TR-45
1972

Texas Water Resources Institute
make every drop count

Mathematical Models of Water Quality Parameters for Rivers and Estuaries

R.W. Hann, Jr.
P.J. Young

Texas Water Resources Institute

Texas A&M University

RESEARCH PROJECT COMPLETION REPORT

Project Number A-014-TEX

October 1, 1969 -- September 30, 1971

Agreement Numbers
14-31-0001-3044
14-31-0001-3244
14-31-0001-3544

MATHEMATICAL MODELS OF WATER QUALITY PARAMETERS FOR RIVERS AND ESTUARIES

Principal Investigators
Roy W. Hann, Jr.
Paul Jonathan Young

The work upon which this publication is based was supported in part by funds provided by the United States Department of the Interior, Office of Water Resources Research, as authorized under the Water Resources Research Act of 1964.

Technical Report No. 45
Water Resources Institute
Texas A&M University

October 1972

FOREWARD

The development of computer models for mass transport in estuaries has been an important engineering activity for the past decade. Initial surveys conducted under this project showed however that only a limited amount of work had been done in modeling two dimensional characteristics in partially stratified estuaries.

As a result the major thrust of this project was directed in this important area. The project effort was coodinated with an existing project dealing with the Houston Ship Channel specifically EPA Research and Development grant 16090 DQW and considerably field data was developed by this EPA project. Additional support was received in the form of a Miles Cox fellowship which partially supported Dr. Young.

This final repot presents the results of the literature survey model development and model calibration.

Project personnel over project time period have included Dr. Roy W. Hann, Jr., the current project director, Dr. Donald Schaezler, Dr. Robert Irvine, Dr. P. Jonathan Young, Mr. Richard Withers, Dr. Richard Allison and other staff members of the Estuarine Systems Projects Research group of The Environmental Engineering Division of Texas A&M Universities Civil Engineering Department.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF TABLES
(Continued)

LIST OF FIGURES

LIST OF FIGURES
(Continued)

CHAPTER I

INTRODUCTION AND GOALS

Estuaries represent an economically and biologically important part of our coast. These bodies of water are used as permanent homes or nursery grounds for many of the important forms of marine life. Estuaries also provide protected transportation routes for cargo ships and barges. Because of their biological productivity and their economically advantageous locations, estuaries often attract industrial and commercial development along with associated dense populations.

Because of industrial and metropolitan development, estuaries are often expected to accept large amounts of waste materials. Many estuaries are large in size and give the appearance of being able to assimilate large volumes of this waste material. In actuality, estuaries may act as natural traps for pollutants. The fresh-water flow into some estuaries is relatively small. Once a pollutant enters such an estuary, the material is carried back and forth by the tides and makes only slow progress toward the ocean. These conditions are especially characteristic of Gulf

Coast estuaries. For example, recent studies (92) indicate that the average flushing time (i.e., the time necessary for the volume of fresh water inflow to equal the volume of the estuary) for the upper 14 miles of the Houston Ship Channel is 26 days and the average flushing time for Galveston Bay is 155 days.

In regard to salinity, estuaries vary between two extremes: the homogeneous estuary and the highly stratified. At each section of a homogeneous estuary, there is complete vertical mixing of the salt and fresh water. In a stratified estuary, there are two layers of water: the upper layer contains fresh water flowing toward the sea, and the lower layer contains salt water moving away from the sea. A sharp change in salinity occurs at the interface between the upper and lower layers. The Delaware Estuary is representative of a homogeneous estuary, whereas the Mississippi Delta represents a highly stratified estuary. Many estuaries fall somewhere between these two extremes and include some mixing and some stratification at each section; hence, they are more difficult to analyze than the idealized homogeneous or stratified cases. The Houston Ship Channel is an example of a partially stratified estuary. A comprehensive discussion of the characteristics of these types of estuaries is available in Ippen (51).

GOALS OF THIS STUDY

Although the past decade has produced an extensive amount of mathematical modeling of estuaries, little effort has gone into modeling of the two dimensional characteristics of partially stratified estuaries. A major goal of this research was to develop computer models which could calculate vertical and horizontal mass transport in partially stratified estuaries. These models were to be applicable for predicting pollutant dispersion and dissolved oxygen distributions.

Two types of finite difference techniques provided the basis for these mathematical models: an explicit method and a Crank-Nicolson implicit method. The second major goal of this research was to compare the accuracy and usefulness of these two techniques. Discussions in literature of this type of comparison are scanty. Where these comparisons do exist for the two-dimensional mass transport equation (28), the equation has been simplified and applied to non-estuary problems with limited success.

A third major goal of this research was to summarize existing one- and two-dimensional mathematical models that have been applied to significant estuary problems.

The final major goal of this study was to demonstrate the applicability of these computer models to the mass transport characteristics of the Houston Ship Channel.

CHAPTER II

CHARACTERISTICS OF THE HOUSTON SHIP CHANNEL

The portion of the Houston Ship Channel being studied in this report is a tributary to upper Galveston Bay (see Figure 2-1). The Houston Ship Channel, shown in Figure 2-2, is a fairly narrow estuary which has been dredged to a depth of about 40 feet to allow the passage of ocean-going vessels. This estuary can be considered laterally homogeneous, especially in the upper 14 miles where there is no pronounced influence from side bays. The channel receives a large amount of industrial and municipal wastes from outfalls which discharge into the channel at many points.

Salinity measurements have been made on a regular basis on the channel since April 1968. Early in the sampling program, salinity was recorded at every ten feet of depth at five points laterally across the channel. These measurements were taken every four miles up the 24-mile long channel. Eventually, it was decided to sample only the centerline of the channel, since the salinity seemed not to vary significantly across the channel. At that time, the sampling grid was tightened to include measurements at every five feet vertically and two miles longitudinally (29).

As can be seen from the raw data and from several Estuarine Systems Projects reports (92, 109), the salinity structure of the

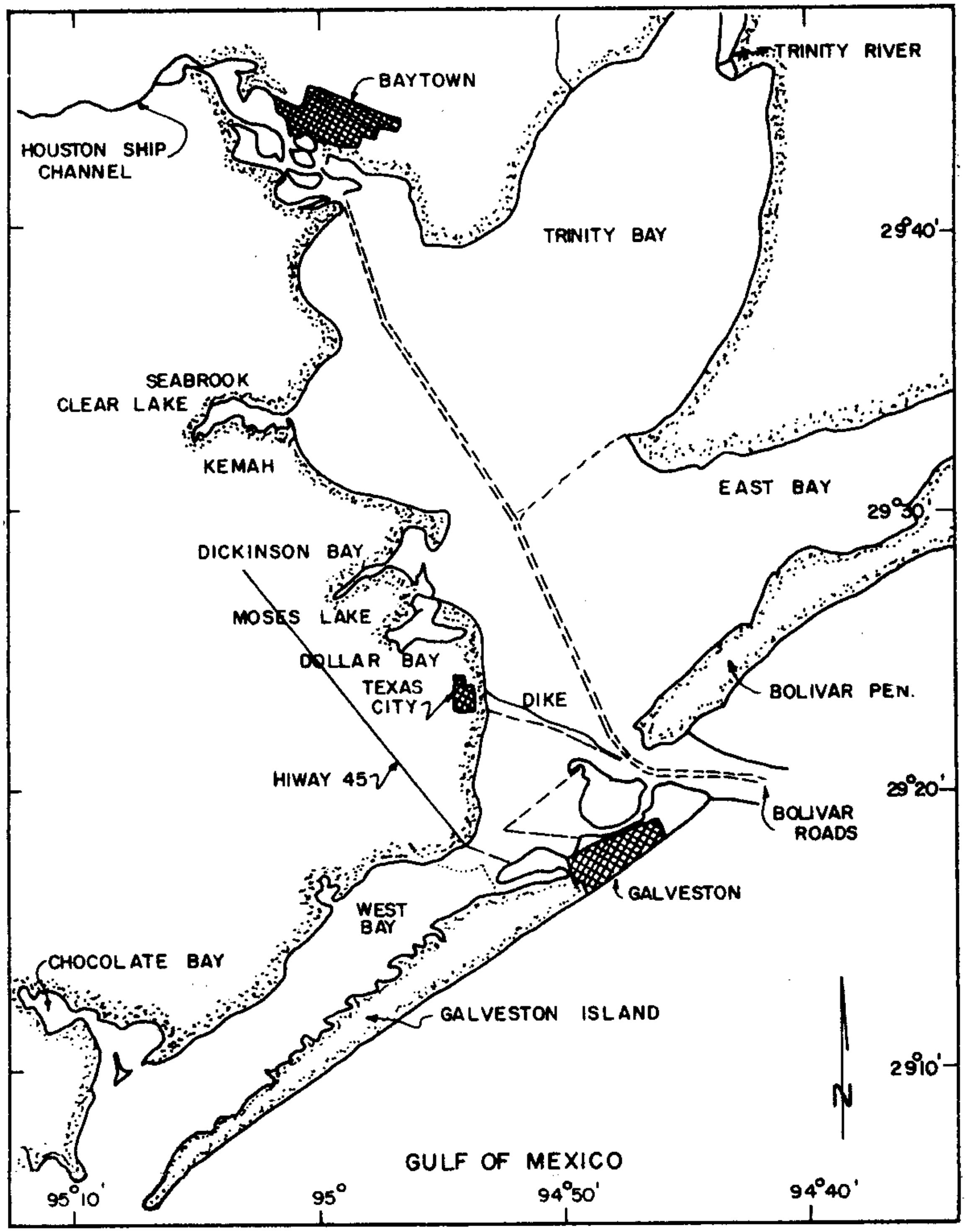


FIGURE 2-1.- GALVESTON BAY AREA

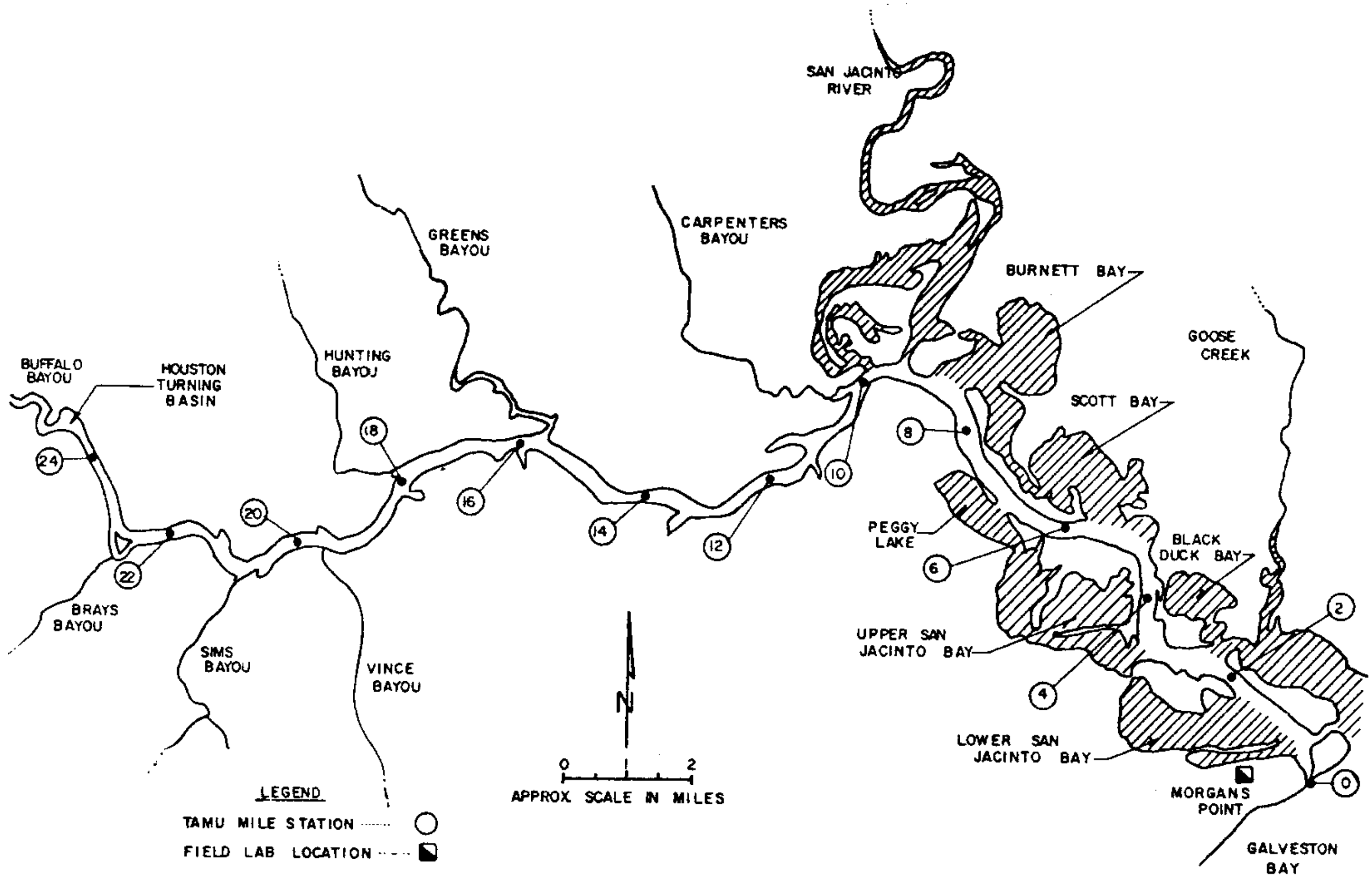


FIGURE 2-2. - HOUSTON SHIP CHANNEL

Houston Ship Channel is extremely variable. During some periods of heavy rainfall, the channel becomes highly stratified. At times of low inflow, the channel may become vertically well-mixed because of tidal agitation. However, the majority of the data indicates that the salinity structure is generally at some intermediate state of stratification, i.e., the salinity varies considerably from top to bottom with no sharp interface between fresh and saline water. Thus, the ship channel can best be classified as being "partially stratified." Calculations using typical values for inflow and tidal prism for the channel indicate that it usually falls within the "partially mixed" classification of previous estuary researchers (51, 107). The terms "partially mixed" and "partially stratified" are used synonymously in this report.

The channel can be adequately modeled by assuming a constant centerline depth and varying width; its behavior is essentially two-dimensional. However, several factors can be listed which keep the channel from being a simple, well-behaved hydraulic system:

1. The downstream end of the channel is a protected bay which is partially stratified and which has changeable salinity.
2. Part of the dredged channel is bordered by large, shallow side bays whose influence on flow patterns and dissolved oxygen patterns is difficult to determine.

3. A large tributary, the San Jacinto River, enters the channel about ten miles from the entrance to upper Galveston Bay and the flow of the river is erratic due to the operation of an upstream dam.
4. Tides in the 40-foot deep channel are erratic and of small amplitude, usually averaging about one foot in height.
5. Many chemical substances are added to the channel waters; their effects on flow patterns, salinity structure, and decay rates are difficult to determine.
6. Ship traffic on the channel has an undetermined effect on flow patterns and reaeration and interrupts the taking of measurements.
7. Benthal deposits and algae have a significant effect on water quality characteristics.

CHAPTER III

DEVELOPMENT OF THE ONE AND TWO DIMENSIONAL MASS TRANSPORT EQUATIONS

Mass transport is governed by the principle of conservation of matter. For the case of materials dissolved in a fluid, conservation of matter must be satisfied for each constituent. For such a system, the conservation principle can be summarized as follows:

$$\left\{\begin{array}{l}\text{TIME RATE OF}\\ \text{ACCUMULATION}\\ \text{OF CONSTITUENT}\\ \text{INSIDE A FLUID}\\ \text{ELEMENT}\end{array}\right\} = \left\{\begin{array}{l}\text{INFLOW OF}\\ \text{CONSTITUENT}\\ \text{TO FLUID}\\ \text{ELEMENT}\end{array}\right\} - \left\{\begin{array}{l}\text{OUTFLOW OF}\\ \text{CONSTITUENT}\\ \text{FROM FLUID}\\ \text{ELEMENT}\end{array}\right\} + \left\{\begin{array}{l}\text{TIME RATE OF PRODUCTION}\\ \text{OF CONSTITUENT BY}\\ \text{CHEMICAL AND BIOLOGICAL}\\ \text{REACTION INSIDE THE}\\ \text{FLUID ELEMENT}\end{array}\right\}$$

This principle can be expressed by the equation

$$\frac{\partial \rho_A}{\partial t} = -\left(\frac{\partial N_x}{\partial x} + \frac{\partial N_y}{\partial y} + \frac{\partial N_z}{\partial z}\right) + r_A \quad \ldots\ldots\ldots\ldots \quad (3\text{-}1)$$

or in the vector form

$$\frac{\partial \rho_A}{\partial t} = - \nabla \cdot \vec{N}_A + r_A \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3\text{-}2)$$

where ρ_A is the mass density of the mixture; $\vec{N}_A$ is a mass flux of constituent A across a boundary and is expressed in the units mass / (Length)^2 (Time); N_x, N_y, N_z are the components of $\vec{N}_A$; and r_A is the time rate of production of constituent A. This equation and its various forms are referred to by the names "equation of continuity for a constituent," "equation for conservation of a dissolved constituent," "mass transport equation," "mass balance equation," "diffusion equation," and the "convective dispersion equation."

The mass flux term, $\vec{N}_A$, represents transport by two phenomena: local fluid velocity and molecular diffusion:

$$N_A = - \rho_A D_A \nabla C_A + \rho_A C_A \vec{V} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3\text{-}3)$$

where D_A is the molecular diffusion coefficient for constituent A in the fluid, ρ_A is the mass density of the mixture; C_A is the concentration of constituent A; and $\vec{V}$ is the velocity vector. The negative diffusion term results from the tendency of a constituent in a mixture to move by molecular action in the direction of decreasing concentration of that constituent.

The velocities have both a time-averaged component, V_i, and a perturbation component, V_i', which represents the random variations in velocity from its average value. The concentrations also have a perturbation component, C_A'. A combination of these terms produces the following form for the turbulent mass flux (14):

$$\text{Turbulent Mass Flux Per Unit Area} = \rho_A \overline{(V_i' C_A')} \quad \ldots\ldots \quad (3\text{-}4)$$

where a time average is denoted by a bar over the appropriate terms.

By analogy with Fick's first law of diffusion, it is often assumed that the turbulent flux is proportional to the gradient of the time-averaged concentration; for example, in the x-direction:

$$\rho_A \overline{(V_x' C_A')} = \rho_A E_x \frac{\partial \overline{C}_A}{\partial x} \quad \ldots\ldots \quad (3\text{-}5)$$

where E_x is the turbulent diffusion coefficient.

If the isotropic molecular diffusion is assumed, these considerations lead to the following vector form of the mass transport equation for a single component, after the A subscripts are dropped:

$$\frac{\partial C}{\partial t} + \nabla \cdot C\vec{V} = D(\nabla \cdot \nabla C) + \nabla \cdot (E_i \nabla C) + \frac{r}{\rho} \quad \ldots\ldots\ldots\ldots \quad (3\text{-}6)$$

When continuity for an incompressible fluid is applied, a more useful form results:

$$\frac{\partial C}{\partial t} + V_x \frac{\partial C}{\partial x} + V_y \frac{\partial C}{\partial y} + V_z \frac{\partial C}{\partial z}$$

$$= \frac{\partial}{\partial x}\left(E_x \frac{\partial C}{\partial x}\right) + \frac{\partial}{\partial y}\left(E_y \frac{\partial C}{\partial y}\right) + \frac{\partial}{\partial z}\left(E_z \frac{\partial C}{\partial z}\right)$$

$$+ D \left(\frac{\partial^2 C}{\partial x^2} + \frac{\partial^2 C}{\partial y^2} + \frac{\partial^2 C}{\partial z^2}\right) + \frac{r}{\rho} \quad \ldots\ldots\ldots\ldots \quad (3\text{-}7)$$

The E_x, E_y, and E_z are the turbulent diffusion coefficients and can be only equal for the case of isotropic turbulence. In a natural system, the turbulent diffusion coefficients are many orders of magnitude larger than the molecular diffusion coefficients; hence it is acceptable to ignore completely the molecular diffusion terms. Of course, the terms on the left side of this equation can be more compactly expressed by the material derivative, DC/Dt.

Equation 3-6 can be simplified by dropping the molecular diffusion term, which is generally insignificant, and by eliminating the term $\frac{r}{\rho}$ by assuming that the material is conservative. This leaves the following form of the mass transport equation:

$$\frac{\partial C}{\partial t} = - \frac{\partial (V_x C)}{\partial x} - \frac{\partial (V_y C)}{\partial y} - \frac{\partial (V_z C)}{\partial z}$$

$$+ \frac{\partial}{\partial x} \left(E_x \frac{\partial C}{\partial x} \right) + \frac{\partial}{\partial y} \left(E_y \frac{\partial C}{\partial y} \right) + \frac{\partial}{\partial z} \left(E_z \frac{\partial C}{\partial z} \right) \quad . . . \text{(3-8)}$$

Equation 3-8 is associated with the general three-dimensional equation of continuity for an incompressible fluid:

$$\frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} + \frac{\partial V_z}{\partial z} = 0 \quad . . . . \text{(3-9)}$$

By expansion of a method suggested by Pritchard (78), Equations 3-8 and 3-9 can be treated to produce the two-dimensional and one-dimensional mass transport equations for a laterally homogeneous estuary. During this derivation, we assume that concentrations do not change with width and that the width does not change with time.

Assume that a(x, z) is the left-hand boundary of the estuary and b(x, z) is the right-hand boundary. By integrating Equation 3-8 between these boundaries we have

$$\int_a^b \frac{\partial C}{\partial t}\, dy = -\int_a^b \frac{\partial (V_x C)}{\partial x}\, dy - \int_a^b \frac{\partial (V_y C)}{\partial y}\, dy - \int_a^b \text{ etc.}$$

$$. . . . . . . \text{(3-10)}$$

By extension of the Leibnitz rule (57), we can write

$$\int_a^b \frac{\partial(V_xC)}{\partial x}\, dy = \frac{\partial}{\partial x}\int_a^b (V_xC)\, dy + (V_xC)_a \frac{\partial a}{\partial x} - (V_xC)_b \frac{\partial b}{\partial x} \qquad \text{(3-11)}$$

and

$$\int_a^b \frac{\partial}{\partial x}\left(E_x \frac{\partial C}{\partial x}\right) dy = \frac{\partial}{\partial x}\int_a^b \left(E_x \frac{\partial C}{\partial x}\right) dy + \left(E_x \frac{\partial C}{\partial x}\right)_a \frac{\partial a}{\partial x} - \left(E_x \frac{\partial C}{\partial x}\right)_b \frac{\partial b}{\partial x} \qquad \text{(3-12)}$$

and similarly for the other terms. We can integrate the y terms directly:

$$\int_a^b \frac{\partial(V_yC)}{\partial y}\, dy = (V_yC)_b - (V_yC)_a \qquad \text{(3-13)}$$

and

$$\int_a^b \frac{\partial}{\partial y}\left(E_y \frac{\partial C}{\partial y}\right) dy = \left(E_y \frac{\partial C}{\partial y}\right)_b - \left(E_y \frac{\partial C}{\partial y}\right)_a \quad \ldots\ldots\ldots \quad (3\text{-}14)$$

Since there can be no mass transfer across the boundaries and since the boundaries do not change with time, the following equations result from analysis of the derivatives:

$$(V_y C)_b = (V_x C)_b \frac{\partial b}{\partial x} + (V_z C)_b \frac{\partial b}{\partial z} \quad \ldots\ldots\ldots \quad (3\text{-}15)$$

$$(V_y C)_a = (V_x C)_a \frac{\partial a}{\partial x} + (V_z C)_a \frac{\partial a}{\partial z} \quad \ldots\ldots\ldots \quad (3\text{-}16)$$

$$\left(E_y \frac{\partial C}{\partial y}\right)_b = \left(E_x \frac{\partial C}{\partial x}\right)_b \frac{\partial b}{\partial x} + \left(E_z \frac{\partial C}{\partial z}\right)_b \frac{\partial b}{\partial z} \quad \ldots\ldots\ldots \quad (3\text{-}17)$$

$$\left(E_y \frac{\partial C}{\partial y}\right)_a = \left(E_x \frac{\partial C}{\partial x}\right)_a \frac{\partial a}{\partial x} + \left(E_z \frac{\partial C}{\partial z}\right)_a \frac{\partial a}{\partial z} \quad \ldots\ldots\ldots \quad (3\text{-}18)$$

If equations 3-11 through 3-18 are substituted into Equation 3-10, we are left with

$$\frac{\partial}{\partial t}\int_a^b C\,dy = -\frac{\partial}{\partial x}\int_a^b \left(V_x C\right) dy - \frac{\partial}{\partial z}\int_a^b \left(V_z C\right) dy$$

$$+ \frac{\partial}{\partial x}\int_a^b \left(E_x \frac{\partial C}{\partial x}\right) dy + \frac{\partial}{\partial z}\int_a^b \left(E_z \frac{\partial C}{\partial z}\right) dy$$

$$\ldots\ldots\ldots\ldots \quad (3\text{-}19)$$

Since we have assumed lateral homogeneity, this equation reduces to

$$\frac{\partial (WC)}{\partial t} = -\frac{\partial (WV_x C)}{\partial x} - \frac{\partial (WV_z C)}{\partial z}$$

$$+ \frac{\partial}{\partial x}\left(E_x W \frac{\partial C}{\partial x}\right) + \frac{\partial}{\partial z}\left(E_z W \frac{\partial C}{\partial z}\right) \quad \ldots\ldots\ldots\ldots \quad (3\text{-}20)$$

where $W = b - a$, which is the estuary width.

