO,GREF,GZERO,OMULT,HZERO,DELTR,DZERO,
3      OREF,VBOMB,EZ0,ER0
COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1      ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2      ITEMP,NED,IINQO,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
COMMON /5/ INDU,INDV,INDB,NREF,NSPSI,ISLIP,LINES
DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
GO TO (100,200,300), JUMP
100 CONTINUE
DO 150 J = 2,NJ
DO 150 K = 2,NK
IF(NC(J,K).NE.10) GO TO 150
DUZ      = .5*(SZ(K,9)*(UX(J,K+1)+UX(J-1,K+1))-
*          SZ(K,8)*(UX(J,K-1)+UX(J-1,K-1))+
*          SZ(K,10)*(UX(J,K)+UX(J-1,K)))
DVR      = .5*(SC(J,9)*(UZ(J+1,K)+UZ(J+1,K-1))-
*          SC(J,8)*(UZ(J-1,K)+UZ(J-1,K-1))+
*          SC(J,10)*(UZ(J,K)+UZ(J,K-1)))
OMEG(J,K)= DUZ-DVR
150 CONTINUE
IF(ISLIP.LT.1) GO TO 195
DO 190 M = 1,ISLIP
J1      = SIND(M,1)
K1      = SIND(M,2)
J2      = SIND(M,3)
K2      = SIND(M,4)
UX(J1,K1) = UX(J2,K2)
UZ(J1,K1) = UZ(J2,K2)
OMEG(J1,K1) = -OMEG(J2,K2)
TEMP(J1,K1) = TEMP(J2,K2)
190 CONTINUE
195 IF(INDB.EQ.0) RETURN
DO 155 M = 1,INDB
J      = INDEX(M,7)
K      = INDEX(M,8)
JM     = INDEX(M,1)
KM     = INDEX(M,2)
JM1    = INDEX(M,3)
KM1    = INDEX(M,4)
OMEG(J,K) = 2.*OMEG(JM,KM)-OMEG(JM1,KM1)
IF(J.GT.JM.AND.UX(J,K).LT.0.) OMEG(J,K) = 0.
IF(J.LT.JM.AND.UX(J,K).GT.0.) OMEG(J,K) = 0.

```



```

      IF(K.GT.KM.AND.UZ(J,K).LT.0.) OMEG(J,K) = 0.
      IF(K.LT.KM.AND.UZ(J,K).GT.0.) OMEG(J,K) = 0.
155 CONTINUE
160 CONTINUE
      RETURN
200 CONTINUE
      RETURN
300 CONTINUE
      RETURN
      END
      SUBROUTINE AROUT(N,VCOORD,RCOORD,ARNAME,LABEL)
      DIMENSION ARNAME(62,32),LABEL(4),RCOORD(62),VCOORD(32)
      LOGICAL CTRL
      INTEGER OUT,SIND
      EQUIVALENCE(VREF,V0)
      REAL LABEL
      COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1 GAM(62,32),Q(62,32),G(62,32),
2 SZ(32,15),Z(32),UZ(32),SC(62,15),R(62),RC(62),
3 T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4 T6(62,32),QS(150),PSTART(5),
5 R05(32),TEM(10),ZC(32),N3OPT(5),
6 OMYR,TIM,TLABEL(8),UB(300),VB(300),
7 CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
      COMMON /2/ NOX(62),NRIFE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1 MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2 INDEX(50,8),NSTART(10),SIND(70,4),NPEN(300)
      COMMON /3/ DX,DX2,OZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1 DPMAX,DDMAX,DDMAX,TINT,START,F0,
2 TREF,TZERO,GREF,GZERO,DMULT,HZERO,DELTR,OZERO,
3 OREF,VBOMB,EZ0,ER0
      COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1 ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2 ITEMP,NED,ITNOO,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
      COMMON /5/ INDU,INDV,INDB,NREF,NPSPI,ISLIP,LINES
      DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
      N2 = 0
60 N1 = N2+1
      N2 = N1+9
      IF(N2.GT.NL)N2=NL
      WRITE(OUT,1000) OMYR,TIM,LABEL,ITNO,TINT
      WRITE(OUT,1001) (NOX(K),K=N1,N2)
65 WRITE(OUT,1002) (RCOORD(K),K=N1,N2)
70 DO 200 K = 1,NH
      KN = NH-K+1
150 WRITE(OUT,1003) KN,VCOORD(KN),(ARNAME(J,KN),J=N1,N2)
200 CONTINUE
      IF(N2.NE.NL) GO TO 60
1000 FORMAT(1H1,*, DATE #A10,*, TIME #A10/
1 1 5X4A10,*, ITERATION NUMBER #,IS,
1 25H COMPUTATION SPEED = F6.3,15H SEC/ITERATION )
1001 FORMAT(/17X3HJ = 10(I3,3X))
1002 FORMAT(12X#XCOORD = #,10(F9.2,2X)/12X#ZCOORD#)
1003 FORMAT( 5H K = I2, 5H Z = F6.2,2X1P10E11.3)

```

```

END
SUBROUTINE OUTPUT(MODE)
LOGICAL      CONTRL
INTEGER      OUT,SIND
EQUIVALENCE(VREF,V0)
COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1          GAM(62,32),Q(62,32),G(62,32),
2          SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3          T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4          T6(62,32),QS(152),PSTART(5),
5          R05(32),TEM(10),ZC(32),N3OPT(5),
6          DMVR,TIM,TLABEL(3),UB(300),VB(300),
7          CONTRL(15),OR(62),XPLOT(300),ZPLOT(300)
COMMON /2/ NOX(62),NWRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1          MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2          INDEX(50,8),NSTART(10),SIND(70,4),NPEN(300)
COMMON /3/ OX,OX2,OZC,R0,V0,RRR,REZ,RRP,ZRP,EXR,EXS,EXT,
1          OPMAX,ODMAX,ODMAX,TINT,START,F0,
2          TREF,TZERU,GREF,GZERO,DMULT,HZERO,DELTR,DZERO,
3          DREF,VBOMB,EZ0,ER0
COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1          ITMAX,ITNO,ITAPE,INP,NTTY,NX,NKK,NPT,KS,JS,
2          ITEMP,NEQ,ITNOO,MONIT,NVT,NX,NPCH,NEDDY,NOTAPE,INTER
COMMON /5/ INOU,INDV,INOB,NREF,NSPSI,ISLIP,LINES
DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
IF(MODE.EQ.5) GO TO 5
IF(MODE.NE.2) GO TO 10
WRITE(OUT,1000) ITNO,NPI,OPMAX,NODE(5),NODE(6),
1          ODMAX,NODE(3),NODE(4),ODMAX,NODE(1),NODE(2)
5 WRITE(OUT,1002) (MON(KK),KK = 1,30)
RETURN
10 CALL TIME(TIM)
IF(MODE.EQ.0) WRITE(OUT,1001) ITNO,OPMAX,ITNO,TINT
TINT = (TIM-START)/NOUT
DO 100 J = 1,15
L = NWRITE(J)
IF(L.EQ.0) GO TO 100
IF(MODE.EQ.1) GO TO 90
IF(L.EQ.1) CALL AROUT(L,Z,R,PSI,
1          40HSTREAM FUNCTION - IRROTATIONAL FLOW )
IF(MODE.EQ.2) GO TO 95
90 IF(L.EQ.2) CALL AROUT(L,Z,R,PSI,
1          40HSTREAM FUNCTION - VISCOUS FLOW )
IF(L.EQ.3) CALL AROUT(L,ZC,RC,DELT,
1          40HBUOYANCY PARAMETER - DELT )
IF(L.EQ.4) CALL AROUT(L,ZC,RC,OMEG,
1          40HVORTICITY - OMEG )
95 CONTINUE
IF(L.EQ.5) CALL AROUT(L,Z,RC,UZ,
1          40HZ-DIRECTION VELOCITY COMPONENT - UZ )
IF(L.EQ.6) CALL AROUT(L,ZC,R,UX,
1          40HX-DIRECTION VELOCITY COMPONENT - UX )
IF(MODE.EQ.0) GO TO 100
IF(L.EQ.7) CALL AROUT(L,ZC,RC,GAM,

```