If we apply the Leibnitz rule and the same boundary conditions to Equation 3-9, we find that the two-dimensional continuity equation with regard to width becomes

$$\frac{\partial (WV_x)}{\partial x} + \frac{\partial (WV_z)}{\partial z} = 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3\text{-}21)$$

Combining Equation 3-20 and Equation 3-21 gives the two-dimensional mass transport equation which is used most frequently in this study:

$$W \frac{\partial C}{\partial t} = - WV_x \frac{\partial C}{\partial x} - WV_z \frac{\partial C}{\partial z}$$

$$+ \frac{\partial}{\partial x}\left(E_x W \frac{\partial C}{\partial x}\right) + \frac{\partial}{\partial z}\left(E_z W \frac{\partial C}{\partial z}\right) \quad . . . . \quad (3\text{-}22)$$

If we wish to derive the one-dimensional mass transport equation, we can apply the same type of analysis to Equation 3-8 using the cross-sectional area, A_x; the following equation results:

$$A_x \frac{\partial C}{\partial t} = - A_x V_x \frac{\partial C}{\partial x} + \frac{\partial}{\partial x}\left(E_x A_x\right)\frac{\partial C}{\partial x} \quad . . . . \quad (3\text{-}23)$$

Appropriate source and sink terms can be added to Equations 3-22 and 3-23 for the various substances being transported.

For an estuary of constant depth, the cross-sectional area term can be replaced by a width term, W_x:

$$W_x \frac{\partial C}{\partial t} = -W_x V_x \frac{\partial C}{\partial x} + \frac{\partial}{\partial x}\left(E_x W_x\right)\frac{\partial C}{\partial x} \quad . . . . \quad (3\text{-}24)$$

CHAPTER IV

RECENT MATHEMATICAL MODELS OF ESTUARIES

Because of increased concern over the degradation of estuaries, advanced techniques are needed to analyze the behavior of pollutants in these bodies of water. When developing a plan for controlling the quality of a body of water, one of the crucial problems is the formulation of a model which accurately simulates the behavior of the system. The most significant work relating to the mathematical modeling of estuaries has been done in the last decade. In the early 1960's, researchers concentrated on applying closed-form, analytical equations to estuaries. As experience has been gained, more complex methods have been developed with a heavier reliance on digital computers. The most recent models generally require a high degree of mathematical sophistication including numerical solution on a computer of complex systems of partial differential equations. Once a simulation model has been developed and verified, techniques of operations research can often be applied to determine the minimum cost solution to the water quality problems.

Most analyses dealing with water quality problems are based on some form of the mass transport equation. Some manipulation of terms in this equation is necessary for constituents which interact, such as biochemical oxygen demand (BOD) and dissolved oxygen. For parameters which undergo progressive changes, such as in the nitrogen cycle, the mass transport equation may have to be used several times to describe the system adequately. When a steady-state condition in a river is assumed, the diffusion and dispersion terms can be dropped and formulations such as the classical Streeter-Phelps equation (94) can be derived directly.

Determining the success of any model in adequately representing a system is highly dependent upon the amount of data available from that system. Assuming a model has been formulated correctly, the accuracy of the model then depends on choosing the correct values for reaeration, decay, dispersion, salinity gradients, velocity of flow, volume of pollution, and various factors related to tidal action and currents. Attempts to use a model to predict water quality should be made only after that model has been shown to fit adequately a large amount of existing data.

Estuary modeling of particular interest has been done by many American researchers. Important work also has been done in foreign countries; for instance, the Water Pollution Research Laboratory of Great Britain has made a landmark study of the Thames (104) and the Delft Hydraulics Laboratory makes regular studies on estuaries in The Netherlands (35). The state of current research for estuary mathematical models can best be indicated by a brief review of the work of some of these research efforts.

MODELING METHODS OF DONALD J. O'CONNOR

O'Connor is the best known researcher in the field of mathematical modeling of estuaries. His 1960 paper (65) entitled, "Oxygen Balance of an Estuary" was the first comprehensive report on modeling of the behavior of non-conservative substances in these bodies of water. During the past decade, O'Connor has written many additional papers (66, 67, 68, 69) dealing with the modeling of various estuaries.

O'Connor's approach to the modeling of estuaries is generally characterized by steady-state analytical solutions to a one-dimensional form of the estuary mass transport equation. Two of the most useful of these equations will be discussed later in this chapter as Equations 4-5 and 4-6. O'Connor's use of analytical equations is most versatile when several of these equations are used simultaneously. This approach was applied to the East River, New York, where the system was divided into segments, each with its appropriate general solution. The concentration profile for the entire profile was then determined by solving a set of equations which matched at the segment boundaries (68). This approach requires that the variation in cross-sectional area of segments of the estuary be expressed as well-behaved functions of distance. O'Connor often uses expressions for cross-sectional area which are linear, monomial, or exponential functions of distance. The dispersion term E_x is often used to include the influence of tides as well as eddy diffusion.

O'Connor's methods have been used to obtain steady-state concentration profiles for many estuaries including the New York Harbor, East River, Delaware River, James River, and Raritan River. Time-varying equations also have been developed and applied.

MODELING METHODS OF ROBERT V. THOMANN

A widely used mathematical model for the Delaware Estuary is primarily the work of Thomann (97, 98, 99, 100, 101, 102, 75). Thomann's initial formulation of the model was presented in his doctoral dissertation, "The Use of Systems Analysis to Describe the Time Variation of Dissolved Oxygen in a Tidal Stream" (97). Thomann later became the technical director of the Delaware Estuary Comprehensive Study (DECS) under the Public Health Service and Federal Water Pollution Control Administration (FWPCA); for several years, a large amount of data was collected by DECS and the model was improved. In July, 1966, the FWPCA published a report (24) which recorded the status of the model and the recommendations of that government agency for the uses of the river. Since that time, the Thomann model has been applied to the Potomac (42) and other estuaries.

A numerical solution to the basic equations in the model was obtained by the DECS from the Re-Entry Systems Department of the General Electric Company. The work at General Electric was supervised by Jeglic (54, 55). A recent digital computer program, called DECS III, is a refined, time-dependent version of the original model. It has been programmed in several versions of of FORTRAN IV and can be used by many of the larger digital computers. A steady-state version of the Thomann model was been documented by Bunce and Hetling (9).

The basic form of the Thomann equation for BOD is as follows:

$$\frac{dL_k}{dt} = \frac{Q_{k-1,\,k}}{VOL_k}\left[\xi_{k-1,\,k}\,L_{k-1} + (1 - \xi_{k-1,\,k})\,L_k\right]$$
$$- \frac{Q_{k,\,k+1}}{VOL_k}\left[\xi_{k,\,k+1}\,L_k + (1 - \xi_{k,\,k+1})\,L_{k+1}\right]$$
$$+ \frac{E'_{k-1,\,k}}{VOL_k}\left(L_{k-1} - L_k\right) + \frac{E'_{k,\,k+1}}{VOL_k}\left(L_{k+1} - L_k\right)$$
$$- d_K\,L_k \pm J_k \quad \text{. . . . . (4-1)}$$

In this equation, k ranges from 1 to n, where n = the number of sections, k-1 = the upstream section, k+1 = the downstream section, L = the ultimate carbonaceous biochemical oxygen demand concentration, Q = the net flow from section to section, VOL = the volume of the section, ξ = an advective coefficient dependent upon the ratio of the dispersion to the advective forces, E' = an eddy exchange coefficient analogous to the classical eddy diffusion coefficient in non-tidal streams, d = the decay rate of biochemical oxygen demand, and J = the direct sources of biochemical oxygen demand.

It is obvious from this equation that Thomann attempts to separate some of the effects of tides from the eddy exchange coefficient, E'. It should be noted that the dimensions of E' must be L^3/T to make the equation dimensionally correct. Thus, Thomann's E' can not be considered equal to the dispersion or diffusion coefficients (D or E) of other investigations.

The advective coefficient, ξ, varies between a value of 0.5 for an environment dominated almost entirely by tides to a value of 1.0 for a non-tidal stream. For situations between these extremes, ξ is assumed to be some function of Q and E'. In some applications, ξ has been assumed equal to either 0.5 (99) or to the ratio $l_k / (l_k + l_{k-1})$ where l is the length of the appropriate segment (9). If ξ is equal to 0.5, the Thomann equation becomes a one-dimensional finite-difference form of the mass transport equation when E' is set equal to $EA/\Delta x$, where A is the cross-sectional area. A discussion of the problems associated with evaluating the various parameters in the Thomann equation is found in the proceedings of the Stanford ASCE Symposium on Estuarine Pollution (75, 76).

Recently, Hays (38, 39) has applied the steady-state version ($dL_k/dt = 0$) of the Thomann model to the Houston Ship Channel. Hays applied non-linear programming techniques to this one-dimensional model and obtained least-cost solutions for producing certain dissolved oxygen profiles in the channel.

MODELING METHOD OF WATER RESOURCES ENGINEERS, INC.

The mathematical model most often applied to the San Francisco Bay and Delta region was developed for the FWPCA (now the Environmental Protection Agency, EPA) by Water Resources Engineers, Inc., under the supervision of Orlob. The first stage of this model was formulated in 1965 to represent the water quality in the Sacramento-San Joaquin Delta (90, 105). Verification of this phase of the model was achieved by comparison with past salinity and hydrological conditions and with the results of a prototype dye tracer study. During the following year, 1966, the model was extended to Suisun and San Pablo Bay (72, 106). The model in this form was the major tool used in predicting the effects of the proposed San Joaquin master drain (25) on the future water quality of the estuary.

The model uses finite-difference approximations to simplified, one-dimensional forms of the equation of motion, the equation of continuity, and the mass transport equation for conservative materials (106):

$$\frac{\partial V_x}{\partial t} + V_x \frac{\partial V_x}{\partial x} + g \frac{\partial H}{\partial x} + K |V_x| V_x = 0 \quad \ldots\ldots\ldots\ldots \quad (4\text{-}2)$$

$$\frac{\partial H}{\partial t} + \frac{\partial Q}{\partial x} = 0 \quad \ldots\ldots\ldots\ldots \quad (4\text{-}3)$$

$$\frac{\partial C}{\partial t} = E_x \frac{\partial^2 C}{\partial x^2} - V_x \frac{\partial C}{\partial x} \quad \ldots\ldots\ldots\ldots \quad (4\text{-}4)$$

where K represents a frictional resistance term; H is the tidal height; Q is the flow rate; V_x is the velocity; and the remaining terms are defined as in previous sections of this report. These one-dimensional equations were applied to a two-dimensional grid representing the Bay-Delta watercourse. The model used a network of channels which are linked at nodes to describe the area covered by the Delta, Suisun Bay, and San Pablo Bay. This grid system was a major distinction between the San Francisco Bay model and the Delaware Estuary model, which simply divided the Delaware into a series of reaches.

Water Resources Engineers chose to employ an explicit scheme in solving the above equations because of the simplicity of this type of formulation and because the step size that can be used is relatively independent of the size of the system (106). The water quality model uses intra-tidal velocities, and it was found that the effects of advective transport were much more important than eddy dispersion. Thus, the dispersion term could be ignored and only the velocity term was used in the refined version of the conservative mass transport equation. Some problems developed because of the "numerical mixing" error which was a consequence of the numerical methods used to integrate the mass transport equation and the application of one-dimensional equations to a two-dimensional system. Techniques have been developed to circumvent these problems by applying a weighted-average factor to the concentrations, but generally either accuracy or stability of the solution must be sacrificed to a certain degree (72).

The Water Resources Engineers' model is operated as follows: The hydrodynamic model is operated for several equal tidal cycles until stable intra-tidal velocities are obtained; normally these velocities are obtained for 30-minute increments (72). These velocities are fed into the water quality model which is run for a large number of equal tidal cycles until a stable equilibrium pattern is developed. The rate of convergence of the quality model is somewhat dependent upon the initial concentrations used; thirty tidal cycles is a representative number of repetitions needed before equilibrium is obtained.

In the use of this model by the FWPCA, primary emphasis was placed on total dissolved solids and total nitrogen, where nitrogen was considered as a conservative element. The principal characteristics of the present nitrogen and salinity distributions in the

Bay-Delta have been reproduced by the model with a satisfactory degree of accuracy. Non-conservative substances can be directly represented by the model and several preliminary studies have been conducted which illustrate the capability of the model to handle BOD and DO distributions. However, the model is set up on the assumption of complete mixing at a cross-section and, therefore, cannot handle a vertically stratified body of water.

This model has been applied to other bodies of water, including the San Diego Bay (26) and the Columbia River estuary (10). Recent documentation of the model by Feigner and Harris (26) show the following characteristics: The hydrodynamic model is typically run for 1 to 4 tidal cycles with a resulting execution time of 7 to 23 minutes on a CDC 6600 computer. The water quality model is typically run for 20 to 56 tidal cycles with a resulting execution time of 3 to 14 minutes on the CDC 6600. Execution time for an IBM 360/65 computer was found to be 2 to 3 times greater than those reported for the CDC 6600. In the above applications of the model, the number of junctions ranged between 112 and 830, and the number of channels ranged between 170 and 1050.

MODELING METHOD OF HANN AND EVETT

A mathematical model developed at Texas A&M University under the supervision of Hann has been applied to several Gulf Coast estuaries including the San Bernard, the Neches, and the Houston Ship Channel. The primary reference for this model is a doctoral dissertation written by Evett (23). A revised edition of this model has been written in a problem-oriented language (POL) format (30).

In comparison with all of the other models discussed thus far, Hann's approach is unique in that it can be applied to partially stratified estuaries. This model uses the following two solutions to a one-dimensional form of the mass transport equation:

$$C = C_o e^{-K_d x/U} \quad \text{. . . . . (4-5)}$$

and

$$C = C_o e^{jx} \quad \text{. . . . . (4-6)}$$

where

$$j = \frac{U}{2E}\left(1 \pm \sqrt{1 + \frac{4EK_d}{U^2}}\right)$$

For these equations, E = the dispersion coefficient, C = the waste concentration, C_o = the concentration at the outfall, x = the horizontal distance, U = the advective velocity, and K_d = the removal coefficient.

Equation 4-6 represents concentration of a non-conservative pollutant in a vertically mixed (homogeneous) estuary where the optional minus sign is used for the downstream direction. Equation 4-5 represents the concentration below an outfall in a river with no horizontal dispersion. Equation 4-5 is used in the upper, fresh-water layer of a stratified estuary where it is assumed that diffusion is insignificant in comparison to advective transport. Allowances also are made to account for salt water flow upward from the saline wedge. The development of both the homogeneous and stratified cases assume a steady-state condition.

The methods of modeling a homogeneous estuary or a highly stratified estuary proceed in essentially the same manner. Initially, the estuary is divided into segments. The lengths of these segments can vary; usually one-mile segments are most convenient. A calculation is then made using Equations 4-5 and 4-6 to determine the concentration that would appear in each segment under steady-state conditions as a result of injecting a unit waste load of one pound per day into a particular segment. After this calculation is repeated for each segment, a matrix of "unit loading coefficients" results. The appropriate unit loading coefficients can then be multiplied by the actual waste load being put into each segment and the waste distribution in the estuary is obtained.

The unit loading coefficients are adjusted for use in a channel with varying area by the following technique. Assume that a unit load (one pound of BOD per day) is introduced into segment 5 and a concentration of C_5 results. The quality in the downstream segment 6 is then calculated assuming that it has a volume equal to segment 5. The calculation is made using Equation 4-6 for a homogeneous estuary and Equation 4-5 for the top layer of a highly stratified estuary. The next step is to modify the quality parameter in segment 6 to reflect the changed volume by means of the following formula:

$$C_{6,\ \text{modified}} = \frac{VOL_5}{VOL_6} C_6 \quad \text{(4-7)}$$

where VOL_5 add VOL_6 equal the volumes of segments 5 and 6, respectively. This procedure is then repeated using the modified C_6 value to calculate C_7 and so on down the estuary. Similar calculations are made in the upstream direction in the case of a homogeneous estuary.

The procedure used by Hann and Evett to analyze a partially mixed estuary is based on the assumption that the waste concentration in each segment will fall at some value between that for a highly stratified estuary and that for a homogeneous estuary. The actual value will depend upon the amount of stratification existing in the estuary. For each segment, a calculation is made to determine a "degree of stratification", DS. The calculation of this term requires the following segment input parameters: average top salinity, A; average bottom salinity, B; and salinity of the ocean or bay outside the estuary, C. The "degree of stratification" is then calculated by the following formula:

$$DS = \frac{B-A}{C-A} \quad \ldots \quad (4\text{-}8)$$

Thus, to determine the waste pattern in a partially stratified estuary, a calculation is made for a homogeneous estuary and for a completely stratified estuary, and then each calculation is weighted in accordance with the above "degree of stratification." Evett (23) applied linear programming techniques to this model to determine the maximum waste discharges which could be applied to the channel while satisfying certain dissolved oxygen criteria.

ANALYTICAL METHODS OF HARLEMAN AND IPPEN

Harleman and Ippen have produced some of the most significant American research in relating experimental findings to the behavior of actual estuaries. Their work is summarized briefly in this section to demonstrate an alternative approach to the numerical analysis techniques employed by other researchers. Likewise, the experiments of Harleman and Ippen give significant insight toward determining some of the important input parameters for mass transport models.

Most of the estuary studies performed by these investigators since 1960 were initiated by the U. S. Army Corps of Engineers Committee on Tidal Hydraulics. The experimental data were obtained in the rectangular salinity flume of the Waterways Experiment Station at Vicksburg and in a smaller flume at the Hydrodynamics

Laboratory of the Massachusetts Institute of Technology. The scope of these investigations covered the following four phases and were reported in the indicated references:

1. The extent of salinity intrusion and the mean salinity distribution in an idealized estuary (50, 51);

2. The vertical mixing of fresh and salt water and the resulting vertical salinity distribution (36);

3. The vertical distribution of current velocities as affected by salinity distribution (36); and

4. The movement and deposition of sediments as affected by the density current phenomena (37).

The estuary research of Harleman and Ippen has led to a number of unique parameters which they feel are important in analyzing estuaries. Much of their work has been presented in a non-dimensional format so that results can be applied to estuaries of widely varying dimensions and behavior.

Most of the work done by Harleman and Ippen has been correlated with a parameter known as the "stratification number." The use of this parameter was concisely explained in these authors' analysis of the Rotterdam waterway (35): "The stratification number, G/J, is defined as the following ratio.

$$\frac{G}{J} = \frac{\text{rate of energy dissipation per unit mass of fluid}}{\text{rate of gain of potential energy per unit mass of fluid}} \qquad (4\text{-}9)$$

The numerator was evaluated from an analysis of the damping of the tidal wave in the channel. The denominator reflects the gain in potential energy due to increasing specific weight as the water becomes saline in moving down the estuary to the ocean. A large value of stratification number (of the order of 100 or more) indicates a well mixed estuary condition. Lower values indicate an increasing tendency toward stratification with a two-layer saline wedge as a limiting case. Although the stratification number was a useful parameter in correlating the Vicksburg Waterway Experiment Station (W.E.S.) channel tests, it is inconvenient to evaluate in actual estuaries. The difficulty is primarily due to the necessity of evaluating the rate of energy dissipation from tidal data."

During a reanalysis of the W.E.S. data it was found that a dimensionless parameter containing the tidal prism, Froude number (based on the maximum tidal velocity), fresh water discharge, and the tidal period was uniquely related to the stratification number. For convenience in later discussion this combination will be called "estuary number." In comparison with the stratification number, the estuary number is relatively easy to evaluate in actual estuaries. The estuary number is defined as follows:

$$\text{estuary number} = \frac{P_t F_o^{\,2}}{Q_f T} \quad \ldots\ldots\ldots\ldots \quad (4\text{-}10)$$

where P_t = tidal prism, the volume of sea water entering the estuary on the flood tide, F_o = Froude number = $u_o/\sqrt{gh}$, u_o is the maximum flood tide velocity at x = 0, and h is the mean depth at x = 0, Q_f = fresh water discharge, and T = tidal period. The relation between the stratification number and the estuary number is shown in Figure 4-1 for the W.E.S. tests.

In their one-dimensional treatment of estuary analysis, Harleman and Ippen use what they term an "apparent diffusion coefficient D_x' (50, 35, 51). Through this term, they attempt to take into consideration the effects of large scale internal circulation estuaries and the net effects of vertical transport. The ratio of D_x' to the actual longitudinal coefficient of turbulent diffusion (D_t) approaches unity for the well-mixed estuaries and approaches an order of 1000 for the highly stratified (51). The D_x'/D_t ratio ranged between 20 and 180 for the W.E.S. studies (50).