```

1          40HGAMMA-CONSTITUENT )
1 IF(L.EQ.8) CALL AROUT(L,ZC,RC,TEMP, )
1          40HTEMPERATURE, DEGREES FAHRENHEIT )
1 IF(L.EQ.9) CALL AROUT(L,ZC,RC,G, )
1          40HSURFACE HEAT FLUX, BTU/HR )
1 IF(L.EQ.10)CALL AROUT(L,Z,RC,UZ, )
1          40HNORMALIZED VERTICAL VELOCITY COMPONENT )
1 IF(L.EQ.11)CALL AROUT(L,ZC,RC,DELT, )
1          40HNORMALIZED TEMPERATURE DISTRIBUTION )
1 IF(L.EQ.15)GO TO 1003
1 IF(L.EQ.14)CALL AROUT(L,ZC,RC,RICH, )
1          40H RICHARDSON NUMBERS )
1000 CONTINUE
1 CALL TIME(START)
1 RETURN
1000 FORMAT(1H1 26H RESULTS FOR ITER. NO. 15/
1/ 35H NO. OF PSI ITERATIONS- - - - - 17,10X4HNODE
2/ 35H MAX CHANGE IN PSI - - - - - 1PE10.3
*,7H (I2,1H,,I2,1H)
3/ 35H MAX CHANGE IN OMEG - - - - - E10.3
*,7H (I2,1H,,I2,1H)
4/ 35H MAX CHANGE IN DELT - - - - - E10.3
*,7H (I2,1H,,I2,1H) )
1001 FORMAT(1H1 45H STREAM FUNCTION RESULTS FOR ITERATION 15/
1/ 30H MAXIMUM RELATIVE ERROR IS 1PE12.3/
2/ 23H TIME REQUIRED FOR I3,14H ITERATIONS = F6.2,
35H SEC ///)
1002 FORMAT(/ 10H ITERATION
1 40H ..... Z-DIRECTION VELOCITY .....
2 40H ***** X-DIRECTION VELOCITY *****
3 40H ..... TEMPERATURE .....
4/8H NUMBER 15(2H (I2,1H,I2,1H))//)
1003 FORMAT( ///# THIS CASE IS APPARENTLY UNSTABLE, RUN ABORTED #)
END
SUBROUTINE MAPPER(NARRAY,LABEL)
DIMENSION NARRAY(62,32),LABEL(4)
LOGICAL CTRL
INTEGER OUT,SIND
EQUIVALENCE(VREF,V0)
COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1 GAM(62,32),G(62,32),G(62,32),
2 SZ(32,15),Z(32),DZ(32),SC(62,15),R(62),RC(62),
3 T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4 T6(62,32),QS(150),PSTART(5),
5 R05(32),TEM(10),ZC(32),N3DPT(5),
6 DMYR,TIM,TLABEL(8),UB(300),VB(300),
7 CTRL(15),OR(62),XPLOT(300),ZPLOT(300)
COMMON /2/ NOX(62),NRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1 MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2 INDEX( 50,8),NSTART(10),SIND(70,4),NPEN(300)
COMMON /3/ OX,OX2,OZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1 DPMAX,D0MAX,D0MAX,TINT,START,F0,
2 TREF,TZERO,GREF,GZERO,DMULT,HZERO,DELTR,DZERO,
3 DREF,VBOMB,EZ0,ER0

```

```

COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1          ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2          ITEMP,NEO,ITNOO,MONIT,NVT,NX,NPCH,NEODY,NOTAPE,INTER
DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/
N2          = 0
10 N1        = N2+1
N2          = N1+39
IF(N2.GT.NJ) N2 = NJ
WRITE(OUT,1000) LABEL
WRITE(OUT,1001) (NOX(L),L=N1,N2)
DO 20 L = 1,NK
KN          = NK-L+1
20 WRITE(OUT,1002) KN,(NARRAY(JN,KN),JN = N1,N2)
IF(N2.NE.NJ) GO TO 10
RETURN
1000 FORMAT(1H1/5X4A10)
1001 FORMAT(/@ J = @ 5X40I3//)
1002 FORMAT(@ K = @ 13,2X40I3)
END
FUNCTION SFLUX(T,T0,QS)
DIMENSION QS(150)
C* SFLUX CALCULATES SURFACE HEAT TRANSFER IN BTU/FT2
N1          = T
N2          = N1+1
SFLUX       = QS(N1)+(QS(N2)-QS(N1))*(T-N1)
IF(N1.LT.0) SFLUX = 0.0
IF(N2.GT.150) SFLUX = 0.0
RETURN
END
SUBROUTINE PCHCRD(UAV,VAV,XFEET,YFEET)
DIMENSION UAV(2000),VAV(2000),XFEET(2000),YFEET(2000)
LOGICAL CTRL
INTEGER OUT,SIN0
EQUIVALENCE(VREF,V0)
COMMON /1/ OMEG(62,32),PSI(62,32),TEMP(62,32),UX(62,32),UZ(62,32),
1          GAM(62,32),Q(62,32),G(62,32),
2          SZ(32,15),Z(32),OZ(32),SC(62,15),R(62),RC(62),
3          T1(62,32),T2(62,32),T3(62,32),T4(62,32),T5(62,32),
4          T6(62,32),QS(150),PSTART(5),
5          R05(32),TEM(10),ZC(32),N3DPT(5),
6          DMYR,TIM,TLABEL(8),UB(300),VB(300),
7          CTRL(15),OR(62),XPLOT(300),ZPLUT(300)
COMMON /2/ NOX(62),NWRITE(15),NODE(6),MON(30),NT(62,32),NC(62,32),
1          MAP(2,4),JINDU(300),KINDU(300),KINDV(300),JINDV(300),
2          INDEX(50,8),NSTART(10),SIND(70,4),NPEN(300)
COMMON /3/ DX,DX2,OZC,R0,V0,RER,REZ,RRP,ZRP,EXR,EXS,EXT,
1          OPMAX,DDMAX,DDMAX,TINT,START,F0,
2          TREF,TZERO,GREF,GZERO,DMULT,HZERO,DELTR,DZERO,
3          DREF,VBOMB,EZ0,ER0
COMMON /4/ NJ,NJJ,NH,NL,NOUT,NPI,IN,OUT,IPMAX,KASE,KT,
1          ITMAX,ITNO,ITAPE,INP,NTTY,NK,NKK,NPT,KS,JS,
2          ITEMP,NEO,ITNOO,MONIT,NVT,NX,NPCH,NEODY,NOTAPE,INTER
COMMON /5/ INOU,INDV,INDB,NREF,NPSPI,ISLIP,LINES
DATA LJ,LK,LBV,L40,L20/62,32,300,50,70/

```