To solve the one-dimensional form of the mass transport equation, it is necessary to express the diffusion coefficient in terms of x. Harleman and Ippen assumed that the apparent diffusion coefficient decreased linearly with x in the upland direction. This approach seemed to work well for a salinity flume with a constant cross-section. Since salinity at low water slack cannot equal "ocean" salinity, the variation of D_x' was extended seaward a distance B where ocean salinity was assumed to exist. The equation for the apparent diffusion coefficient becomes:

$$D_x' = \frac{D_o' B}{x + B} \quad \text{(4-11)}$$

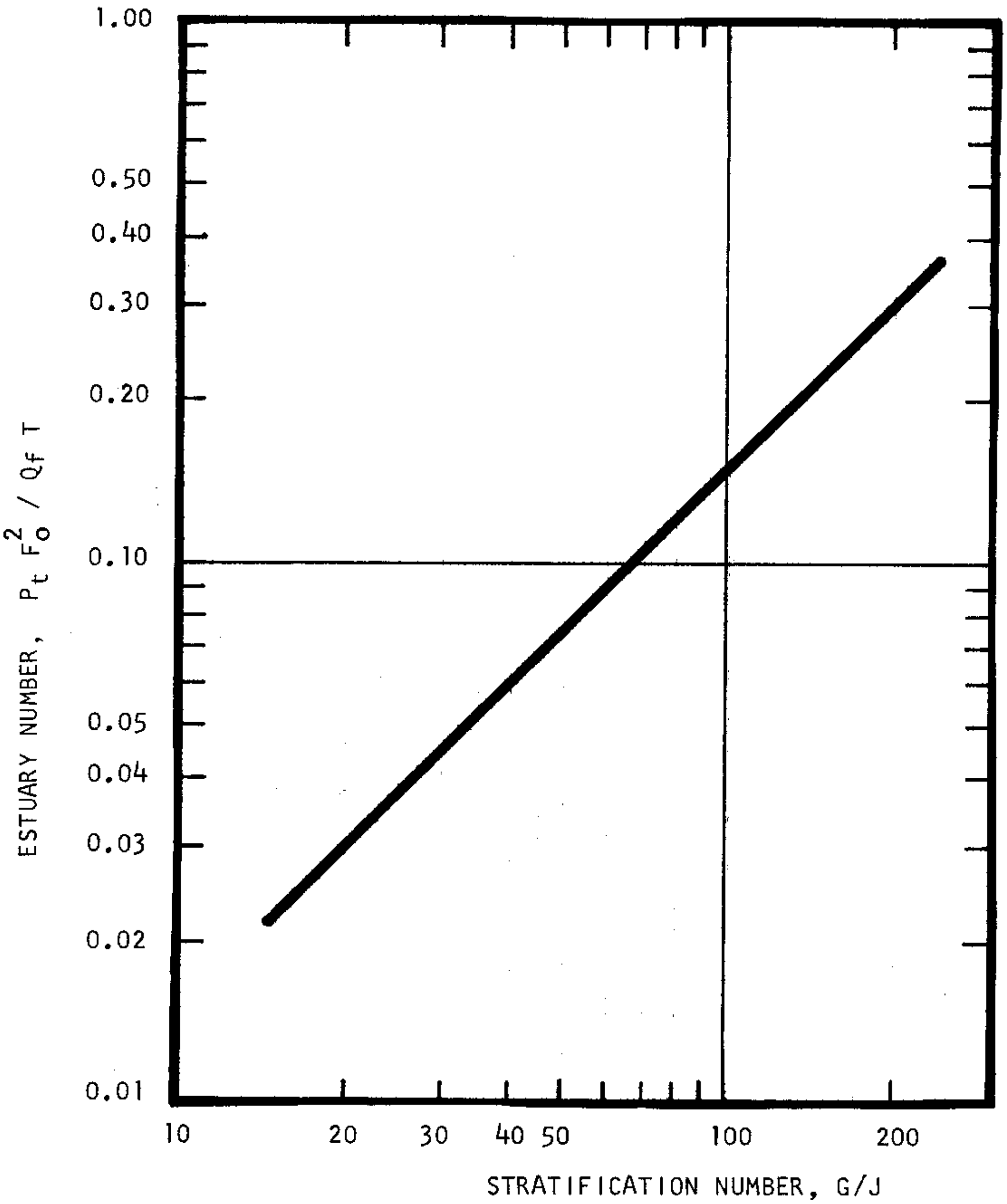


FIGURE 4 - 1. - RELATIONSHIP BETWEEN STRATIFICATION NUMBER AND ESTUARY NUMBER

where $D'_x = D'_o$ at $x = 0$, the mouth of the channel. Thus, the governing equation for salinity in the one-dimensional analysis becomes:

$$\frac{\bar{s}_{lws}}{s_o} = \exp\left(- V_f \left(x + B \right)^2 / 2 D'_0 B\right) \quad . \; . \; . \; . \; . \; . \; . \; . \; . \; . \quad (4\text{-}12)$$

where $\bar{s}_{lws}$ = local salinity at low water slack, s = ocean salinity, and V_f is the velocity due to fresh water discharge. If salinity is known at low water slack for at least two points, then the parameters D'_0 and B can be determined.

Other parameters which Harleman and Ippen feel are important in one-dimensional analysis are $D'_0 / V_f B$, $2 B/u_0 T$, and $Q_f T/P_t$; all the parameters in these ratios have been defined previously. Analysis using these parameters has been applied to the Rotterdam Waterway by Harleman and Abraham (35). Several characteristics of the waterway were defined; unfortunately, most of the Vicksburg flume data falls outside the range of the waterway data.

Most recent studies of the two-dimensional behavior of estuaries have been performed by Harleman and Ippen to analyze vertical transport (36) and shoaling (37). A large amount of information was collected on tidal time-averaged values of vertical velocity, horizontal velocity, salinity profiles, and vertical dispersion in an idealized estuary of constant cross section. Vertical velocities were found to be in the downward direction in the seaward half of the intrusion region and upward in the landward half (36). this finding is in contradiction to some other researchers who assume that a net vertical motion from bottom to surface exists throughout the intrustion length. In addition, certain parameters were found which proved to be useful in the qualitative analysis of estuary shoaling.

TWO-DIMENSIONAL MODELS OF GALVESTON BAY

Several studies of Galveston Bay, Texas, have been made recently to evaluate the bay's hydrodynamic and mass transport characteristics. Reid and Bodine modeled the behavior of the bay for storm surge conditions (84). Masch and Shankar extended this work to include mass transport considerations (61, 62, 88, 89). The Galveston Bay Study, through the consulting firm Tracor, Inc., is applying similar models to Galveston Bay (21, 22). In addition, the Texas Water Development Board is extending the use of the Masch/Shankar models to other estuaries along the Texas coast (96).

The work of Reid and Bodine is based on a vertically integrated form of the equations of motion and continuity. These equations allow for rainfall, wind stress, and quadratic bottom friction but ignore terms dealing with momentum advection and the Coriolis force:

$$\frac{\partial U}{\partial t} + gZ \frac{\partial H}{\partial x} = KW^2 \cos\Psi - fQUZ^{-2} \quad \ldots\ldots\ldots\ldots\ldots \quad (4\text{-}13)$$

$$\frac{\partial V}{\partial t} + gZ\frac{\partial H}{\partial y} = KW^2\sin\Psi - fQVZ^{-2} \quad \text{(4-14)}$$

$$\frac{\partial H}{\partial t} + \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y} = R \quad \text{(4-15)}$$

where U and V are the vertically integrated x and y horizontal components, respectively, of transport per unit width; H is the water level elevation relative to the local mean sea level datum; Z is the depth of water at position x, y, at time t; Q is equal to the vector average of transport per unit width and is obtained from the positive root of the radical $Q = \sqrt{U^2 + V^2}$; R equals the rainfall rate; f is a dimensionless bed resistance coefficient; W is the wind speed 10 meters above the water; Ψ is the angle between the wind velocity vector and the x-axis; K is the dimensionless Van Dorn coefficient for wind stress and is considered to be a function of W. Reid and Bodine solve Equations 4-13, 4-14, and 4-15 for U, V, and H, using a finite-difference recursion equation (84). The two-dimensional grid for Galveston Bay uses a two-nautical-mile spacing. The model has been verified for Hurricanes Carla of 1961 and Cindy of 1963.

Masch and his associates (61) have made certain revisions to the Reid/Bodine hydrodynamics model in order to make the model more applicable to conditions other than storm surges. Part of this refinement included changing the spacing to a one-mile grid. Shankar (88, 89, 62) added supplemental versatility to the model by coupling it with a two-dimensional salinity model.

Shankar applies the following form of the mass transport equation, where the terms are defined as in previous examples:

$$\frac{\partial C}{\partial t} = \frac{\partial}{\partial x}\left(E_x \frac{\partial C}{\partial x}\right) + \frac{\partial}{\partial y}\left(E_y \frac{\partial C}{\partial y}\right) - U \frac{\partial C}{\partial x} - V \frac{\partial C}{\partial y} \quad . \; . \; . \; . \; (4\text{-}16)$$

The Bay was represented according to a repeating grid of cells shown in Figure 4-2. In this grid, the dispersion terms and concentration are cell-centered, whereas the velocities are defined at the cell boundaries. Thus, the following explicit finite-difference equation was developed (88, 89) to approximate Equation 4-16:

$$\frac{C_{i,j}^{(k+1)} - C_{i,j}^{(k)}}{\Delta t} = \frac{1}{2(\Delta x)^2}\left[E_{x_{i,j}} C_{i+1,j}^{(k)} - 2E_{x_{i,j}} C_{i,j}^{(k)} + E_{x_{i,j}} C_{i-1,j}^{(k)} + E_{x_{i+1,j}}\left(C_{i+1,j}^{(k)} - C_{i,j}^{(k)}\right) - E_{x_{i-1,j}}\left(C_{i,j}^{(k)} - C_{i-1,j}^{(k)}\right)\right]$$

$$+ \frac{1}{2(\Delta y)^2}\left[E_{y_{i,j}} C_{i,j+1}^{(k)} - 2E_{y_{i,j}} C_{i,j}^{(k)} + E_{y_{i,j}} C_{i,j-1}^{(k)} + E_{y_{i,j+1}}\left(C_{i,j+1}^{(k)} - C_{i,j}^{(k)}\right)\right.$$

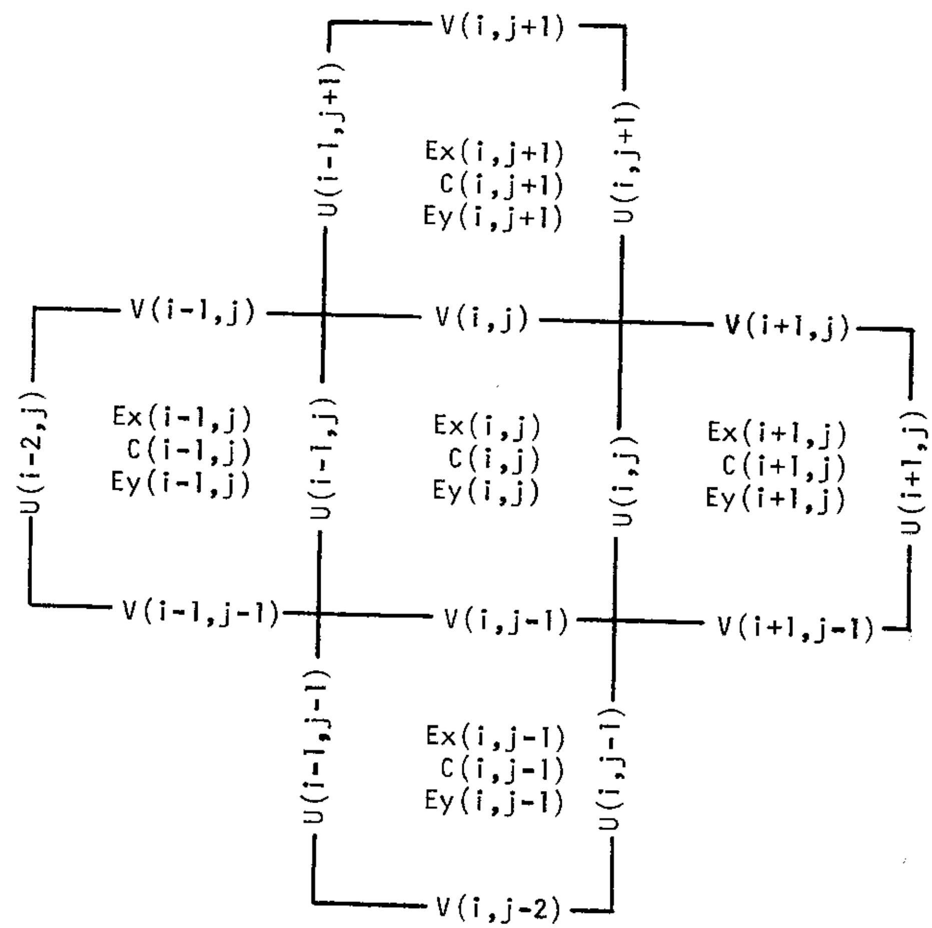


FIGURE 4-2. - CELL STRUCTURE FOR SALINITY MODEL OF SHANKAR AND MASCH

$$- E_{y_{i,j-1}} \left(C_{i,j}^{(k)} - C_{i,j+1}^{(k)} \right) \Bigg]$$

$$- \left(\frac{U_{i,j} + U_{i-1,j}}{2} \right) \left[\frac{C_{i+1,j}^{(k)} - C_{i-1,j}^{(k)}}{2\Delta x} \right]$$

$$- \left(\frac{V_{i,j} + V_{i,j-1}}{2} \right) \left[\frac{C_{i,j+1}^{(k)} - C_{i,j-1}^{(k)}}{2\Delta y} \right] \quad \ldots\ldots\ldots (4\text{-}17)$$

This equation can be rearranged to compute the unknown salinities, $C_{i,j}^{(k+1)}$, at time t + Δt from the known salinities, $C^{(k)}$, at time t.

Tracor and the Galveston Bay Study (GBS) use a hydrodynamic model for the Bay which is basically the same as that used by Masch (21). Likewise, the one-mile grid used by the GBS for mass transport is defined in a manner similar to that of Shankar (Figure 4-2), except that the GBS expresses U, V, and C at the cell center and dispersion at the cell walls (22). In addition, the GBS model includes terms for various sources and sinks. Tracor has applied the GBS model extensively to predict salinity, BOD, temperature, and nutrient patterns in Galveston Bay. Tracor has recently edited a noteworthy report which assesses the state-of-the-art of estuary modeling techniques (20) additional information on the Galveston Bay model can be found in that report.

Recently, Leendertse (59, 60) has also applied finite difference schemes to hydrodynamic and water quality phenomena.

VERTICAL MODEL BY PRITCHARD AND WILSON

A two-layered, segmented model was suggested by Pritchard (81) in 1969. His approach has been applied recently by Wilson (108) at the Chesapeake Bay Institute under Pritchard's supervision. This analysis was directed towards the flushing of pollutants from a partially mixed estuary in which the flow is strongly two-layered. The segmentation method used by Wilson is shown in Figure 4-3. Horizontal dispersion was ignored on the assumption that mixing could be represented properly by the combined use of vertical mixing and horizontal velocity.

A set of equations was developed to represent the mass balances between the segments and these equations were integrated with time by a Hamming predictor-corrector scheme. This approach was applied to the Northwest Branch of the Baltimore Harbor.

Wilson (108) concluded that the two-layered, segmented model could describe adequately the gross features of advection and diffusion within the Northwest Branch. Use of the model led to estimates of steady-state concentrations; these estimates were considered to be in good accord with field data.

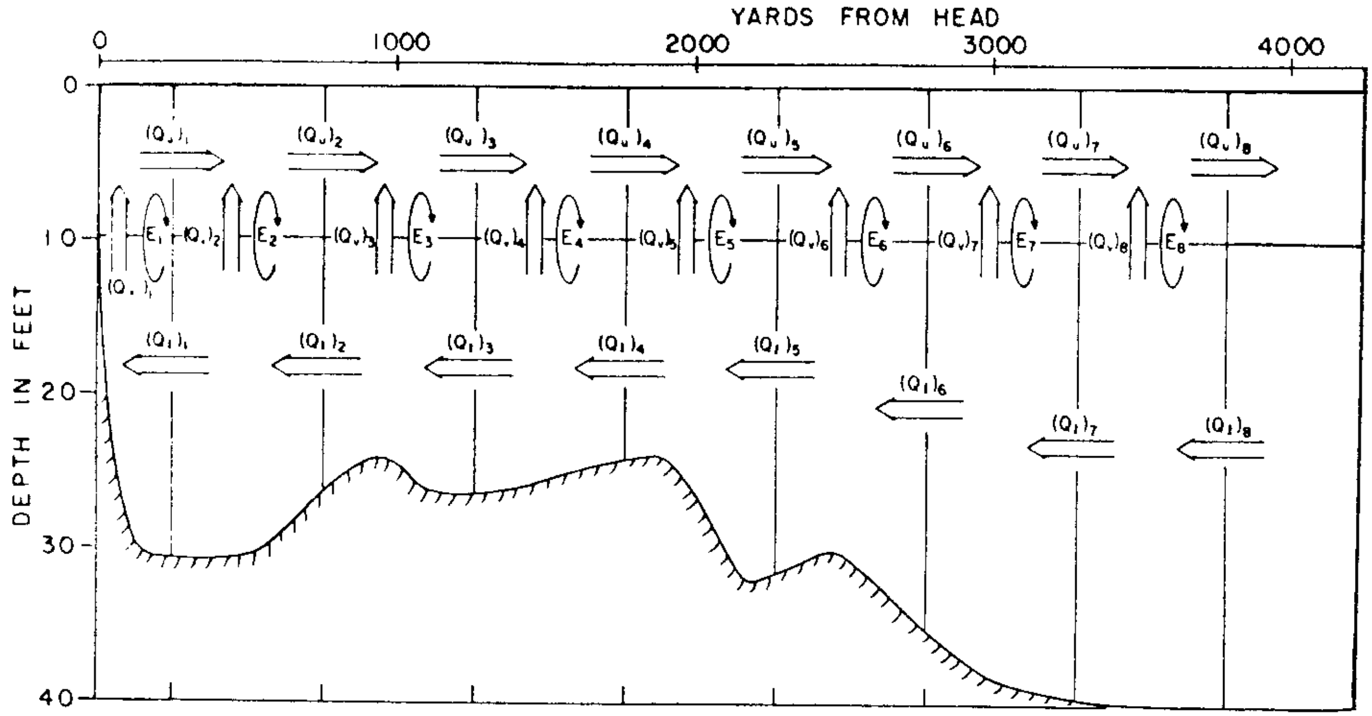


FIGURE 4-3. - SCHEMATIC FOR THE TWO-DIMENSIONAL MODEL OF WILSON AND PRITCHARD INDICATING SEGMENTATION AND PARAMETERS USED TO CHARACTERIZE FLOW. (Q_u, Q_l, and Q_v ARE VOLUME RATES OF NET ADVECTIVE FLOW; E IS THE COEFFICIENT OF VERTICAL DIFFUSIVE EXCHANGE.) (108)

CHAPTER V

DEVELOPMENT OF FINITE DIFFERENCE SCHEMES

As the advantages of high speed digital computers are becoming more widely recognized in the field of mathematical modeling, finite difference techniques are becoming more popular. Finite difference methods are based on the assumption that derivatives can be approximated by using the values of functions at points which are separated by finite increments of space or time. The finite difference approximations that are used to represent partial derivatives can be derived by truncating a Taylor series expansion of a function at a point or, more simply, by visualizing the relationships between slopes and values of a function at separated points. For instance, the following relationships are often used to approximate first and second-order derivatives:

$$\frac{\partial C}{\partial x} = \frac{C_{x+1,t} - C_{x,t}}{\Delta x} + O(\Delta x) \quad \text{(5-1)}$$

$$\frac{\partial C}{\partial x} = \frac{C_{x,t} - C_{x-1,t}}{\Delta x} + O(\Delta x) \quad \text{(5-2)}$$

$$\frac{\partial C}{\partial x} = \frac{C_{x+1,t} - C_{x-1,t}}{2\Delta x} + O[(\Delta x)^2] \quad \text{(5-3)}$$

$$\frac{\partial^2 C}{\partial x^2} = \frac{C_{x-1,t} - 2C_{x,t} + C_{x+1,t}}{(\Delta x)^2} + O[(\Delta x)^2] \quad \text{(5-4)}$$

The x and t subscripts denote location of the value C in a space grid and time grid, respectively, and the $O(\Delta x)$ symbol represents the order of magnitude of the error. The Δx represents a dimensionless distance fraction. Equation 5-1 is a forward form of the finite difference representation and Equation 5-2 is a backward form; Equations 5-3 and 5-4 are central difference forms. By using more neighboring points, an unlimited number of other finite difference approximations can be obtained. However, the above forms are the most compact and were judged to be the most useful ones for this study.

The mass transport equation which we are dealing with in this research is a parabolic partial differential equation. When developing a numerical procedure to approximate this equation, a choice must often be made between and "explicit" formulation and an "implicit" formulation. An approach which expresses one unknown pivotal value directly in terms of know pivotal values is called an "explicit" formulation. An approach in which several unknowns are related to one or several unknowns by an equation is called an "implicit" formulation. The implicit approach requires the solution of a set of simultaneous equations, whereas the explicit approach requires the solution of only one equation for each point.

ONE DIMENSIONAL EQUATIONS WITH CONSTANT COEFFICIENTS

These concepts can be demonstrated by showing the derivation of the finite difference equations for the one-dimensional, constant coefficient case. The partial differential equation can be expressed as

$$\frac{\partial C}{\partial t} = E \frac{\partial^2 C}{\partial x^2} - U \frac{\partial C}{\partial x} - K_d C \quad \ldots \ldots \ldots \ldots \ldots \ldots \ldots \quad (5\text{-}5)$$

where C is concentration (ppm), E is a constant dispersion (ft^2/sec), U is a constant velocity (ft/sec), and K_d is a constant decay rate (/sec).

Explicit Formulation. In terms of the appropriate finite difference equations, an explicit form of this equation is as follows:

$$\frac{C_{x,t+1} - C_{x,t}}{\Delta t} = E\left(\frac{C_{x+1,t} - 2C_{x,t} + C_{x-1,t}}{(\Delta x)^2}\right)$$

$$- U\left(\frac{C_{x+1,t} - C_{x-1,t}}{2\Delta x}\right) - K_d \, (C_{x,t}) \quad \ldots \ldots \quad (5\text{-}6)$$

After rearranging terms, the following equation results:

$$C_{x,t+1} = \frac{E\Delta t}{(\Delta x)^2}\left(C_{x+1,t} - 2C_{x,t} + C_{x-1,t}\right)$$

$$- \frac{U\Delta t}{2\Delta x}\left(C_{x+1,t} - C_{x-1,t}\right)$$

$$- \Delta t \, K_d\left(C_{x,t}\right) + C_{x,t} \quad \ldots \ldots \ldots \ldots \ldots \ldots \quad (5\text{-}7)$$

This equation is used repeatedly at each time step until all the new values for the t+1 set of points are calculated. Since all the concentrations with a t subscript are known before the equation is applied, the calculation of the t+1 concentrations is straightforward except at the boundaries. At the boundaries, certain procedures must be followed to obtain a solution. These boundary conditions are discussed in a later chapter of this report.

Implicit Formulation. Using a more general approach, the mass transport equation can be represented by the following weighted average approximation (after Smith, 91):

$$\frac{C_{x,t+1} - C_{x,t}}{\Delta t} = \frac{E}{(\Delta x)^2} \left\{ \Theta\left(C_{x+1,t+1} - 2C_{x,t+1} + C_{x-1,t+1}\right) + (1 - \Theta)\left(C_{x+1,t} - 2C_{x,t} + C_{x-1,t}\right) \right\}$$
$$- \frac{U}{2\Delta x} \left\{ \Theta\left(C_{x+1,t+1} - C_{x-1,t+1}\right) + (1 - \Theta)\left(C_{x+1,t} - C_{x-1,t}\right) \right\}$$
$$- K_d \left\{ \Theta\left(C_{x,t+1}\right) + (1 - \Theta)\left(C_{x,t}\right) \right\} \quad . \; . \quad (5\text{-}8)$$

This equation uses the same approximations for the derivatives as were used previously for the explicit formulation. If we substitute $\Theta = 0$, we obtain Equation 5-7 of the explicit scheme. If we substitute $\Theta = 1/2$, we obtain what is known as the Crank-Nicolson implicit formulation. If we substitute $\Theta = 1$, we obtain a fully implicit, backward difference formula. These formulations are compared schematically in Figures 5-1, 5-2, and 5-3. For any value of Θ greater than zero, a set of simultaneous equations must be solved.

Using $\Theta = 1/2$ in Equation 5-8, and collecting terms, the following Crank-Nicolson representation is obtained:

$$- A_i\left(C_{x-1,t+1}\right) + B_i\left(C_{x,t+1}\right) - D_i\left(C_{x+1,t+1}\right)$$

$$= A_i\left(C_{x-1,t}\right) - B_i'\left(C_{x,t}\right) + D_i\left(C_{x+1,t}\right) \quad \text{. . . . . (5-9)}$$

where

$$A_i = \left(\frac{E\Delta t}{2(\Delta x)^2} + \frac{U\Delta t}{4\Delta x}\right) \quad \text{. (5-10)}$$

$$B_i = \left(\frac{E\Delta t}{(\Delta x)^2} + \frac{K_d\Delta t}{2} + 1\right) \quad \text{. . . . . (5-11)}$$

$$B_i' = B_i - 2$$

$$D_i = \left(\frac{E\Delta t}{2(\Delta x)^2} - \frac{U\Delta t}{4\Delta x}\right) \quad \text{. (5-12)}$$

for $i = 2, 3, 4, \ldots . N - 3$, where N is the number of points.

For N points, there are N - 2 equations. By arranging the knowns and unknowns for a series of distance steps, the following

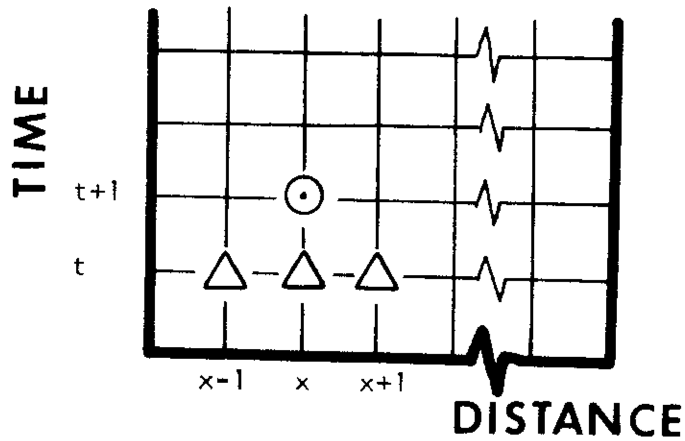


FIGURE 5-1. - EXPLICIT ($\theta = 0$)

△ = KNOWN VALUES

⊙ = UNKNOWN VALUES

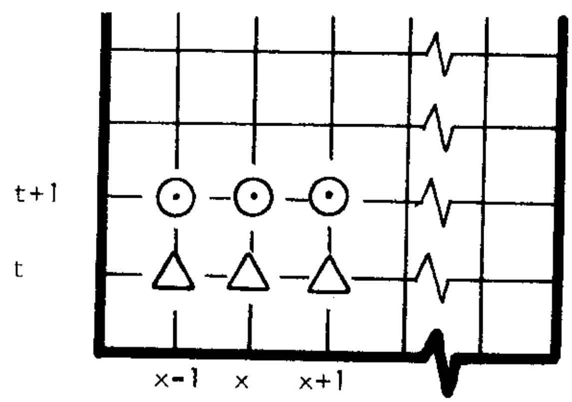


FIGURE 5-2. - CRANK-NICOLSON ($\theta = 1/2$)

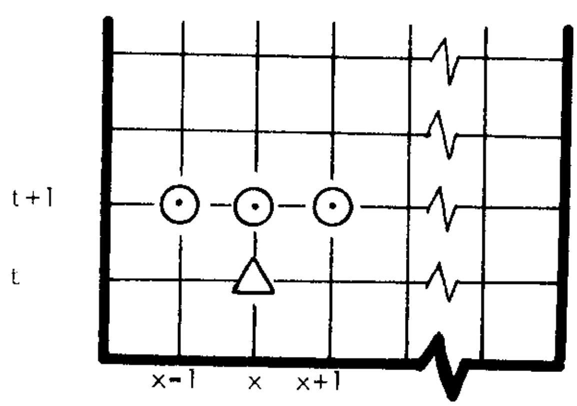


FIGURE 5-3. - FULLY IMPLICIT ($\theta = 1$)

set of simultaneous equations is obtained:

$$
\begin{aligned}
B_1 C_{2,t+1} - D_1 C_{3,t+1} & = W_1 \\
- A_2 C_{2,t+1} + B_2 C_{3,t+1} - D_2 C_{4,t+1} & = W_2 \\
- A_3 C_{3,t+1} + B_3 C_{4,t+1} - D_3 C_{5,\ t+1} & = W_3 \\
\vdots \qquad \vdots \qquad \vdots & \\
- A_{N-2} C_{N-2,t+1} + B_{N-2} C_{N-1} & = W_{N-2}
\end{aligned}
\qquad \ldots\ldots\ldots\ldots (5\text{-}13)
$$

The W_i terms represent the collection of known concentrations and their coefficients, as shown by the right side of Equation 5-9. The values of the terms B_1, D_1, W_1, A_{N-2}, B_{N-2}, and W_{N-2} are dependent upon the boundary conditions which are applied to the problem; these considerations will be discussed in a later chapter of this report.

Fortunately, the coefficients of these simultaneous equations form a tridiagonal matrix which can be solved by a relatively simple Gauss elimination method, also known as the "Thomas Algorithm." If more than three unknowns are used in each equation, a more time-consuming matrix solution technique would be needed; this discourages the use of formulas which require more than three points to approximate derivatives.

The Gauss method begins by using the first equation to eliminate $C_{2,t+1}$ from the second equation. Then the new second equation is used to eliminate $C_{3,t+1}$ from the third equation, and so on. Finally, the new last-but-one equation can be used to eliminate $C_{N-2,t+1}$ from the last equation. This leaves one equation with only one unknown, $C_{N-1,t+1}$. Thus, the unknown $C_{i,t+1}$ values can be found by back-substitution. This procedure is summarized in the following steps (11, 91):

1. Define all A_i, B_i, D_i, and W_i values;
2. Define and calculate alpha's (α_i);

$$\alpha_1 = B_1 \quad \text{(5-14)}$$

$$\alpha_i = B_i - \frac{A_i \ D_{i-1}}{\alpha_{i-1}} \quad \text{(5-15)}$$

for i = 2, 3, ... N-2

3. Define and calculate S's

$$S_1 = W_1 \quad \text{(5-16)}$$

$$S_i = W_i + \frac{A_i \ S_{i-1}}{\alpha_{i-1}} \quad \text{(5-17)}$$

for i = 2, 3, ... N-2

4. Calculate C's starting with $C_{N-1,t+1}$

$$C_{N-1,t+1} = \frac{S_{N-2}}{\alpha_{N-2}} \quad \text{. . . . . (5-18)}$$

$$C_{i,t+1} = \frac{1}{\alpha_i} S_i + D_i \; C_{i+1,t+1} \quad \text{. . . . . (5-19)}$$

for i = N-2, N-3, . . . 2.

ONE DIMENSIONAL EQUATIONS WITH VARYING COEFFICIENTS

The finite difference equations become more complicated when the coefficients are allowed to vary with time and distance. The appropriate partial differential equation which permits varying coefficients and varying width is the following:

$$W_x \frac{\partial C}{\partial t} = - W_x V_x \frac{\partial C}{\partial t} + \frac{\partial}{\partial x}\left(E_x W_x \frac{\partial C}{\partial x}\right) - W_x K D_x C \;\; + \text{(Sources - Sinks)}$$

. . . . (5-20)

The terms in this equation are as defined in Equation 3-24.

Explicit Formulation. - The explicit formulation for this equation uses the approximations listed in Equations 5-1, 5-3, and 5-4. In the following finite difference equations, a z subscript is used in order to be consistent with the two-dimensional formulations; this z is always equal to 1 in the one-dimensional formulas. The one-dimensional, explicit, finite difference approximation to Equation 5-20 is shown in Figure 5-4, as well as

Partial Differential Equation For One-Dimensional Mass Transport:

$$W_x \frac{\partial C}{\partial t} = - W_x V_x \frac{\partial C}{\partial x} + \frac{\partial}{\partial x}\left(E_x W_x \frac{\partial C}{\partial x}\right) - W_x KD_x C + (\text{Sources} - \text{Sinks}) \quad \text{. . . . .} \quad (5\text{-}20)$$

Explicit Finite Difference Formulation:

$$\frac{W_x}{\Delta t}\left(C_{x,z,t+1} - C_{x,z,t}\right) = - \frac{W_x VX_{x,z}}{2\Delta x}\left(C_{x+1,z,t} - C_{x-1,z,t}\right)$$
$$+ \left(\frac{W_{x+1} EX_{x+1,z} + W_x EX_{x,z}}{2\Delta x}\right)\left(\frac{C_{x+1,z,t} - C_{x,z,t}}{\Delta x}\right)$$
$$- \left(\frac{W_x EX_{x,z} + W_{x-1} EX_{x-1,z}}{2\Delta x}\right)\left(\frac{C_{x,z,t} - C_{x-1,z,t}}{\Delta x}\right)$$
$$- W_x KD_x C_{x,z,t} + (\text{Sources} - \text{Sinks}) \quad \text{. . . . .} \quad (5\text{-}21)$$

FIGURE 5-4. - EXPLICIT FINITE DIFFERENCE EQUATIONS FOR ONE-DIMENSIONAL MASS TRANSPORT WITH VARYING COEFFICIENTS

(Continued on next page)

Rearranged Explicit Equation:

$$\begin{aligned} C_{x,z,t+1} = &- C_{x-1,z,t}\left(\frac{\Delta t}{W_x}\right)\left(\frac{W_x VX_{x,z}}{2\Delta x} + \frac{W_x EX_{x,z} + W_{x-1} EX_{x-1}}{2(\Delta x)^2}\right) \\ &- C_{x,z,t}\left(\frac{\Delta t}{W_x}\right)\left(\frac{W_{x+1} EX_{x+1,z} + W_x\ EX_{x,z}}{2(\Delta x)^2} + \frac{W_x EX_{x,z} + W_{x-1} EX_{x-1,z}}{2(\Delta x)^2}\right) \\ &+ C_{x+1,z,t}\left(\frac{\Delta t}{W_x}\right)\left(- \frac{W_x VX_{x,z}}{2\Delta x} + \frac{W_{x+1} EX_{x+1,z} + W_x EX_{x,z}}{2(\Delta x)^2}\right) \\ &+ C_{x,z,t} - KD_x \Delta t C_{x,z,t} + (\text{Sources} - \text{Sinks}) \quad \text{. . . . .} \quad (5\text{-}22) \end{aligned}$$

(End of Figure 5-4)

a more useful rearranged version. The Equation 5-22 is solved repeatedly for each time step as in the explicit equation with constant coefficients discussed previously.

Implicit Formulation. - The Crank-Nicolson implicit finite difference formulation for the one-dimensional mass transport equation with varying coefficients is shown in Figure 5-5; the z subscript is equal to 1 in this one-dimensional framework and terms of the form " - - NEXT" refer to values at time t+1.

This set of equations can be arranged into a tridiagonal matrix. For easier programming, the indexing system is different than the previous one-dimensional, implicit array:

$$B(2)C_{2,z,t+1} + G(2)C_{3,z,t+1} = D(2)$$

$$A(3)C_{2,z,t+1} + B(3)C_{3,z,t+1} + G(3)C_{4,z,t+1} = D(3)$$

$$A(4)C_{3,z,t+1} + B(4)C_{4,z,t+1} + G(4)C_{5,z,t+1} = D(4)$$

$$\vdots \qquad \vdots \qquad \vdots$$

$$A(N-1)C_{N-2,z,t+1} + B(N-1)C_{N-1,z,t+1} = D(N-1)$$

$$\ldots \ldots \ldots \ldots \quad (5\text{-}24)$$

This set of N-2 equations is used for a grid with N horizontal points. The values for B(2), G(2), D(2), A(N-1), B(N-1), and

D(N-1) depend upon the boundary conditions of the problems. The values for other D(I) are found by collecting all the C terms in Equation 5-23 with a subscript of t and their associated coefficients. This yields the values for the arrays A, B, G, and D shown in Figure 5-6. The tridiagonal matrix for this formulation is solved by the Gauss elimination scheme outlined previously in Equations 5-14 through 5-19.

Partial Differential Equation for One Dimensional Mass Transport:

$$W_x \frac{\partial C}{\partial t} = - W_x V_x \frac{\partial C}{\partial x} + \frac{\partial}{\partial x}\left(E_x W_x \frac{\partial C}{\partial x} \right) - W_x K D_x C + (\text{Sources - Sinks}) \quad \ldots\ldots\ldots\ldots \quad (5\text{-}20)$$

Implicit Crank-Nicolson Finite Difference Formulation:

$$\begin{aligned}
\frac{W_x}{\Delta t}\left(C_{x,z,t+1} - C_{x,z,t} \right) = & - \frac{W_x VXNEXT_{x,z}}{2} \left\{ \frac{C_{x+1,z,t+1} - C_{x-1,z,t+1}}{2\Delta x} \right\} \\
& - \frac{W_x VX_{x,z}}{2} \left\{ \frac{C_{x+1,z,t} - C_{x-1,z,t}}{2\Delta x} \right\} \\
& + \frac{1}{2}\left(\frac{W_{x+1} EXNEXT_{x+1,z} + W_x EXNEXT_{x,z}}{2\Delta x} \right) \left\{ \frac{C_{x+1,z,t+1} - C_{x,z,t+1}}{\Delta x} \right\} \\
& + \frac{1}{2}\left(\frac{W_{x+1} EX_{x+1,z} + W_x EX_{x,z}}{2\Delta x} \right) \left\{ \frac{C_{x+1,z,t} - C_{x,z,t}}{\Delta x} \right\} \\
& - \frac{1}{2}\left(\frac{W_x EXNEXT_{x,z} + W_{x-1} EXNEXT_{x-1,z}}{2\Delta x} \right) \left\{ \frac{C_{x,z,t+1} - C_{x-1,z,t+1}}{\Delta x} \right\}
\end{aligned}$$

FIGURE 5-5. - IMPLICIT CRANK-NICOLSON FINITE DIFFERENCE EQUATION FOR ONE-DIMENSIONAL MASS TRANSPORT WITH VARYING COEFFICIENTS

(Continued on next page)

$$- \frac{1}{2}\left(\frac{W_x EX_{x,z} + W_{x-1} EX_{x-1,z}}{2\Delta x}\right)\left\{\frac{C_{x,z,t} - C_{x-1,z,t}}{\Delta x}\right\}$$

$$- \frac{1}{2} W_x KD_x \left\{C_{x,z,t+1}\right\} - \frac{1}{2} W_x KD_x \left\{C_{x,z,t}\right\} + (\text{Sources - Sinks}) \qquad \text{(5-23)}$$

(End of Figure 5-5)

$$A(X) = \left(\frac{\Delta t}{W_x}\right)\left(-\frac{W_x VXNEXT_{x,z}}{4\Delta x} - \frac{W_x EXNEXT_{x,z} + W_{x-1} EXNEXT_{x-1,z}}{4(\Delta x)^2}\right) \quad (5\text{-}25)$$

$$B(X) = 1. + \left(\frac{\Delta t}{W_x}\right)\left\{\left(\frac{W_{x+1} EXNEXT_{x+1,z} + 2W_x EXNEXT_{x,z} + W_{x-1} EXNEXT_{x-1,z}}{4(\Delta x)^2}\right) + \frac{W_x KD_x}{2}\right\} \quad (5\text{-}26)$$

$$G(X) = \left(\frac{\Delta t}{W_x}\right)\left(\frac{W_x VXNEXT_{x,z}}{4\Delta x} - \frac{W_{x+1} EXNEXT_{x+1,z} + W_x EXNEXT_{x,z}}{4(\Delta x)^2}\right) \quad (5\text{-}27)$$

$$\begin{aligned} D(x) = {} & C_{x,z,t} + \left(\frac{\Delta t}{W_x}\right)\left(-\frac{W_x VX_{x,z}}{4\Delta x}\right)\left\{C_{x+1,z,t} - C_{x-1,z,t}\right\} \\ & + \left(\frac{\Delta t}{W_x}\right)\left(\frac{W_{x+1} EX_{x+1,z} + W_x EX_{x,z}}{4(\Delta x)^2}\right)\left\{C_{x+1,z,t} - C_{x,z,t}\right\} \\ & - \left(\frac{\Delta t}{W_x}\right)\left(\frac{W_x EX_{x,z} + W_{x-1} EX_{x-1,z}}{4(\Delta x)^2}\right)\left\{C_{x,z,t} - C_{x-1,z,t}\right\} - \frac{KD_x}{2}\left\{C_{x,z,t}\right\} \end{aligned} \quad (5\text{-}28)$$

FIGURE 5-6. - COEFFICIENTS FOR TRIDIAGONAL MATRIX

TWO DIMENSIONAL EQUATIONS WITH VARYING COEFFICIENTS

The most complicated form of the mass transport equation which has been programmed in this research allows for two dimensional convective-dispersion with variable coefficients:

$$W_x \frac{\partial C}{\partial t} = \frac{\partial}{\partial x}\left(E_x W_x \frac{\partial C}{\partial x}\right) + \frac{\partial}{\partial z}\left(E_z W_x \frac{\partial C}{\partial z}\right)$$

$$- W_x V_x \frac{\partial C}{\partial x} - W_x V_z \frac{\partial C}{\partial z} - W_x K D_x C + (\text{Sources} - \text{Sinks}) \quad . \quad (5\text{-}29)$$

This equation was derived and discussed in Chapter III.

Explicit Formulation. The two dimensional explicit formulation takes the same approach as Equations 5-21 and 5-22 of the one-dimensional explicit method, but in addition includes terms in the vertical z direction. The two-dimensional explicit finite difference equation is shown in Figure 5-7. A more convenient rearranged version is shown also.

The Equation 5-30 can be visualized as the computational molecule shown in Figure 5-8.

Partial Differential Equation for Two Dimensional Mass Transport:

$$W_x \frac{\partial C}{\partial t} = - W_x V_x \frac{\partial C}{\partial x} - W_x V_z \frac{\partial C}{\partial z} + \frac{\partial}{\partial x}\left(E_x W_x \frac{\partial C}{\partial t}\right) + \frac{\partial}{\partial z}\left(E_z W_x \frac{\partial C}{\partial z}\right) - W_x KD_x C + (\text{Sources} - \text{Sinks}) \quad \ldots\ldots (5\text{-}29)$$

Explicit Finite Difference Formulation:

$$\frac{W_x}{\Delta t}\left\{C_{x,z,t+1} - C_{x,z,t}\right\} = - \frac{W_x V_x}{2\Delta x}\left\{C_{x+1,z,t} - C_{x-1,z,t}\right\} - \frac{W_x V_z}{2\Delta z}\left\{C_{x,z+1,t} - C_{x,z-1,t}\right\}$$

$$+ \left(\frac{W_{x+1} EX_{x+1,z} + W_x EX_{x,z}}{2\Delta x}\right)\left\{\frac{C_{x+1,z,t} - C_{x,z,t}}{\Delta x}\right\}$$

$$- \left(\frac{W_x EX_{x,z} + W_{x-1} EX_{x-1,z}}{2\Delta x}\right)\left\{\frac{C_{x,z,t} - C_{x-1,z,t}}{\Delta x}\right\}$$

$$+ \left(\frac{W_x EZ_{x,z+1} + W_x EZ_{x,z}}{2\Delta z}\right)\left\{\frac{C_{x,z+1,t} - C_{x,z,t}}{\Delta z}\right\}$$

FIGURE 5-7. - EXPLICIT FINITE DIFFERENCE EQUATIONS FOR TWO-DIMENSIONAL MASS TRANSPORT WITH VARYING COEFFICIENTS

(Continued on next page)

$$- \left(\frac{W_x EZ_{x,z} + W_x E_{x,z-1}}{2\Delta z} \right) \left\{ \frac{C_{x,z,t} - C_{x,z-1,t}}{\Delta z} \right\}$$

$$- \; W_x KD_x C_{x,z,t} + (\text{Sources} - \text{Sinks}) \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \quad (5\text{-}30)$$

Rearranged Explicit Equation:

$$C_{x,z,t+1} = C_{x-1,z,t} \left(\frac{\Delta t}{W_x} \right) \left(\frac{W_x VX_{x,z}}{2\Delta x} + \frac{W_x EX_{x,z} + W_{x-1} EX_{x-1,z}}{2(\Delta x)^2} \right)$$

$$- C_{x,z,t} \left(\frac{\Delta t}{W_x} \right) \left(\frac{W_{x+1} EX_{x+1,z} + W_x EX_{x,z}}{2(\Delta x)^2} + \frac{W_x EX_{x,z} + W_{x-1} E_{x-1,z}}{2(\Delta x)^2} \right.$$

$$\left. + \frac{W_x EZ_{x,z+1} + W_x EZ_{x,z}}{2(\Delta z)^2} + \frac{W_x EZ_{x,z} + W_x EZ_{x,z-1}}{2(\Delta z)^2} \right)$$

$$+ C_{x+1,z,t} \left(\frac{\Delta t}{W_x} \right) \left(- \frac{W_x VX_{x,z}}{2\Delta x} + \frac{W_{x+1} EX_{x+1,z} + W_x EX_{x,z}}{2(\Delta x)^2} \right)$$

$$+ C_{x,z-1,t} \left(\frac{\Delta t}{W_x} \right) \left(\frac{W_x VZ_{x,z}}{2\Delta z} + \frac{W_x EZ_{x,z} + W_x EZ_{x,z-1}}{2(\Delta z)^2} \right)$$

FIGURE 5-7. - (Continued)

$$+ \; C_{x,z+1,t} \left(\frac{\Delta t}{W_x}\right)\left(- \frac{W_x VZ_{x,z}}{2\Delta z} + \frac{W_x EZ_{x,z+1} + W_x EZ_{x,z}}{2(\Delta z)^2}\right)$$

$$+ \; C_{x,z,t} - \Delta t \; KD_x C_{x,z,t} + (\text{Sources} - \text{Sinks}) \quad \text{(5-31)}$$

(End of Figure 5-7)

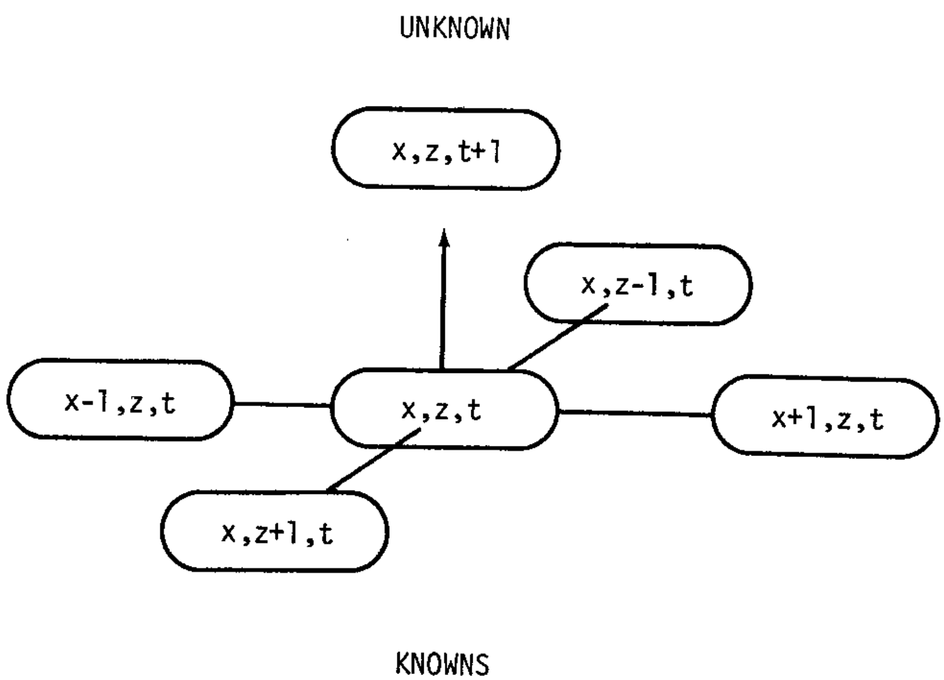


FIGURE 5-8. - TWO-DIMENSIONAL EXPLICIT COMPUTATION MOLECULE

Implicit Formulation. In order to avoid solving a large, two-dimensional matrix by a time-consuming iterative technique, a procedure known as the "implicit alternating-direction method" can be used. This approach, as discussed by Peaceman and Rachford (74), and later modified by Douglas (18), permits the application of the efficient tridiagonal matrix solution explained earlier in this report to a two-dimensional, Crank-Nicolson implicit approximation to the mass transport equation. Additional discussions of this approach are found in other sources (11, 28, 91). This method requires a two-phase procedure. During the initial phase, a Crank-Nicolson scheme is applied only to concentrations in the x-direction. The new concentrations, called CSTAR, which result from the solution of this system of equations are used in a second phase where the Crank-Nicolson scheme is applied to concentrations in the z-direction. The values of CSTAR can be considered as initial approximations which are used to find a more accurate solution during the second phase. This procedure can be expressed schematically as follows:

First Phase.

$$\frac{W_x}{\Delta t}\left(CSTAR_{x,z} - C_{x,z,t}\right)$$

$= \frac{1}{2}$ (Derivatives and coefficients of C in x-direction at time t+1)

+ $\frac{1}{2}$ (Derivatives and coefficients of C in x-direction at time t)

+ (Derivatives and coefficients of C in z-direction at time t)

. . . . (5-32)

Second Phase.

$$\frac{W_x}{\Delta t}\left(C_{x,z,t+1} - C_{x,z,t}\right)$$

= $\frac{1}{2}$ (Derivatives and coefficients of CSTAR in x direction).