```

      KOUNT      = 0
      DO 90 J = 1,NJ
        T5(J,1) = 0.
        T6(J,1) = 0.
90    CONTINUE
      DO 95 K = 1,NK
        T5(1,K) = 0.
        T6(1,K) = 0.
95    CONTINUE
      DO 100 K = 2,NK
        DO 100 J = 2,NJ
          T5(J,K) = 0.
          T6(J,K) = 0.
          IF(NC(J,K).EQ.50) GO TO 100
          KOUNT = KOUNT+1
          T5(J,K) = .5*(UX(J,K)+UX(J-1,K))*V0
          T6(J,K) = .5*(UZ(J,K)+UZ(J,K-1))*V0
          VAV(KOUNT) = T6(J,K)
          UAV(KOUNT) = T5(J,K)
          YFEET(KOUNT) = ZC(K)*R0
          XFEET(KOUNT) = RC(J)*R0
100   CONTINUE
      NCARDS = (KOUNT+1)/2
      PUNCH 1000,((XFEET(I),YFEET(I),UAV(I),VAV(I)),I=1,KOUNT)
      WRITE(6,1001) NCARDS,KOUNT
      CALL AROUT(1,ZC,RC,T5,
1      40HLATERAL VELOCITIES - - CELL CENTERED      )
      CALL AROUT(1,ZC,RC,T6,
1      40HVERTICAL VELOCITIES - - CELL CENTERED      )
      RETURN
1000 FORMAT(2(2F9.0,2F9.5))
1001 FORMAT(///# CARDS PUNCHED =# I5,# FOR #I5,# DATA POINTS#)
END
C
      FUNCTION MOD(N,M)
      RETURNS ZERO WHENEVER N IS EVENLY DIVISIBLE BY M
      MOD = N-(N/M)*M
      END
      SUBROUTINE EDOY(M)
      RETURN
      END
      FUNCTION FR(J,K)
      FR = 1.
      RETURN
      END
      FUNCTION FZ(J,K)
      FZ = 1.
      RETURN
      END

```

DISTRIBUTION

No. of
Copies

No. of
Copies

OFFSITE

A. A. Churm
ERDA Chicago Patent Group
Chicago Operations Office
9800 South Cass Avenue
Argonne, IL 60439

3 M. Jinks
U.S. Nuclear Regulatory Commission
Central Mail and Files
Washington, DC 20555

27 ERDA Technical Information
Center

J. B. Blanton
Energy Research and Development
Administration
Division of Biomedical and
Environmental Research
Environmental Programs
Washington, DC 20545

J. L. Liverman
Assistant Administrator for
Environment and Safety
Energy Research and
Development Administration
Washington, DC 20545

R. Overstreet
Hydrologic Engineering Section,
Site Analysis Branch
Division of Technical Review
U.S. Nuclear Regulatory Commission
Washington, DC 20555

R. Codell
Hydrologic Engineering
Section, Site Analysis Branch
Division of Technical Review
U.S. Nuclear Regulatory
Commission
Washington, DC 20555

D. Schrieber
Hydrologic Engineering
Section, Site Analysis Branch
Division of Technical Review
U.S. Nuclear Regulatory
Commission
Washington, DC 20555

G. Hulman
Hydrologic Engineering
Section, Site Analysis
Branch
Division of Technical Review
U.S. Nuclear Regulatory
Commission
Washington, DC 20555

A. Policastro
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439

G. Marmer
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439

ONSITE

2 ERDA Richland Operations
Office, Program Division

P. G. Holsted
B. J. Melton

63 Battelle-Northwest

D. W. Dragnich
R. F. Foster
L. Kannberg
W. W. Laity
D. E. Olesen

No. of
Copies

Y. Onishi
A. M. Sutey
D. S. Trent (50)
B. Vaughan
C. L. Wheeler
Technical Information (3)
Technical Publications