+ $\frac{1}{2}$ (Derivatives and coefficients of C in x-direction at time t)

+ $\frac{1}{2}$ (Derivatives and coefficients of C in z-direction at time t+1)

+ $\frac{1}{2}$ (Derivatives and coefficients of C in z-direction at time t)

. . . . (5-33)

This two-phase approach requires the solution of two tri-diagonal matrices. The arrays are of the same form as Equation

5-24. The equations for these arrays are shown in Figures 5-9 and 5-10. As in the previous implicit cases, values for B(2), G(2), D(2), A(N-1), B(N-1), and D(N-1) are dependent upon the conditions at the boundaries. The computational molecule for the two-dimensional Crank-Nicolson scheme is shown in Figure 5-11. An example of a two-dimensional grid is shown in Figure 5-12. Values for dispersion, velocity, decay, and other variables must be defined for each grid point.

Where NX equals the number of points in the x-direction and for X = 3, 4, 5, . . . NX-2:

$$A(X) = \left(\frac{\Delta t}{W_x}\right)\left(-\frac{W_x VXNEXT_{x,z}}{4\Delta x} - \frac{W_x EXNEXT_{x,z} + W_x EXNEXT_{x-1,z}}{4(\Delta x)^2}\right) \quad (5\text{-}34)$$

$$B(X) = 1. + \left(\frac{\Delta t}{W_x}\right)\left(\frac{W_{x+1} EXNEXT_{x+1,z} + W_x EXNEXT_{x,z}}{4(\Delta x)^2} + \frac{W_x EXNEXT_{x,z} + W_{x-1} EXNEXT_{x-1,z}}{4(\Delta x)^2}\right) \quad (5\text{-}35)$$

$$G(X) = \left(\frac{\Delta t}{W_x}\right)\left(\frac{W_x VXNEXT_{x,z}}{4\Delta x} - \frac{W_{x+1} EXNEXT_{x+1,z} + W_x EXNEXT_{x,z}}{4(\Delta x)^2}\right) \quad (5\text{-}36)$$

$$D(X) = C_{x,z,t} + \left(\frac{\Delta t}{W_x}\right)\Bigg[-\frac{W_x VX_{x,z}}{4\Delta x}\left\{C_{x+1,z,t} - C_{x-1,z,t}\right\}$$

$$- \frac{2W_x VZ_{x,z}}{4\Delta z}\left\{C_{x,z+1,t} - C_{x,z-1,t}\right\} + \left(\frac{W_{x+1} EX_{x+1,z} + W_x EX_{x,z}}{4(\Delta x)^2}\right)\left\{C_{x+1,z,t} - C_{x,z,t}\right\}$$

$$- \left(\frac{W_x EX_{x,z} + W_{x-1} EX_{x-1,z}}{4(\Delta x)^2}\right)\left\{C_{x,z,t} - C_{x-1,z,t}\right\} + 2\left(\frac{W_x EZ_{x,z+1} + W_x EZ_{x,z}}{4(\Delta z)^2}\right)$$

FIGURE 5-9. - ARRAY VALUES FOR THE FIRST PHASE OF THE IMPLICIT PROCEDURE

(Continued on next page)

$$\left\{C_{x,z+1,t} - C_{x,z,t}\right\}$$

$$- 2\left(\frac{W_x EZ_{x,z} + W_x EZ_{x,z-1}}{4(\Delta z)^2}\right)\left\{C_{x,z,t} - C_{x,z-1,t}\right\}\Bigg] \quad \text{(5-37)}$$

(End of Figure 5-9)

Where NZ equals the number of points in the z-direction and for Z = 3, 4, 5, . . . NZ-2:

$$A(Z) = \left(\frac{\Delta t}{W_x}\right)\left(-\frac{W_x VZNEXT_{x,z}}{4\Delta z} - \frac{W_x EZNEXT_{x,z} + W_x EZNEXT_{x,z-1}}{4(\Delta z)^2}\right) \quad (5\text{-}38)$$

$$B(Z) = 1. + \left(\frac{\Delta t}{W_x}\right)\left(\frac{W_x EZNEXT_{x,z+1} + W_x EZNEXT_{x,z}}{4(\Delta z)^2} + \frac{W_x EZNEXT_{x,z-1} + W_x EZNEXT_{x,z}}{4(\Delta z)^2}\right) \quad (5\text{-}39)$$

$$G(Z) = \left(\frac{\Delta t}{W_x}\right)\left(\frac{W_x VZNEXT_{x,z}}{4\Delta z} - \frac{W_x EZNEXT_{x,z+1} + W_x EZNEXT_{x,z}}{4(\Delta z)^2}\right) \quad (5\text{-}40)$$

$$D(Z) = C_{x,z,t} + \left(\frac{\Delta t}{W_x}\right)\Bigg[-\frac{W_x VXNEXT_{x,z}}{4\Delta x}\left\{CSTAR_{x+1,z} - CSTAR_{x-1,z}\right\}$$

$$-\frac{W_x VX_{x,z}}{4\Delta x}\left\{C_{x+1,z,t} - C_{x-1,z,t}\right\} - \frac{W_x VZ_{x,z}}{4\Delta z}\left\{C_{x,z+1,t} - C_{x,z-1,t}\right\}$$

$$+\left(\frac{W_x EXNEXT_{x+1,z} + W_x EXNEXT_{x,z}}{4(\Delta x)^2}\right)\left\{CSTAR_{x+1,z} - CSTAR_{x,z}\right\}$$

$$-\left(\frac{W_x EXNEXT_{x,z} + W_x EXNEXT_{x-1,z}}{4(\Delta x)^2}\right)\left\{CSTAR_{x,z} - CSTAR_{x-1,z}\right\}$$

(Continued on next page)

FIGURE 5-10. - ARRAY VALUES FOR THE SECOND PHASE OF THE IMPLICIT PROCEDURE

$$+\left(\frac{W_{x+1}EX_{x+1,z} + W_x EX_{x,z}}{4(\Delta x)^2}\right)\left\{C_{x+1,z,t} - C_{x,z,t}\right\} - \left(\frac{W_x EX_{x,z} + W_{x-1}EX_{x-1,z}}{4(\Delta x)^2}\right)$$

$$\left\{C_{x,z,t} - C_{x-1,z,t}\right\}$$

$$+\left(\frac{W_x EZ_{x,z+1} + W_x EZ_{x,z}}{4(\Delta z)^2}\right)\left\{C_{x,z+1,t} - C_{x,z,t}\right\} - \left(\frac{W_x EZ_{x,z} + W_x EZ_{x,z-1}}{4(\Delta z)^2}\right)$$

$$\left\{C_{x,z,t} - C_{x,z-1,t}\right\} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (5\text{-}41)$$

For mass transport problems with decay and other source/sink terms, this additional equation is used after the solution of the tridiagonal matrix:

$$C^{FINAL}_{x,z,t+1} = C_{x,z,t+1} - \left(\frac{\Delta t KD_x}{2}\right)\left\{C_{x,z,t+1} + C_{x,z,t}\right\} + (\text{Sources} - \text{Sinks}) \quad \ldots\ldots\ldots \quad (5\text{-}42)$$

(End of Figure 5-10)

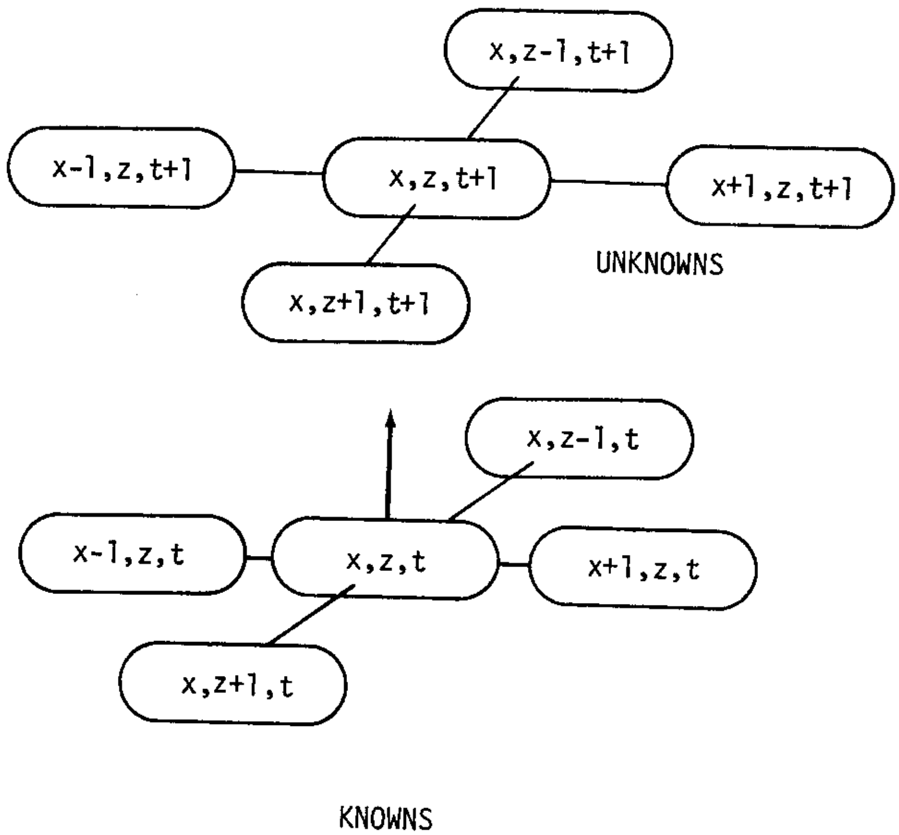
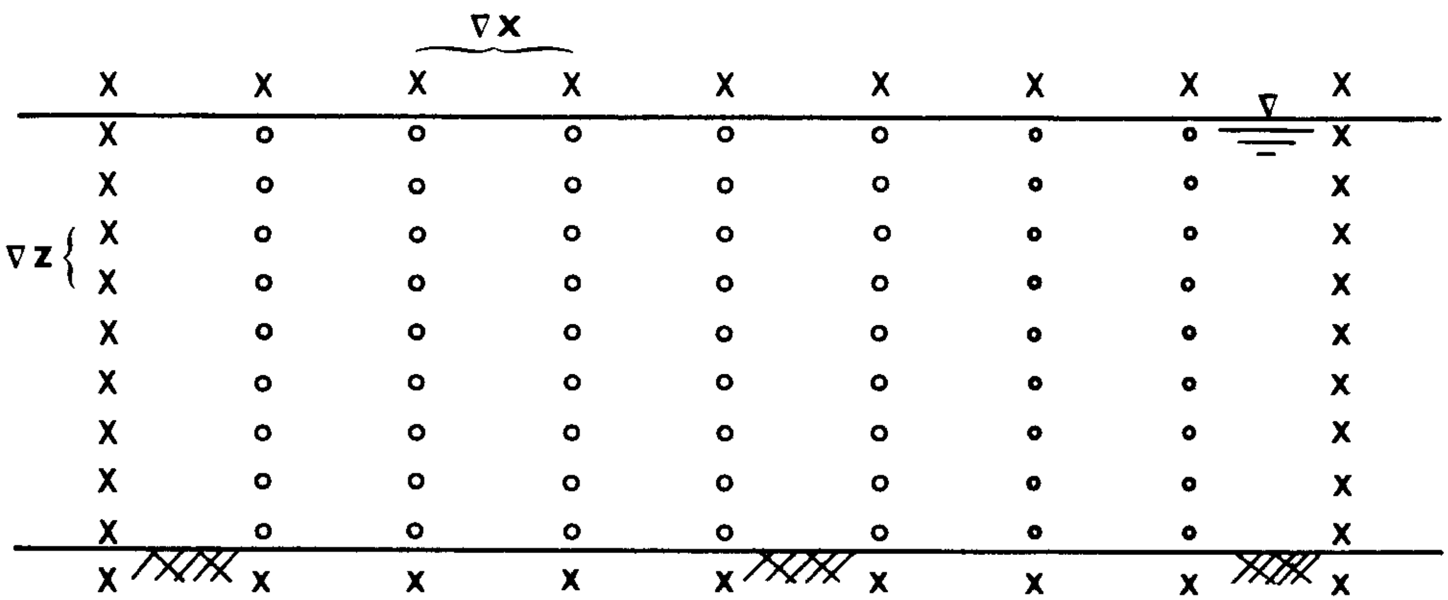


FIGURE 5-11. - TWO-DIMENSIONAL IMPLICIT COMPUTATIONAL MOLECULE



o CALCULATED CONCENTRATIONS

X EXTRAPOLATED VALUES AND IMAGE POINTS

________ WATER BOUNDARIES

FIGURE 5-12. - TWO-DIMENSIONAL VERTICAL GRID

CHAPTER VI

STABILITY CONSIDERATIONS

Instability is an important problem which must be dealt with when applying finite difference techniques. Instability refers to the tendency of errors to grow without limit as the computation progresses. These errors can be generated by inexact initial conditions, inexact boundary conditions, oscillations generated in early time steps, or round-off errors. Stability, on the other hand, is the tendency to damp out errors as the computation progresses. If a solution is unstable, it becomes hopelessly inaccurate. On the other hand, a stable solution is not necessarily an accurate solution. The accuracy of a solution must be investigated after the stability criteria has been established.

EXPLICIT METHOD

Several sophisticated techniques are available for analyzing the stability of explicit finite difference formulations. However, in this study, the following simple rule was found to be the most convenient: stability is assured when the coefficient for each concentration value is greater than zero. Thus, examination of the Equation 5-31 leads to the following inequalities if variations with distance are considered to be small, and if the subscripts x, z, and t are dropped:

$$EX \frac{\Delta t}{(\Delta x)^2} - VX \frac{\Delta t}{2\Delta x} > 0 \quad \text{(6-1)}$$

$$EZ \frac{\Delta t}{(\Delta z)^2} - VZ \frac{\Delta t}{2\Delta z} > 0 \quad \text{(6-2)}$$

$$1\text{-}2EX \frac{\Delta t}{(\Delta x)^2} - 2EZ \frac{\Delta t}{(\Delta z)^2} - \Delta t\ KD > 0 \quad \text{(6-3)}$$

According to Equations 6-1 and 6-2, dispersion coefficients in a certain direction can not be set equal to zero when a velocity exists in that direction. Solution of these equations, in order, leads to the following stability criteria:

$$\Delta x < \frac{2EX}{VX} \quad \text{(6-4)}$$

$$\Delta z < \frac{2EZ}{VZ} \quad \text{(6-5)}$$

$$\Delta t < \frac{(\Delta x)^2 (\Delta z)^2}{2EX(\Delta z)^2 + 2EZ(\Delta x)^2 + KD(\Delta x)^2 (\Delta z)^2} \quad \text{(6-6)}$$

The corresponding stability criteria for one dimensional analysis is

$$\Delta x < \frac{2EX}{VX} \quad \text{(6-7)}$$

$$\Delta t < \frac{(\Delta x)^2}{2EX + (\Delta x)^2 KD} \quad \text{(6-8)}$$

One-dimensional stability restrictions can be examined rapidly by plotting coaxial graphs on log-log paper for Equations 6-7 and 6-8. Figures 6-1 and 6-2 display such graphs for a useful range of velocity and dispersion values.

When applying these criteria to the data in this study, the minimum numerators and maximum denominators were used. For example, when applying Equation 6-6, the maximum values for EX, EZ, and KD were used; likewise, when applying Equation 6-4, the minimum value for EX and the maximum value for VX were used. This approach leads to a conservative estimate of stability criteria, since maximum velocities and minimum dispersions rarely occur at the same point.

IMPLICIT METHOD

The Crank-Nicolson implicit method is not limited by the same stability criteria as the explicit formulation. However, literature and computation experience demonstrate that severe oscillations and inaccuracies can occur when the time increment exceeds twice the time criteria of the explicit approach. Thus, for the Crank-Nicolson formulation, the time increment should not exceed the following value when a new range of data is being analyzed:

$$\Delta t < \frac{2(\Delta x)^2 \, (\Delta z)^2}{2EX(\Delta z)^2 + 2EZ(\Delta x)^2 + KD(\Delta x)^2(\Delta z)^2} \quad . \ . \ . \ . \ . \ . \ . \ . \ . \quad (6\text{-}9)$$

The corresponding one-dimensional limit is

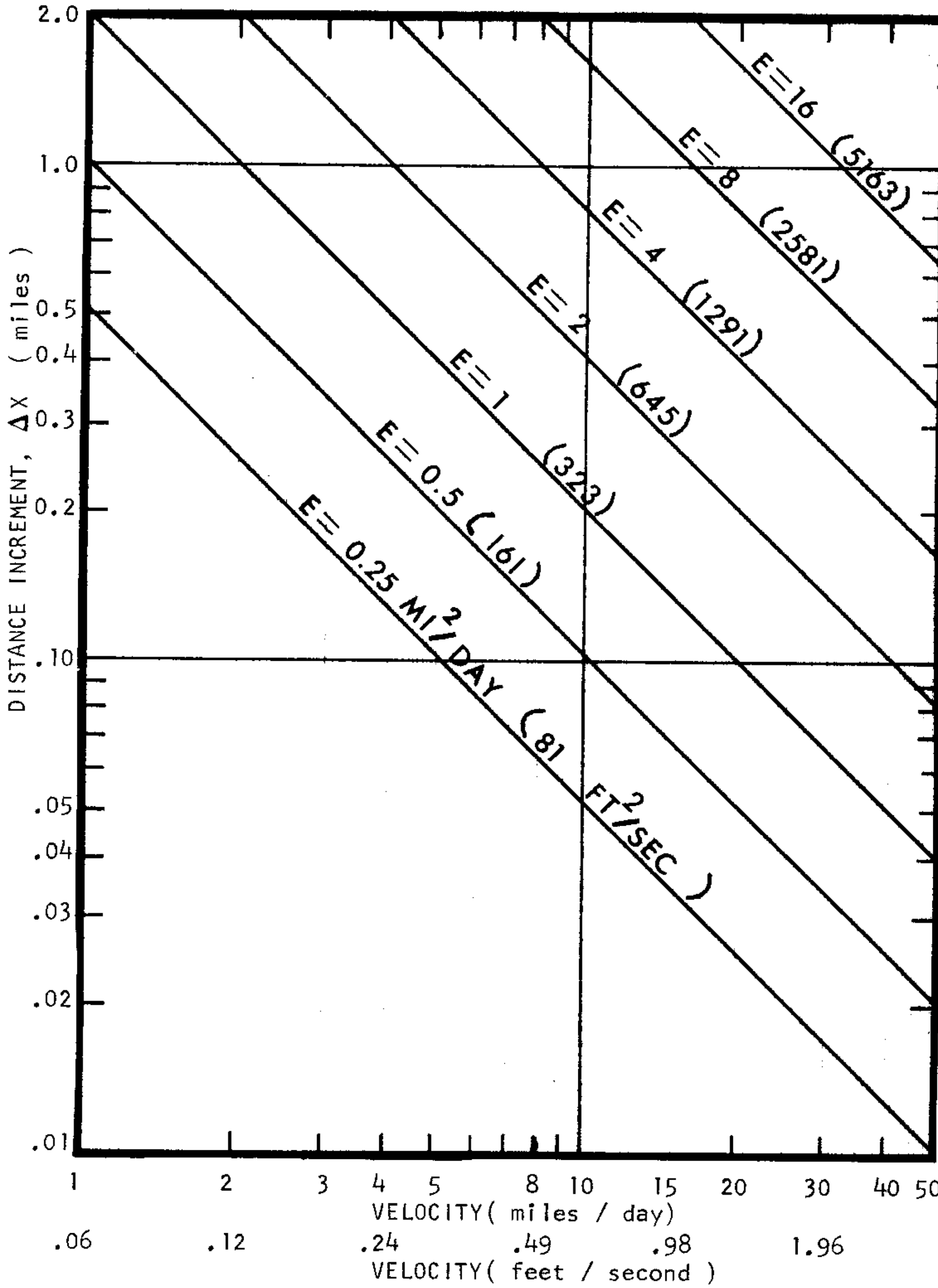


FIGURE 6-1. - DISTANCE INCREMENTS FOR STABILITY

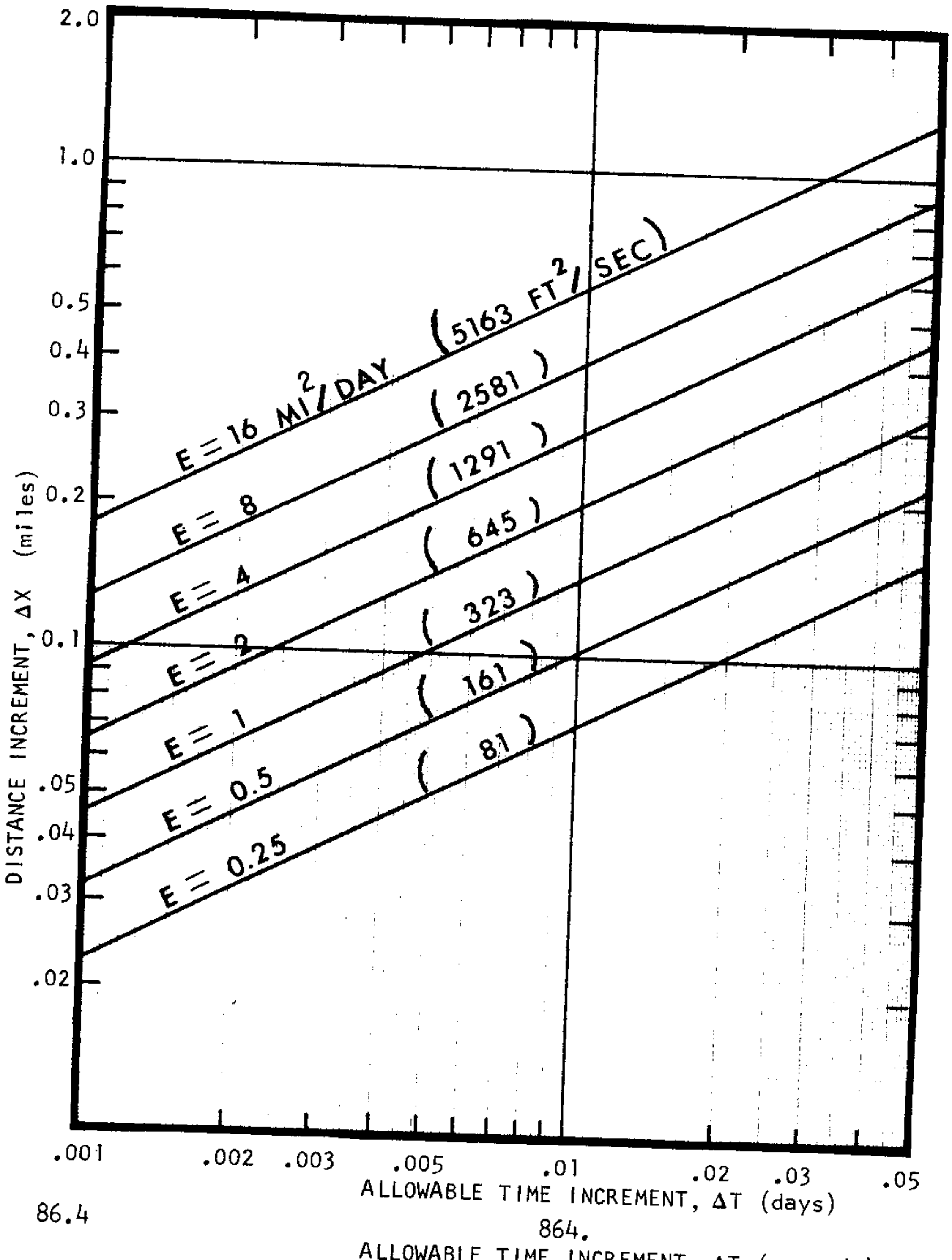


FIGURE 6-2. - TIME INCREMENTS FOR STABILITY

$$\Delta t < \frac{2(\Delta x)^2}{2EX + (\Delta x)^2 KD} \quad \text{(6-10)}$$

CHAPTER VII

BOUNDARY CONDITIONS

Both the explicit and the implicit methods applied in this research use central difference formulas to represent first and second derivatives. As shown in Equations 5-3 and 5-4, these formulas require grid concentrations to the left and right of the point being analyzed. This requirement presents certain difficulties at the grid boundaries; concentration profiles must be extrapolated one point past the boundary to allow for the application of the central difference formula to a boundary concentration. Since inaccurate boundary concentrations can severely distort the entire concentration profile, care must be taken to choose the proper extrapolation technique. Profile distortion becomes a significant problem when concentration values are changing rapidly near the boundary or when initial concentration profiles are placed near the borders of the grid. One solution to these problems is to enlarge the grid to such a size that the concentrations at the boundary have an insignificant effect on the whole profile; this is the best solution when unlimited computer time is available. When economy requires that the grid must be as compact as possible, more care must be taken to choose a proper extrapolation technique.

Numerous extrapolation techniques were investigated during the development of computer programs for this research; the most useful techniques are reviewed below.

BOUNDARIES ALLOWING NO TRANSFER

A common condition encountered in estuary modeling is a situation where no transfer of mass is allowed across the air-water surface or through the channel bottom. This condition is handled easily by including a set of "image" points at the border of the grid. At the air-water interface, these image concentrations are set equal to the corresponding concentrations at the surface. At the bottom of the channel, the image concentrations are set equal to the corresponding bottom-most concentrations.

For the two-dimensional explicit formulation, the upper image concentration is labelled $C_{x,1,t+1}$, and the surface concentration is labelled $C_{x,2,t+1}$. Thus, after Equation 5-31 has been applied to all the internal points, $C_{x,1,t+1}$ is set equal to the just-calculated value for $C_{x,2,t+1}$. The same procedure is followed for the bottom concentrations.

For the two-dimensional implicit formulation, a corresponding change must be made to the coefficient matrix represented by Equations 5-38 through 5-40. At the z-boundaries, the appropriate equations for a grid with NZ grid points would be

$$A(2)\ C_{x,1,t+1} + B(2)\ C_{x,2,t+1} + G(2)\ C_{x,3,t+1} = D(2) \quad . \ . \ (7\text{-}1)$$

$$A(NZ-1)\ C_{x,NZ-2,t+1},\ +\ B(NZ-1)\ C_{x,\ NZ-1,t+1}$$

$$+\ G(NZ-1)\ C_{x,NZ,t+1} = D(NZ-1)\ \ldots\ldots\ldots\ldots\ldots \quad (7\text{-}2)$$

Since $C_{x,1,t+1}$ and $C_{x,NZ,t+1}$ are image concentrations, these equations become

$$[A(2) + B(2)]\ C_{x,2,t+1} + G(2)\ C_{x,3,t+1} = D(2)\ \ldots\ldots \quad (7\text{-}3)$$

$$A(NZ-1)\ C_{x,NZ-2,t+1} + [B(NZ-1) + G(NZ-1)]$$

$$C_{x,NZ-1,t+1} = D(NZ-1)\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (7\text{-}4)$$

Equations 7-3 and 7-4 become the first and last equations for the tridiagonal matrix.

These image concentrations block the dispersion of mass across a boundary. Loss of mass by advection is prevented by setting the vertical velocity term equal to zero at the surface and the bottom.

CONSTANT SLOPE EXTRAPOLATION

A convenient method of extrapolating profiles is to assume that a straight line can be extended from the two concentrations nearest the boundary. This approach is fairly accurate for profiles which have flattened out; it is extremely inaccurate for profiles whose slopes are changing rapidly near the boundaries. This extrapolation technique is frequently used for the Crank-Nicolson implicit formulation because, unfortunately, only the two adjacent concentrations can be manipulated if the form of the tridiagonal matrix is to be preserved.

For the explicit method, concentrations are extended in the horizontal direction according to the following equations:

$$C_{1,z,t+1} = 2\, C_{2,z,t+1} - C_{3,z,t+1} \quad \ldots \ldots \ldots \quad (7\text{-}5)$$

$$C_{NX,z,t+1} = 2\, C_{NX-1,z,t+1} - C_{NX-2,z,t+1} \quad \ldots \ldots \ldots \quad (7\text{-}6)$$

Similar equations are applied if extrapolation in the vertical direction is needed.

The coefficient Equations 5-34 through 5-36 for the implicit method are represented as follows at the boundaries in the x-direction:

$$[2\, A(2) + B(2)]\, C_{2,z,t+1} + [G(2) - A(2)]\, C_{3,z,t+1}$$

$$= D(2) \quad \ldots \ldots \ldots \quad (7\text{-}7)$$

$$[A(NX-1) - G(NX-1)]\, C_{NX-2,z,t+1} + [B(NX-1) + 2G(NX-1)]$$

$$C_{NX-1,z,t+1} = D(NX-1) \quad \ldots \ldots \ldots \quad (7\text{-}8)$$

Extrapolations in the z-direction are represented by similar equations.

EXPONENTIAL EXTRAPOLATIONS

In an idealized estuary, where dispersion and velocity are constant throughout, the following two-dimensional equation can be applied to the instantaneous release of a slug of pollutant (16):

$$C_{x,z,t} = \left(\frac{m}{4\pi t\sqrt{EX \cdot EZ}}\right) \exp\left[-\frac{(x - VX \cdot t)^2}{4EX \cdot t} - \frac{(z - VZ \cdot t)^2}{4EZ \cdot t}\right] \quad \text{(7-9)}$$

where m equals amount per unit depth, and x and z are the distances from the source point. By comparing this equation for concentrations at x - Δx, x + Δx, and x, the following relationships are found:

$$C_{x-\Delta x,z,t} = C_{x,z,t} \cdot \exp\left[\frac{2x\Delta x - (\Delta x)^2 - 2\Delta x \cdot VX \cdot t}{4EX \cdot t}\right] \quad \text{(7-10)}$$

$$C_{x+\Delta x,z,t} = C_{x,z,t} \cdot \exp\left[-\frac{2x\Delta x - (\Delta x)^2 + 2\Delta x \cdot VX \cdot t}{4EX \cdot t}\right] \quad \text{(7-11)}$$

Similar expressions can be derived for concentrations at $z+\Delta z$ and $z-\Delta z$.

Equations 7-10 and 7-11 have obvious potential for extrapolating water quality profiles, especially if an idealized estuary is being modeled. For an estuary where average velocities and dispersion coefficients are used throughout, this extrapolation provides the exact relationship between the final two concentrations on the grid at the appropriate boundaries. For an estuary with varying characteristics, these equations can give good approximations if the proper velocities and dispersion coefficients are provided.

For the explicit method, Equations 7-10 and 7-11 can be applied directly. For the implicit method, these equations can be easily represented in the coefficient matrix. For example, in the x-direction, the first line and last line of the tridiagonal matrix can be represented by Equations 7-12 and 7-13:

$$\left[-A(2) \cdot \exp\,(EXPON1) + B(2)\right] C_{2,z,t+1}$$

$$+ G(2)\, C_{3,z,t+1} = D(2) \quad \ldots \ldots \ldots \ldots \ldots \quad (7\text{-}12)$$

$$A(NX-1)\, C_{NX-2,z,t} +$$

$$+ \left[B(NX-1) + G(NX-1) \cdot \exp(EXPON2)\right] C_{NX-1,z,t+1}$$

$$= D(NX-1) \quad \ldots \ldots \ldots \ldots \ldots \ldots \quad (7\text{-}13)$$

where EXPON1 stands for the bracketed expression in Equation 7-10 and EXPON2 stands for the bracketed expression in Equation 7-11.

Similar expressions can be inserted for extrapolation in the z-direction.

These equations can be applied only to instantaneous slug releases; formulas could be devised also for other types of profiles, such as a steady-state profile from a constant source.

CONTINUED FRACTIONS AND INVERTED DIFFERENCES

Several interpolation techniques are available in the IBM Scientific Subroutine package (48). With small modifications, these programs can be used for extrapolation. Three of these techniques were investigated during this study: the Aitkens scheme of Lagrange interpolation (ALI); the Aitkens scheme of Hermite interpolation (AHI); and the continued fractions and inverted differences scheme (ACFI). All of these techniques are explained in Hildebrand (43). These techniques were designed to interpolate equations containing low powers of the variable; thus, they are very accurate when applied to gradually changing profiles. However, serious inaccuracies can result when the slope of the profile is changing rapidly near the boundary: extrapolated concentrations can become negative or can become greater than the concentration at the adjacent point.

The continued fractions and inverted differences scheme (ACFI) was found to be the most reliable of these methods and was modified to be used as an extrapolation technique for the explicit formulation. An advantage of this technique is that the programmer can choose the number of internal points to be included in the extrapolation formula.

EXTRAPOLATION BY PROPORTIONS

A final extrapolation method that was found to be useful was the simple proportion:

$$\frac{C_{3,z,t+1}}{C_{2,z,t+1}} = \frac{C_{2,z,t+1}}{C_{1,z,t+1}} \quad \ldots\ldots\ldots\ldots\ldots \quad (7\text{-}14)$$

For example, when this formula is rearranged to extrapolate in the x-direction, it becomes

$$C_{1,z,t+1} = \frac{(C_{2,z,t+1})^2}{C_{3,z,t+1}} \quad \ldots\ldots\ldots\ldots\ldots \quad (7\text{-}15)$$

This technique can be fairly accurate for extrapolating a curve whose slope is changing rapidly. The proportion technique and the continued fractions technique were both applied to the explicit method; neither was applied to the implicit method.

CHAPTER VIII

MATHEMATICAL MODELS AND SUPPORTING COMPUTER PROGRAMS

The estuary mass transport equation was programmed in finite difference form at several levels of complexity as part of this research. The simplest form deals with an idealized, one-dimensional estuary with constant values for dispersion, velocity, decay, and cross-section. The most complicated form is applicable to an estuary with characteristics that can be varied in two dimensions throughout the estuary. This progression toward increasing complexity brought about a more economical use of computer time. The numerical methods that are used in this study require considerable computational experience in order to establish accuracy and stability guidelines; this experience is more easily gained by working with less complex programs which take less computer time.

The basic computer programs that have been developed during this study are outlined below; they are written in FORTRAN IV and are discussed in detail in the sections which follow:

One-Dimensional Numerical Analysis With Constant Coefficients. - Two finite difference models were developed for evaluating mass transport in an idealized, one-dimensional estuary:

1. IDEAL - I : explicit method;
2. IDEAL - II: implicit method.

One- and Two-Dimensional Numerical Analysis With Varying

Coefficients. - Three finite difference models were developed for evaluating mass transport and oxygen transport in an estuary with varying characteristics:

1. MASSTRANS - I : mass transport by the explicit method;
2. MASSTRANS - II: mass transport by the implicit method;
3. OXTRANS - I : mass and oxygen transport by the explicit method.

Auxiliary Programs. - Several auxiliary computer programs were developed for accuracy and stability analysis:

1. STABLE - I : one-dimensional stability analysis;
2. STABLE - II : two-dimensional stability analysis;
3. EXACT - I : one-dimensional, time-changing, analytical solutions;
4. EXACT - II : two-dimensional, time-changing, analytical solutions for instantaneous releases;
5. PROFILE - I : steady-state, one-dimensional solutions for non-conservative substances; and
6. PROFILE - II : steady-state, one-dimensional solutions for BOD and dissolved oxygen profiles.

IDEAL - I

This program applies an explicit finite difference method to analyze mass transport in a one-dimensional estuary. The model is useful in evaluating transport in a flume or well-mixed estuary where velocity, dispersion, decay, and cross-sectional area can be considered constant throughout.

The primary use of this program is to test the accuracy of the explicit finite difference method as compared with analytical, closed-form solutions for the estuary mass transport equation. The program can analyze instantaneous releases or steady-state profiles.

The basis of computation is Equation 5-7. IDEAL-I uses an exponential extrapolation or constant slope extrapolation at the boundaries. The program calculates and points out the appropriate stability criteria according to Equations 6-7 and 6-8.

IDEAL - II

This program has the same purpose and application as IDEAL - I except that the Crank-Nicolson implicit finite differences method is applied.

The basis of computation is Equations 5-9 through 5-19. IDEAL - II uses an exponential extrapolation or constant slope extrapolation at the boundaries. The program calculates and prints out the appropriate criteria to avoid severe oscillations; this criteria is based on Equation 6-10.

MASSTRANS - I

This program was developed to analyze estuaries whose characteristics do not vary significantly with width. The width of the estuary may vary throughout but the concentrations of dissolved materials at each cross-section are considered to be unchanging in the lateral direction. All physical and hydrodynamic characteristics may vary with time and distance in the longitudinal and vertical directions. The program can be applied with equal ease to the two horizontal directions, allowing depth to vary rather than width.

Concentration profiles can be calculated by this program for continuous or instantaneous releases. The input data may include grid dimensions, distance increments, time increments, widths, loading parameters, velocities, dispersion coefficients, decay rates, sedimentation rates, and other source and sink terms. The time increment can be increased or decreased at any time during the calculation of the concentration profile. The distances between grid points can be increased at any time; a routine within the program will choose the appropriate values from the previously calculated profile and will place these values in the desired locations in the new grid system. The number of grid points may be increased or decreased at any time. Likewise, at any time during the calculation, a new set of data for physical and hydrodynamic conditions may be read into the program.

MASSTRANS-I applies an explicit finite difference method to analyze mass transport in an estuary. A user of this program must be familiar with the limitations on accuracy and stability inherent in this type of numerical procedure. A subroutine within the program prints out the proper increments to insure stability and terminates the program if this criteria is violated by the input parameters. Another subroutine extrapolates concentrations at the boundaries; several methods can be used for these extrapolations depending on the type of profile being analyzed and the choice of the user. A subroutine is also included which prints out error messages and terminates the program if certain inconsistencies occur in the input data.

This program was developed primarily to analyze partially stratified estuaries which have been dredged out to a fairly constant depth at the centerline of the channel; these estuaries are common in the Gulf Coast region. Application of this program to partially stratified estuaries with variable depths would require moderate revisions to the program and would make the program estuary-dependent.

MASSTRANS-I also can be applied to estuaries which are well-mixed in the vertical direction. This option allows for varying width or varying depth and uses most of the routines available to the two-dimensional analysis.

The basis for one-dimensional applications is Equation 5-22. The basis for two-dimensional applications is Equation 5-31. Stability criteria is defined by Equations 6-4 through 6-8.

MASSTRANS-II

The essential difference between this program and MASSTRANS-I is that a Crank-Nicolson implicit finite difference method is used to solve the equations for estuary mass transport. Otherwise, the programs operate and are applied in the same manner.

The basis for one-dimensional applications is Equations 5-20 through 5-28. The basis for two-dimensional applications is Equations 5-32 through 5-42. Criteria to avoid severe oscillations is generated through the use of Equations 6-9 and 6-10, but the program is not terminated if this criteria is violated.

OXTRANS-I

This program analyzes the relationship between a primary pollutant and a dissolved gas in a partially stratified estuary. The principle application of OXTRANS-I is to the interaction of biochemical oxygen demand (BOD) and dissolved oxygen (DO).

OXTRANS-I performs two basic calculations during each time step. The first calculation is to determine the profile for the primary pollutant, such as BOD, by the same methods available in MASSTRANS-I. The second calculation uses information from the first calculation to determine the values for the dissolved gas.

The input data for OXTRANS-I is the same as for MASSTRANS-I except that additional data is required for reaeration co-efficients, aerobic decay rates, anaerobic decay rates, and other oxygen demands or sources. During the calculations for BOD and DO, the program automatically switches to anaerobic decay rates whenever the concentration of oxygen is zero. Reaeration terms can be applied to any point on the grid and, thus, the effects of mechanical aeration at any point or points can be investigated.

At the option of the user, the secondary calculations can be bypassed and the program can be operated as MASSTRANS-I.

STABLE-I and STABLE-II

These programs allow for a quick analysis of stability for the explicit finite difference method; they can process a large number of data combinations for dispersion, velocity, and decay with each run.

STABLE-I is applied to one-dimensional stability calculations through the use of Equations 6-7 and 6-8. STABLE-I generates the stability criteria for a large number of combinations of velocity and dispersion after one data set is read in.

STABLE-II is applied to two-dimensional stability calculations through the use of Equations 6-4 through 6-6. STABLE-II computes the stability criteria for each set of data that is read into the program.

EXACT-I and EXACT-II

These programs calculate concentration profiles which change with time according to known solutions to the partial differential equation for mass transfer in an estuary. These solutions can be applied only to an idealized estuary where velocities, dispersion coefficients, decay rate, and cross-sectional area are constant throughout. EXACT-I and EXACT-II were used extensively in this study to evaluate the accuracy of the finite difference methods.

Solutions for three types of loading conditions were programmed for EXACT-I: a constant concentration defined at a source point; an instantaneous release; a continuous discharge at a point.

Constant Concentration. - The following solution is applicable to a condition where the concentration, C_o, is constant at and above a certain point in the estuary at time zero (51):

$$\frac{C}{C_o} = \frac{1}{2} \exp\left(\frac{u \cdot x}{E}\right) \operatorname{erfc}\left(\frac{x + u \cdot t}{2\sqrt{E \cdot t}}\right) + \frac{1}{2} \operatorname{erfc}\left(\frac{x - u \cdot t}{2\sqrt{E \cdot t}}\right) \quad . \ . \ . \ (8\text{-}1)$$

where erfc is the coerror function, u is net velocity, E is the dispersion coefficient, x is distance, and t is time.

A second solution is applicable when the velocity equals zero:

$$\frac{C}{C_o} = \operatorname{erfc}\left(\frac{x}{2\sqrt{E \cdot t}}\right) \quad . \ (8\text{-}2)$$

Instantaneous Release. - The following equation represents the concentration resulting from an instantaneous release of amount **m** per unit cross-sectional area (16).

$$C = \left(\frac{m}{\sqrt{4\pi \cdot E \cdot t}}\right) \exp\left[-\frac{(x - u \cdot t)^2}{4 \cdot E \cdot t}\right] - K_d \cdot t \quad . \ . \ . \ . \ . \ . \ . \ (8\text{-}3)$$

This equation was used extensively to test the accuracy of IDEAL-I and IDEAL-II.

Continuous Release. - The following equation is applicable to a conservative substance released at a rate m per unit cross-sectional area per unit time into an estuary with no net velocity (13).

$$C = \left(\frac{m \cdot t^{\frac{1}{2}}}{\sqrt{\pi} \cdot E}\right) \exp\left[-\frac{x^2}{4 \cdot E \cdot t}\right] - \left(\frac{mx}{2E}\right) \operatorname{erfc}\left[\frac{x}{2\sqrt{E \cdot t}}\right] \quad . \; . \; . \; . \; (8\text{-}4)$$

EXACT-II was used extensively to test the accuracy of MASSTRANS-I and MASSTRANS-II. The appropriate equation for two-dimensional convective-dispersion from an instantaneous release of amount m per unit depth is the following:

$$C = \left(\frac{m}{4\pi t\sqrt{E_x \cdot E_y}}\right) \exp\left[-\frac{(x-V_x t)^2}{4 \cdot E_x \cdot t} - \frac{(y-V_y t)^2}{4 \cdot E_x \cdot E_y} - K_d \cdot t\right] . \quad (8\text{-}5)$$

where x and y represent perpendicular axes, V is velocity, and E is dispersion.

PROFILE I and PROFILE II

PROFILE-I calculates the steady-state profile for a non-conservative substance which is continuously released into an idealized estuary at the rate of W pounds per day. PROFILE-II calculates the steady-state profiles for BOD, oxygen deficit, and dissolved oxygen (DO). These programs were used to test steady-state applications of MASSTRANS-I, MASSTRANS-II, and OXTRANS-I. The following equations were programmed (67):

$$BOD = C_o \cdot \exp\left[\frac{u \cdot x}{2E}(1 \pm m_1)\right] \quad \text{(8-6)}$$

$$DEFICIT = \frac{K_d \cdot W}{(K_2 - K_d)Q}\left\{\frac{1}{m_1}\exp\left[\frac{u \cdot x}{2E}(1 \pm m_1)\right] - \frac{1}{m_2}\exp\left[\frac{u \cdot x}{2E}(1 \pm m_2)\right]\right\} \quad \text{(8-7)}$$

$$DO = C_{SAT} - DEFICIT \quad \text{(8-8)}$$

in which

$$m_1 = \sqrt{1 + \frac{4K_d \cdot E}{u^2}} \quad \text{(8-9)}$$

$$m_2 = \sqrt{1 + \frac{4K_2 \cdot E}{u^2}} \quad \text{(8-10)}$$

$$C_o = \frac{W}{Qm_1} \quad \text{(8-11)}$$

and Q equals flowrate in pounds per day, K_d equals the decay rate,

K_2 equals the reaeration rate, u is the net velocity, and C_{SAT} is the saturation value for dissolved oxygen. The positive sign refers to values of $x < 0$ and the negative sign refers to values of $x > 0$. Care must be taken to insure consistency of units. An equation similar to Equation 8-7 was applied for the special case where $K_d = K_2$.

CHAPTER IX

ANALYSIS OF ACCURACY

Establishing the accuracy of a modeling technique is of utmost importance before that technique can be used as a basis for engineering decisions. The most straightforward and appropriate manner in which to test the accuracy of finite difference estuary models is to compare their results with exact solutions to the partial differential equations for mass transport in an estuary; these solutions are readily available for certain idealized types of estuary behavior. Such a comparison should be a requirement for any type of estuary model before the model is applied to the oftentimes inexact data of real estuaries.

The finite difference models were tested for the following cases for which exact, analytical solutions are available: one-dimensional, steady-state profiles for non-conservative substances and dissolved oxygen; one-dimensional, instantaneous release of a slug load; and two-dimensional, instantaneous release of a slug load. The accuracy characteristics of the two-dimensional models were found to be the same as the one-dimensional models and, therefore, most of the testing was done on the one-dimensional models in order to conserve computer time.

STEADY-STATE PROFILES

Steady-state profiles are obtained in the finite difference models by setting a concentration at a point and then iterating the calculations until an unchanging profile develops. This is an inefficient way of obtaining such a profile but, as it turns out, a very accurate way.

The exact analytical solutions for steady-state profiles were obtained through the computer programs PROFILE-I and PROFILE-II which apply Equations 8-6 through 8-8.

Profiles for Non-Conservative Substances. - Excellent accuracy for steady-state profiles can be obtained by finite difference methods, even when large distance increments are used. Comparison between an exact solution and the profile calculated by the explicit method is shown in Table 9-1. The input values were as follows: dispersion coefficient = 10.0 mi^2/day; velocity = 1.0 mile / day; decay rate (base e) = 0.25/day. For the explicit program, IDEAL-I, the distance increment was 2.0 miles and the time increment was 0.01 days. The implicit solution for the same increments differed from the explicit solution by only one unit in the fourth decimal place. A constant slope extrapolation was applied at the downstream boundary and this appears to significantly affect the accuracy of only the last four concentrations.

In the Table 9-1, two columns are included which refer to the time at which steady-state was achieved in the finite difference model at different points. The first of these columns refers to the time it took for the solution to stabilize to four

TABLE 9-1.-COMPARISON BETWEEN EXACT AND FINITE DIFFERENCE SOLUTIONS

Distance (miles)	Exact (ppm)	Finite Difference (ppm)	Days till steady-state	
			4 places	1 place
0	10.0000	10.0000		
2	7.9321	7.9398	23.40	4.96
4	6.2919	6.3040	24.30	7.76
6	4.9908	5.0052	25.25	8.81
8	3.9588	3.9740		
10	3.1402	3.1552	27.60	11.86
12	2.4908	2.5051		
14	1.9758	1.9888		
16	1.5672	1.5789		
18	1.2431	1.2534		
20	0.9861	0.9948	30.95	
22	0.7822	0.7893		
24	0.6204	0.6258		
26	0.4921	0.4956		
28	0.3904	0.3915		
30	0.3096	0.3078		
32	0.2456	0.2396		
34	0.1948	0.1830	32.15	
36	0.1548	0.1342		
38	0.1226	0.0894		
40	0.0972	0.0446	32.65	

decimal points. The second of these columns refers to the time needed to reach a point in the calculations where the values remain essentially unchanged to one decimal point. Since the time increment was 0.01 days, one day in these columns would represent 100 time iterations.

The implicit method tended to take 1.75 times as much computer time as the explicit method for the same time and distance increments. When a time step of 0.1 days was used, the steady-state solutions were only slightly less accurate for both finite difference methods than for a time increment of 0.01 days.

BOD - DO Profiles. - OXTRANS-I, the explicit finite difference model for dissolved oxygen, was investigated to determine its ability to calculate steady-state profiles for BOD and dissolved oxygen. Table 9-2 shows a comparison between the exact solution and the finite difference calculation for BOD and DO profiles based on the following input data: dispersion coefficient = 400 ft^2/sec. (1.24 mi^2/day); velocity = 0.2 ft/sec (3.27 miles/day); decay rate (base e) = 0.23/day; reaeration rate = 0.1/day; C_{SAT} = 8.0 ppm; and C_0 = 8.817. OXTRANS-I used a distance increment of 0.5 miles and a time increment of 0.05 days.

OXTRANS-I provided excellent accuracy downstream from the source point at Mile 3.5. The extrapolation at the downstream boundary was based on a continued fractions and inverted differences scheme and this method provided outstanding accuracy at the final

TABLE 9-2.-BOD AND DO PROFILES

Mile	BOD (ppm)		DO (ppm)	
	Exact	OXTRANS-I	Exact	OXTRANS-I
0.0	0.001	0.000	8.00	8.00
0.5	0.003	0.001	8.00	8.00
1.0	0.010	0.003	8.00	8.00
1.5	0.039	0.014	7.99	8.00
2.0	0.152	0.068	7.98	7.99
2.5	0.588	0.345	7.93	7.96
3.0	2.276	1.745	7.81	7.85
3.5	8.817	8.817	7.56	7.55
4.0	8.520	8.520	7.27	7.27
4.5	8.233	8.233	7.00	7.00
5.0	7.956	7.956	6.75	6.74
5.5	7.688	7.688	6.50	6.50
6.0	7.429	7.430	6.27	6.26
6.5	7.179	7.179	6.05	6.04
7.0	6.937	6.938	6.84	5.84
7.5	6.704	6.704	5.64	5.64
8.0	6.478	6.478	5.46	5.45
8.5	6.260	6.260	5.28	5.28
9.0	6.049	6.050	5.12	5.11
9.5	5.845	5.846	4.96	4.96
10.0	5.649	5.649	4.81	4.81
10.5	5.458	5.459	4.67	4.67
11.0	5.275	5.275	4.54	4.54
11.5	5.097	5.098	4.42	4.42
12.0	4.925	4.926	4.31	4.30
12.5	4.759	4.760	4.20	4.19
13.0	4.599	4.600	4.10	4.09
13.5	4.444	4.445	4.00	4.00
14.0	4.295	4.295	3.92	3.91
14.5	4.150	4.151	3.84	3.83
15.0	4.010	4.011	3.76	3.76
15.5	3.875	3.876	3.69	3.69
16.0	3.745	3.746	3.63	3.63
16.5	3.619	3.619	3.57	3.57
17.0	3.497	3.498	3.52	3.51
17.5	3.379	3.380	3.47	3.47
18.0	3.265	3.266	3.43	3.42
18.5	3.156	3.156	3.39	3.38

point, Mile 18.5. The steady-state solution at Mile 4.0 stabilized after 60 time iterations and the solution at Mile 18.0 stabilized after 180 time iterations.

The profile upstream of the source point is fairly inaccurate, but this problem did not adversely affect the downstream profile. The inaccuracy in the upstream profile was caused by attempting to use a large distance increment in an area where the slope of the concentration profile was changing rapidly with distance. This rapid change in the slope of the steady-state profile is produced when the velocity is in the opposite direction from dispersion. This inaccuracy can be corrected by using a smaller distance increment in areas where concentration profiles have steep slopes. Tables 9-3 and 9-4 demonstrate the change in accuracy when the distance increment is changed from 0.5 miles to 0.1 miles. The time increment was changed to 0.0025 days to insure stability. Even better accuracy would be attained if the distance increment were decreased even further.

These results demonstrate that the choice of a distance increment has a limiting effect on the accuracy which can be obtained. For steady-state profiles, several distance increments should be tried in order to insure that an accurate profile has been calculated.

ONE-DIMENSIONAL INSTANTANEOUS RELEASE OF A SLUG LOAD

An important application of the finite difference models that

TABLE 9-3.-ACCURACY FOR BOD PROFILES FOR SEVERAL DISTANCE INCREMENTS

Mile	Exact	OXTRANS-I $\Delta x = 0.5$ $\Delta t = 0.05$	OXTRANS-I $\Delta x = 0.1$ $\Delta t = 0.0025$
0.0	0.001	0.000	-
0.5	0.003	0.001	-
1.0	0.010	0.003	-
1.5	0.039	0.014	0.029
2.0	0.152	0.068	0.140
2.5	0.588	0.345	0.571
3.0	2.276	1.745	2.252
3.5	8.817	8.817	8.817

TABLE 9-4.-ACCURACY FOR DISSOLVED OXYGEN PROFILES FOR SEVERAL DISTANCE INCREMENTS

Mile	Exact (ppm)	OXTRANS-I $\Delta x = 0.5$ $\Delta t = 0.05$	OXTRANS-I $\Delta x = 0.1$ $\Delta t = 0.0025$
0.0	8.00	8.00	-
0.5	8.00	8.00	-
1.0	8.00	8.00	-
1.5	7.99	8.00	8.00
2.0	7.98	7.99	7.98
2.5	7.93	7.96	7.94
3.0	7.81	7.85	7.83
3.5	7.56	7.55	7.55

have been developed in this study will be to calculate the concentration profiles resulting from slug loads released into an estuary. Such loading might result from spills of polluting materials or dye releases. Therefore, a considerable amount of time has been spent during this study to evaluate the ability of the models to predict accurately the concentration profiles produced by slug loads.

Finite difference methods are generally regarded as being accurate for diffusion predictions, as in heat transfer problems. However, when velocity is also included as a parameter, a phenomenon known as "numerical dispersion" results. This phenomenon caused significant difficulties in the finite difference methods used in the San Francisco Bay-Delta model (72). Numerical dispersion is related to the fact that the distance increment used in a finite difference model is rarely a multiple of the velocity at which the material is moving.

In this study, numerical dispersion was found to produce distortions and severe inaccuracies if improper choices were made for distance increments and time increments. However, excellent accuracy was achieved and the effects of numerical dispersion were minimized when a small enough distance increment was chosen; for the dispersion coefficients and velocities used in this study, good accuracy was obtained for horizontal distance increments between 0.1 miles and 0.25 miles.

When comparing finite difference solutions with exact analytical solutions, initial concentration values for the model must be chosen to represent accurately the mass of the slug load. These initial values can be chosen in two ways: either by obtaining an initial distribution of values at several points from the analytical solution or by calculating an equivalent "slug" value for a single starting value in the model. The analytical solution, as represented by Equation 8-3 , calculates the concentration profile for a mass m per unit cross-sectional area. For a single initial concentration, the finite difference model "sees" this mass as a triangle, as shown in Figure 9-1, below:

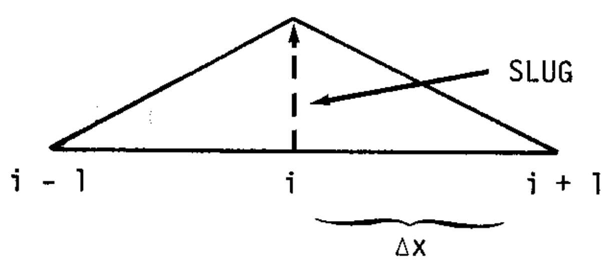


FIGURE 9-1.-SLUG LOAD "SEEN" BY FINITE DIFFERENCE MODEL

The value for SLUG is equal to $m/\Delta x$, where the m used for accuracy investigations was 10^7. Thus, for a distance increment of 0.25 miles (1,320 feet), SLUG = $10^7/1320$ = 7575.8. For a distance increment of 0.10 miles (528 feet), SLUG = $10^7/528$ = 18939.39. The Δx must be expressed in feet.

Analysis of Dispersion Without Velocity. - When using a single value of a finite difference model, distortions develop in the early calculations of the profile. When a sufficiently small distance increment is chosen along with a proper time increment, these distortions are rapidly damped out. Tables 9-5 and 9-6 demonstrate the ability of the models to converge on the correct solution. For this example, the dispersion coefficient = 1 mi^2/day (322.7 ft^2/sec), the time increment = 0.01 days, the distance increment = 0.25 miles, and SLUG = 7575.8 at time = 0.0 days. The velocity and decay rate were set equal to zero. The profiles were essentially symmetrical; therefore only half of the profiles are shown.

Effect of Velocity on the Concentration Profile from a Slug Load. - Table 9-7 shows the effects of a velocity of 5 miles/day (.306 ft/sec) on the distribution described in the previous example. Velocity distorts the concentration profiles; however, this distortion is minor when proper time and distance increments are chosen, except at the leading and trailing edges. A comparison between the exact profile after one day elapsed time and the profile predicted by the implicit method is shown in Figure 9-2, for a distance increment of 0.25 miles. Although the explicit method is more accurate for dispersion alone, the implicit method appears to be more accurate when velocity is included.

Error Analysis for a Slug Load. - The finite difference models

TABLE 9-5.-DISTRIBUTION FROM A SLUG LOAD AT TIME = 0.2 DAYS

Mile	Exact	Explicit	Implicit
0.00	1194.7	1196.1	1219.9
0.25	1104.9	1105.7	1120.4
0.50	874.0	873.8	870.6
0.75	591.4	590.5	577.0
1.00	342.3	341.6	329.8
1.25	169.4	169.3	164.5
1.50	71.75	71.88	72.46
1.75	25.98	26.15	28.49
2.00	8.05	8.15	10.10
2.25	2.13	2.17	3.26
2.50	0.48	0.49	0.96
2.75	0.09	0.09	0.26
3.00	0.02	0.02	0.04

TABLE 9-6.-DISTRIBUTION FROM A SLUG LOAD AT TIME = 1.0 DAYS

Mile	Exact	Explicit	Implicit
0.00	534.3	534.3	536.2
0.25	526.0	526.1	527.8
0.50	501.9	502.0	503.3
0.75	464.2	464.2	464.9
1.00	416.1	416.1	416.1
1.25	361.5	361.5	360.9
1.50	304.4	304.4	303.3
1.75	248.5	248.4	247.0
2.00	196.6	196.5	195.1
2.25	150.7	150.6	149.3
2.50	112.0	112.0	110.8
2.75	80.66	80.63	79.77
3.00	56.31	56.29	55.69

TABLE 9-7.-ONE DIMENSIONAL CONVECTIVE-DISPERSION OF A SLUG LOAD AFTER ONE DAY

Mile	Exact	IDEAL-II (Implicit)	IDEAL-I (Explicit)	
		ΔT = .01 Days	ΔT = .01	ΔT = .0025
-5.00 (Input point)	1.03	0.54	0.40	0.51
-4.75	1.90	1.11	0.82	1.03
-4.50	3.38	2.16	1.62	2.02
-4.25	5.84	4.07	3.07	3.81
-4.00	9.79	7.34	5.61	6.89
-3.75	15.88	12.73	9.87	12.00
-3.50	25.00	21.21	16.74	20.08
-3.25	38.10	33.97	27.35	32.32
-3.00	56.31	52.34	43.05	50.03
-2.75	80.66	77.55	65.30	74.54
-2.50	112.0	110.6	95.46	106.9
-2.25	150.7	151.9	134.5	147.7
-2.00	196.6	201.0	182.7	196.7
-1.75	248.5	256.1	239.2	252.5
-1.50	304.4	315.7	301.9	312.6
-1.25	361.5	375.1	364.5	373.4
-1.00	416.1	430.5	431.4	430.8
-0.75	464.2	477.6	488.4	480.2
-0.50	501.9	512.4	533.4	517.3
-0.25	526.0	532.1	561.9	539.0
-0.00 (Peak)	534.3	535.1	571.1	543.4
0.25	526.0	521.6	560.1	530.4
0.50	501.9	492.9	530.1	501.5
0.75	464.2	452.1	484.3	459.6
1.00	416.1	402.5	427.0	408.4
1.25	361.5	348.2	363.5	352.1
1.50	304.4	292.9	298.7	294.7
1.75	248.5	239.6	237.1	239.5
2.00	196.6	190.8	181.7	189.1
2.25	150.7	147.9	134.5	145.2
2.50	112.0	111.7	96.15	108.4
2.75	80.66	81.95	66.71	78.58
3.00	56.31	57.21	46.57	54.86

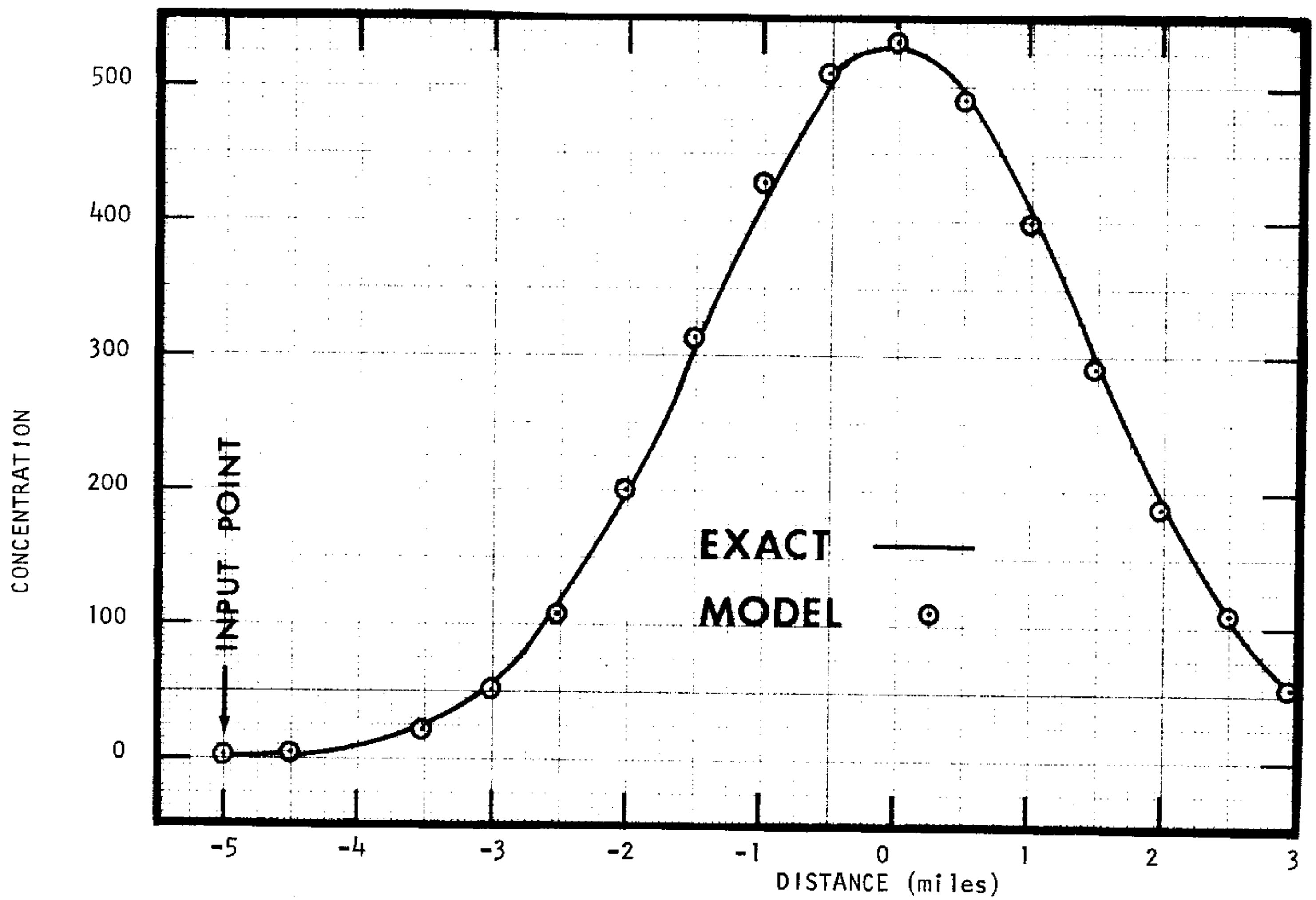


FIGURE 9-2. - PROFILE FROM A SLUG LOAD

developed in this study should be useful in predicting the concentration profiles resulting from slug loads; therefore, a more extensive analysis of stability and accuracy characteristics of the finite difference methods was believed to be worthwhile.

The explicit and implicit models were applied to a slug loading using several different time increments. The profiles were calculated until 0.6 days had elapsed and then the results were compared. The following parameters were the same for all runs: distance increment = 0.1 miles; dispersion coefficient = 1.0 mi^2/day (322.7 ft^2/sec); velocity = 5.0 miles/day (.306 ft/sec); and decay rate = 0.0. Two initial starting conditions were applied. One starting condition placed a single concentration value of 18939.39 ppm at the source point at time = 0.0 days. The second starting condition applied a known concentration profile calculated by the analytical solution for an elapsed time of 0.1 days; this known initial distribution included 28 points and is shown in Table 9-8.

The accuracy of the computed profiles were compared for a series of different time increments. The maximum time increment for stability of the explicit method was calculated to be 0.005 days.

Table 9-9 shows the profiles for the Crank-Nicolson implicit method with a time increment of 0.002 days and the profile for the explicit method with a time increment of

TABLE 9-8.-INITIAL CONCENTRATION VALUES

Mile	Concentration (ppm)
-3.0	904.3
-2.9	1132.5
-2.8	1349.1
-2.7	1528.7
-2.6	1647.8
-2.5	1689.5
-2.4	1647.8
-2.3	1528.7
-2.2	1349.1
-2.1	1132.5
-2.0	904.3
-1.9	686.9
-1.8	496.3
-1.7	341.1
-1.6	223.0
-1.5	138.7
-1.4	82.04
-1.3	46.17
-1.2	24.71
-1.1	12.58
-1.0	6.09
-0.9	2.81
-0.8	1.23
-0.7	0.51
-0.6	0.20
-0.5	0.08
-0.4	0.03
-0.3	0.01

TABLE 9-9.-PROFILES CALCULATED BY FINITE DIFFERENCE METHODS FOR A SLUG LOAD; ELAPSED TIME = 0.6 DAYS

Distance from peak (miles)	Exact	Implicit ΔT = .002 Days	% Error	Explicit ΔT = .0005 Days	% Error
-2.5	51.0	50.1	-1.7	49.7	-2.5
-2.4	62.6	61.7		61.2	
-2.3	76.1	75.2		74.7	
-2.2	91.8	91.0		90.4	
-2.1	109.8	109.2		108.5	
-2.0	130.3	129.8	-.37	129.0	-.97
-1.9	153.3	153.0		152.2	
-1.8	178.8	178.9	~0.0	177.9	
-1.7	206.9	207.3		206.3	~0.0
-1.6	237.4	238.2		237.1	
-1.5	270.1	271.3		270.2	
-1.4	304.8	306.4		305.4	
-1.3	341.1	343.2		342.1	
-1.2	378.5	381.0		380.1	
-1.1	416.6	419.5		418.7	
-1.0	454.7	457.9	+.70	457.2	+.55
-0.9	492.2	495.6		495.1	
-0.8	528.3	531.9		531.6	
-0.7	562.4	566.0		566.0	
-0.6	593.7	597.2		597.5	
-0.5	621.5	624.8		625.4	
-0.4	645.3	648.2		649.1	
-0.3	664.4	666.8		668.0	
-0.2	678.3	680.2		681.7	
-0.1	686.9	688.1		689.9	
+0.0	689.7	690.2	+.07	692.3	+.37
+0.1	686.9	686.6	~0.0	688.8	
+0.2	678.3	677.4		679.7	
+0.3	664.4	662.7		665.1	
+0.4	645.3	642.9		645.4	~0.0
+0.5	621.5	618.6		621.0	
+0.6	593.7	590.4		592.7	
+0.7	562.4	558.7		560.9	
+0.8	528.3	524.5		526.5	
+0.9	492.2	488.4		490.3	
+1.0	454.7	451.2	-.77	453.0	-.39

0.0005 days. The percentage of error is shown at certain points. Similar analyses of profiles for other time increments are summarized in Tables 9-10, 9-11, 9-12, and 9-13. In these four tables, execution time is included to demonstrate the accuracy which can be obtained for a certain expenditure of computer time.

The percentage of error for various time increments in Table 9-10 are shown graphically in Figure 9-3. Error in terms of concentration is shown for the same data in Figure 9-4; the concentrations should be compared with the peak concentration of 689.7 ppm for 0.6 days lapsed time.

Discussion of Accuracy for Slug Loads. - The analyses in this section demonstrate that the Crank-Nicolson implicit method generally provides better accuracy and more freedom from oscillations than the explicit method. In addition, for one-dimensional applications the implicit method requires less computer time than the explicit method for an accurate solution. However, excellent accuracy also can be obtained by the explicit method when the time increment is chosen properly. The choice of a distance increment is also important in limiting the degree of accuracy which can be obtained by either method.

The accuracy of the explicit method improves noticeably for smaller and smaller time increments. This accuracy will have a lower limit because round-off errors become more significant as

TABLE 9-10.-ACCURACY ANALYSIS: EXPLICIT METHOD.
PROFILE AFTER 0.6 DAYS FOR AN INITIAL SLUG LOAD OF 18939.39

Time increment in days ΔT	Execution time for 0.6 days (seconds)	% Error at peak	No error distance from peak (miles)		% Error at certain distances from peak: +1.0 miles	-1.0 miles	-2.0 miles	-2.5 miles	Oscillations and stability
0.0005	33.72	+0.37	+0.4	-1.6	-0.39	+0.55	-.97	-2.5	Stable
0.001	17.90	+0.67	+1.0	-1.3	0.00	+0.40	-1.6	-3.2	Stable
0.00167	10.62	+1.06	-	-1.1	+0.52	+0.19	-2.4	-4.2	Stable
0.002	9.69	+1.25	-	-1.1	+0.78	+0.07	-2.8	-4.7	Stable
0.003	6.01	+1.85	-	-0.9	+1.60	-0.26	-4.0	-6.2	Stable
0.004	4.55	+2.45	-	-0.8	+2.40	-0.62	-5.2	-7.6	Early oscillations quickly damped out
0.0045	4.20	+2.80	-	-0.7	+1.0	-1.9	-7.3	-9.5	Early oscillations damped out
0.005	3.84	+205.70	-	-	-	-	-	-	Unstable oscillations

TABLE 9-11.-ACCURACY ANALYSIS: EXPLICIT METHOD.
PROFILE AFTER 0.6 DAYS FOR AN INITIAL KNOWN DISTRIBUTION AT 28 POINTS

Time increment in days ΔT	Execution time for 0.5 days (seconds)	% Error at peak	No error distance from peak (miles)		% Error at certain distances from peak +1.0 miles	-1.0 miles	-2.0 miles	-2.5 miles	Oscillations and stability
0.0005	29.37	+0.42	+0.5	-2.3	-0.41	+0.82	+0.28	-1.8	Stable
0.002	7.63	+1.16	+1.2	-1.3	+0.32	+0.45	-1.20	-2.0	Stable
0.004	3.59	+2.16	+1.3	-1.0	+1.30	-0.08	-3.10	-4.3	Stable
0.005	3.37	+2.67	+1.3	-0.9	+1.80	-0.36	-4.10	-5.5	Stable
0.006	3.15	-	-	-	-	-	-	-	Unstable oscillations after .2 days

TABLE 9-12.-ACCURACY ANALYSIS: CRANK-NICOLSON IMPLICIT METHOD. PROFILE AFTER 0.6 DAYS FOR AN INITIAL SLUG LOAD OF 18939.39

Time increment in days ΔT	Execution time for 0.6 days (seconds)	% Error at peak	No error distance from peak (miles)		% Error at certain distance from peak				Oscillations and stability
					+1.0 miles	-1.0 miles	-2.0 miles	-2.5 miles	
.002	16.83	+ .07	+0.1	-1.8	-.77	+0.70	-0.37	-1.7	Stable
.005	6.81	+ .13	+0.1	-1.8	-.74	+0.75	-0.35	-1.7	Stable
.01	3.80	+ .17	+0.1	-1.8	-.79	+0.77	-0.46	-1.9	Stable
.015	2.67	+ 2.00	-	-2.4	+.77	+2.70	+1.30	-1.7	Stable
.02	1.92	+24.00	-	-	-	-	-	-	Early oscillations severly dis-tort profile

TABLE 9-13.-ACCURACY ANALYSIS: CRANK-NICOLSON IMPLICIT METHOD.
PROFILE AFTER 0.6 DAYS FOR AN INITIAL KNOWN DISTRIBUTION AT 28 POINTS

Time increment in days ΔT	Execution time for 0.5 days (seconds)	% Error at peak	No error distance from peak (miles)		% Error at certain distance from peak				Oscillations and stability
					+1.0 miles	-1.0 miles	-2.0 miles	-2.5 miles	
.001	30.93	+ 0.22	+.2	>-3.0	- 0.61	+0.98	+0.81	+ 0.45	Stable
.002	14.54	+ 0.17	+.2	>-3.0	- 0.66	+0.94	+0.76	+ 0.39	Stable
.01	5.66	+ 0.26	+.2	-3.0	- 0.70	+0.99	+0.70	+ 0.29	Stable
.02	3.40	+ 0.45	+.3	-2.4	- 0.94	+1.10	+0.43	- 0.12	Stable
.05	1.94	+ 1.80	+.4	-1.6	- 2.70	+1.80	-1.40	- 2.90	Stable
.10	1.18	+ 6.90	+.4	-1.3	-10.40	+4.00	-9.20	-11.80	Stable
.20	0.61	+19.50	-	-	-	-	-	-	Severe oscillations and distorted profile

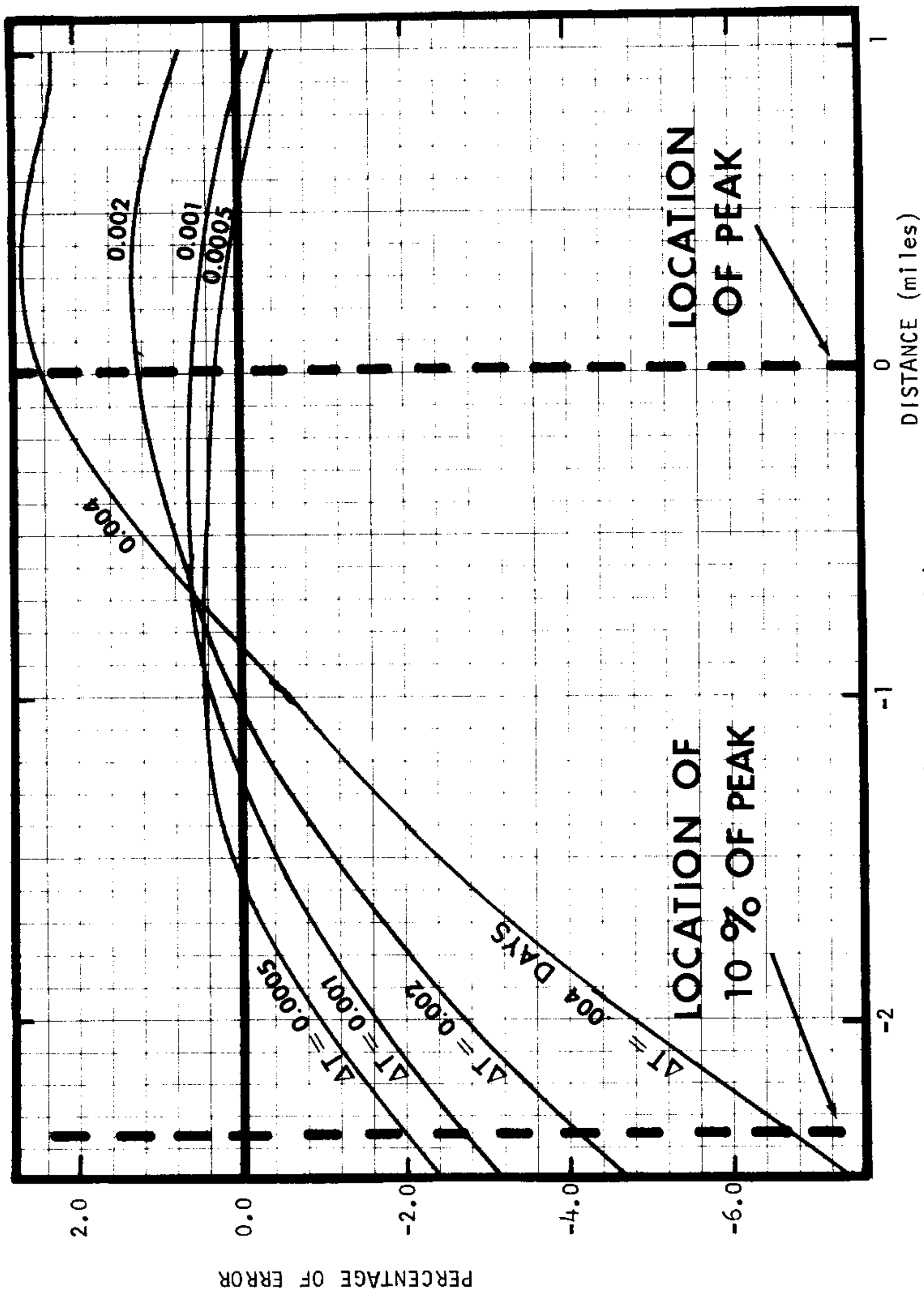


FIGURE 9-3. - PERCENTAGE OF ERROR (see TABLE 9-10)

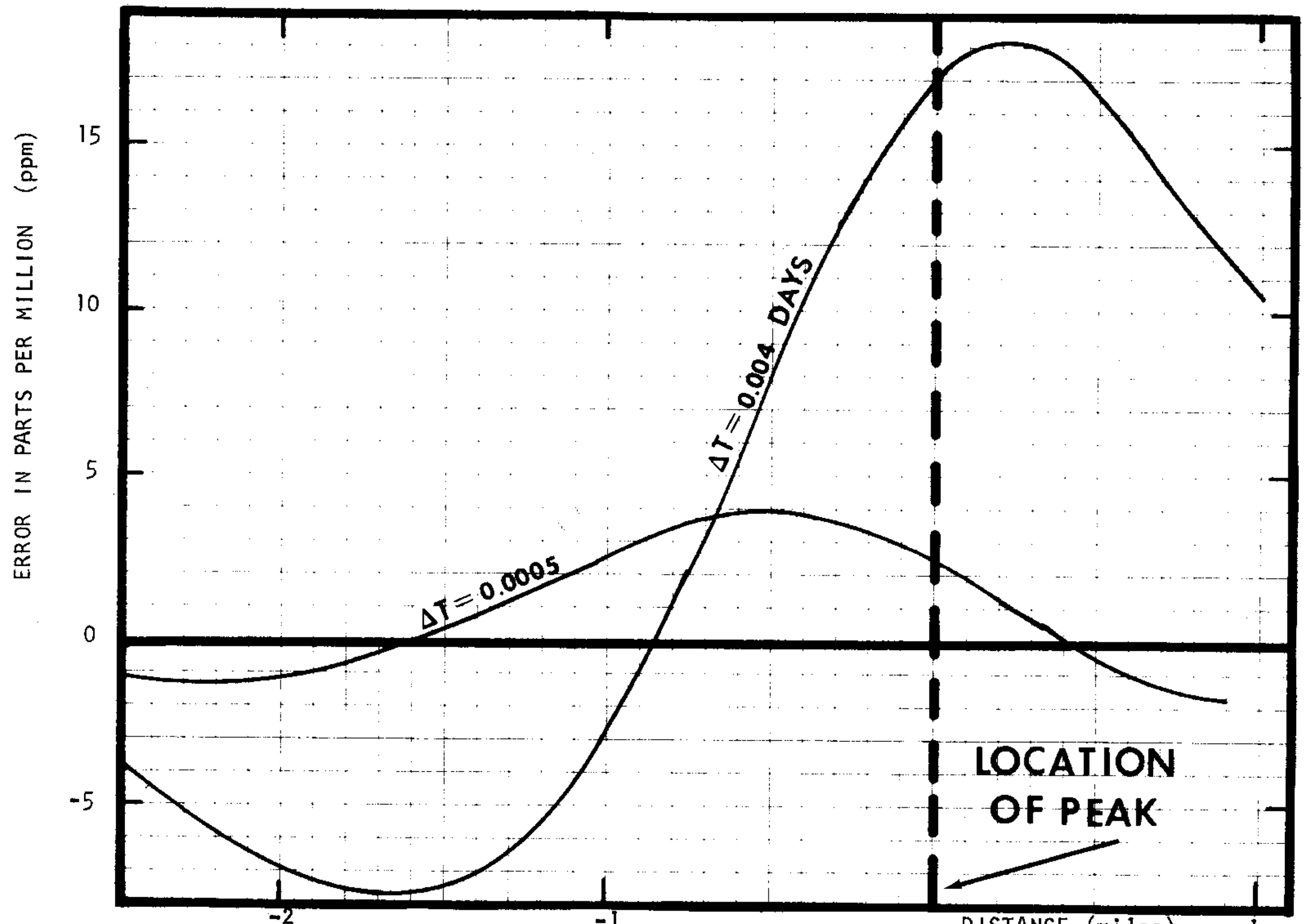


FIGURE 9-4. - ERROR FOR A PEAK 689.7 PPM (see TABLE 9-10)

the time increment becomes extremely small. Likewise, the distance increment limits the final accuracy which may be obtained, even when small time increments are used.

The implicit method has a more consistent accuracy throughout the useful range of time increments; this allows the use of larger time increments with the assurance that good accuracy can be obtained.

Starting the calculations with an initial distribution rather than a slug concentration at a single point has certain advantages. Although the accuracy near the peak is not significantly affected by the initial conditions, the accuracy of the leading and trailing edges is noticeably improved by using a distribution of values rather than a single concentration; this difference would become less significant as the dispersion coefficient is increased. Also, it should be noted that the leading and trailing edges of a profile for a slug load are generally not of much importance because they represent such a small percentage of the total mass of material.

Another advantage of using an initial distribution is that the implicit method remains accurate for much larger time increments. Table 9-12 shows that the implicit method is very accurate for a single starting concentration up to the time increment of 0.01 days, which is twice the value for stability for the explicit method. For an initial known distribution,

Table 9-13 shows that the implicit method is accurate up to a time increment of 0.05 days, which is ten times the stability criteria of the explicit method.

The distance increment of 0.1 miles provided good accuracy for any reasonable time increment applied in the error analysis. For example, with the explicit method a maximum error of 18.2 ppm is shown in Figure 9-4 near the peak concentration of 689.7 ppm and an error of about 4 ppm is shown for the trailing edge of the concentration. Thus, if a similar distribution were encountered from real data with a maximum BOD concentration of 6.89 mg/l, the maximum expected error from the explicit model would be 0.18 mg/l near the peak and about 0.04 mg/l near the edges of the profile; this is an acceptable error for real data. It should be noted that this magnitude of error resulted from a time increment of 0.004 days, which is larger than the time increment that would probably be used for these circumstances.

The following conclusions are clear from the calculations presented in this section: when a careful choice is made for the distance increment, the explicit method can provide good accuracy until the stability criteria is exceeded; the implicit method can provide good accuracy up to three times the stability criteria of the explicit method. When an initial distribution is known, the implicit method may remain accurate for up to ten times the stability criteria for time of the explicit method. Some authors recommend that the time increment for the explicit method be restricted to one-half the stability criteria and that the Crank-Nicolson method be limited to twice this time increment in order to avoid inaccuracies caused by oscillations; these recommendations appear to be overly conservative for the examples presented in this section.

TWO-DIMENSIONAL ACCURACY ANALYSIS

The explicit and implicit finite difference models for two dimensions exhibited the same type of accuracy and stability as the one-dimensional models.

Steady-State Solutions. - Excellent accuracy can be obtained for steady-state solutions in two dimensions. However, iterating until the steady-state is reached can consume a large amount of computer time unless an initial distribution is used which is fairly close to the final values.

Instantaneous Releases of Slug Loads. - Excellent accuracy can be obtained in two dimensions when only dispersion is modeled. The introduction of velocity causes the same type of skewing and 'numerical dispersion' problems as in the one-dimensional cases; these inaccuracies can be minimized greatly by the proper choice of time and distance increments.

Table 9-14 shows the degree of accuracy which can be expected when both velocity and dispersion are included in the calculations. To obtain this distribution, an initial symmetrical

TABLE 9-14.-CONCENTRATIONS AT TIME = 0.11 DAYS

Miles	-.5	-.3	-.1	+.1	+.3	+.5	+.7
-.5	77	174	329	518	681	745	681
-.3	174	394	745	1174	1543	1689	1543
-.1	329	745	1408	2219	2915	3192	2915
+.1	518	1174	2219	3496	4592	5029	4592
+.3	681	1543	2915	4592	6032	6606	6032
+.5	745	1689	3192	5029	6606	7234	6606
+.7	681	1543	2915	4592	6032	6606	6032

(a) Exact Solution

Miles	-.5	-.3	-.1	+.1	+.3	+.5	+.7
-.5	75	171	327	519	680	739	669
-.3	171	392	750	1188	1558	1693	1532
-.1	327	750	1434	2273	2980	3238	2931
+.1	519	1188	2273	3603	4723	5131	4645
+.3	680	1558	2980	4723	6193	6727	6090
+.5	739	1693	3238	5131	6727	7308	6616
+.7	669	1532	2931	4645	6090	6616	5989

(b) Implicit Solution for ΔT = 0.0025 Days

Miles	-.5	-.3	-.1	+.1	+.3	+.5	+.7
-.5	73	167	320	507	669	734	670
-.3	167	259	541	941	1529	1678	1532
-.1	320	730	1393	2212	2920	3203	2925
+.1	507	1159	2212	3513	4635	5084	4641
+.3	669	1529	2920	4635	6113	6700	6112
+.5	734	1678	3203	5084	6700	7334	6684
+.7	670	1532	2925	4641	6112	6684	6085

(c) Explicit Solution for ΔT = 0.0010 Days distribution was read into the models for a starting time of 0.01 days. The models then calculated the distribution until the elapsed time was 0.11 days. The explicit method used a time increment of 0.0025 days. The stability limits for the explicit method were 0.0025 days and 0.4 miles. The models used a distance increment of 0.1 miles; however, Table 9-14 presents the results for only every 0.2 miles. The exact solution was obtained by applying Equation 8-5 with m = 10,000; dispersion coefficient = 1 mi^2/day (322.7 ft^2/sec); and velocity = 5 miles/day (.306 ft/sec). Almost equivalent accuracy was obtained by the models when a time increment of 0.002 days was used for the explicit method and a time increment of 0.005 days was used for the implicit method.

Good accuracy was also obtained when the dispersion coefficients and velocities varied in the two dimensions. The explicit model was applied to the following parameters: EX = 1 mi^2/day (322.7 ft^2/sec); EZ = 0.025 ft^2/sec ($.775 \times 10^{-4}$ mi^2/day); VX = 5 miles/day (.306 ft/sec); VZ = 0.0 ft/sec. The initial profile at 0.01 days is shown in Table 9-15 for selected values, and the distribution calculated for 0.11 days is shown in Table 9-16. The vertical distance increment is 5 feet and the horizontal distance increment is 0.1 miles. The model used a time increment 0.002 days and the stability criteria for time was 0.0027 days. The exact solution is based on a value m = 10 in Equation 8-5.

TABLE 9-15.-INITIAL VALUES FOR NON-SYMMETRICAL DISTRIBUTION

Depth (Feet)	-0.5 Miles	-0.4 Miles	-0.3 Miles	-0.2 Miles	-0.1 Miles	L.P.*	0.1 Miles	0.2 Miles	0.3 Miles	0.4 Miles	0.5 Miles
0 ft.	17.	166.	953.	3325.	7040.	9039.	7040.	3325.	953.	166.	17.
5 ft.	13.					6769.					13.
10 ft.	5.					2842.					5.
15 ft.	1.					669.					1.
20 ft.	0.	2.	9.	32.	69.	88.	69.	32.	9.	2.	0.
25 ft.	0.					7.					0.
30 ft.	0.					0.					0.
35 ft.	0.	0.	0	0.	0.	0.	0.	0.	0.	0.	0.

*Loading Point

TABLE 9-16.-TWO DIMENSIONAL DISTRIBUTION AT TIME = 0.11 DAYS

Depth (Feet)	L.P.*	0.1 Miles	0.2 Miles	0.3 Miles	0.4 Miles	0.5 Miles	0.6 Miles	0.7 Miles	0.8 Miles	0.9 Miles	1.0 Miles
0 ft.	458. / 466.	565. / 571.	666. / 670.	749. / 750.	805. / 803.	827. / 822.	810. / 803.	759. / 750.	679. / 670.	580. / 571.	474. / 466.
5 ft.	446. / 453.					805. / 800.					461. / 453.
10 ft.	413. / 419.					745. / 740.					426. / 419.
15 ft.	363. / 367.					654. / 649.					372. / 367.
20 ft.	303. / 306.	374. / 375.	440. / 440.	495. / 493.	531. / 527.	545. / 540.	533. / 527.	498. / 493.	444. / 440.	379. / 375.	309. / 306.
25 ft.	240. / 241.					431. / 426.					242. / 241.
30 ft.	181. / 181.					323. / 319.					180. / 181.
35 ft.	129. / 128	159. / 157.	187. / 185.	210. / 207.	225. / 221.	230. / 227.	225. / 221.	209. / 207.	186. / 185.	157. / 157.	127. / 128.

* Loading Point

Figure Key: $\frac{458.}{466.} = \frac{\text{Exact}}{\text{Model}}$

For the examples shown in this chapter, the accuracy was poorest during early time steps and continued to improve as time progressed.

CHAPTER X

COMPUTATIONAL EXPERIENCE

Three aspects of the use of the computer models will be discussed briefly in this chapter: time requirements; use of boundary conditions; and choice of input parameters.

TIME REQUIREMENTS

The computer programs developed in this study are very economical in their use of computer time when compared to some other models for large estuaries (26). However, the programs are time-consuming with regard to the computer budget of a student; this is especially true for the two-dimensional programs when a large number of alternatives must be investigated to insure accuracy.

Table 10-1 presents data on the time requirements for some typical runs on an IBM 360/65 of the most recent versions of the models. The information includes the size of the grid, the number of iterations, the time to compile and initialize the data, the execution time for the computation phase, the number of seconds of execution time per iteration, and the number of seconds of execution time per iteration per grid point. This final parameter allows for the computation of an approximate run time of the models on an IBM 360/65. For instance, the approximate time requirement for MASSTRANS-I would be:

TABLE 10-1.-COMPUTATION TIMES

Grid	Itera-tions	Initial-ization (sec)	Execution Time (Seconds)	Seconds / Iteration	Seconds / Inter.-Pt.
			(a) MASSTRANS-I (Explicit)		
15 x 15	102	22.	111.5	1.09	484×10^{-5}
15 x 15	52	22.	55.2	1.06	472×10^{-5}
36 x 11	42	22.	81.6	1.94	490×10^{-5}
42 x 1	122	20.	21.0	0.172	409×10^{-5}
36 x 1	182	20.	21.6	0.119	331×10^{-5}
			(b) MASSTRANS-II (Implicit)		
15 x 15	42	25.	115.2	2.75	1225×10^{-5}
15 x 15	22	25.	58.2	2.65	1180×10^{-5}
			(c) IDEAL-I (Explicit)		
51 x 1	1200	1.	33.72	.0281	55×10^{-5}
51 x 1	600	1.	17.90	.0298	58×10^{-5}
51 x 1	300	1.	9.69	.0323	63×10^{-5}
			(d) IDEAL-II (Implicit)		
51 x 1	600	1.	30.93	.0516	101×10^{-5}
51 x 1	300	1.	16.83	.0562	110×10^{-5}
51 x 1	120	1.	6.81	.0567	111×10^{-5}
			(e) OXTRANS-I (Explicit)		
42 x 1	204	21.	82.80	.406	965×10^{-5}
42 x 1	102	21.	34.20	.335	798×10^{-5}
31 x 1	402	21.	106.00	.265	853×10^{-5}
33 x 7	68	23.	147.00	2.160	935×10^{-5}

Total Computer Time in Seconds

$$= 25 + (500 \times 10^{-5} \times \text{iterations} \times \text{total number of grid points})$$

Since different options and boundary conditions were used for the various programs, the time requirements are not completely comparable. However, it is clear that the one-dimensional version of the Crank-Nicolson implicit method tends to take 1.75 times as long as the explicit method for the same time and distance increments. Likewise, the two-dimensional version of the implicit method takes 2.5 times as long as the explicit method. However, since the implicit method remains accurate for much larger time increments than the explicit method, the implicit method often proves to be the more economical choice.

OXTRANS-I, the largest program, has the following storage requirements:

Object Code = 66,000 bytes; Array Area = 59,000 bytes.

BOUNDARY CONDITIONS

For steady-state profiles and for leading and trailing edges of a concentration profile, inaccurate boundary conditions significantly affect only the closest three internal concentrations. However, as the concentrations near the boundary become a high percentage of the peak value, the entire profile can be distorted by inaccurate boundary extrapolations. The most straightforward way of avoiding this "feedback" is to keep the peak far away from the boundaries where values must be extrapolated; this can be accomplished by using a large grid or by moving the grid when the peak gets too close to the boundaries.

CHOICE OF INPUT PARAMETERS

Two-dimensional, time-varying data is difficult and expensive to obtain. However, this type of data is required to completely calibrate and verify two-dimensional models.

In spite of the logistical problems involved, velocity is one of the more easily involved parameters because it can be measured directly with sensitive current meters. If horizontal velocities are carefully measured, vertical velocities can be calculated through the two-dimensional continuity equation. For estuaries whose width and velocities vary throughout the estuary, care must be taken to insure that input values for velocity satisfy continuity. If measured data is not available, approximate intra-tidal velocity values also can be obtained by coupling the finite difference models to computer models for tidal velocities; a one-dimensional tidal model is available for the Houston Ship Channel (110).

Dispersion coefficients are particularly troublesome to obtain in two dimensions. As shown by Ippen (51), the value of the horizontal dispersion coefficient for one-dimensional models is much larger than the horizontal dispersion coefficients appropriate for two-dimensional models. Horizontal dispersion coefficients for one-dimensional models must represent the effects of the more localized eddy diffusion; in two-dimensional analysis, the two extra variables of vertical velocity and vertical dispersion are available to aid in representing these effects. More research needs to be done to determine representative values for vertical dispersion for individual estuaries. Work by Prichard (80) and Bowden (5) suggests that vertical dispersion coefficients can be expected to be in the range of 0.001 to 0.050 ft^2/sec, where the higher values indicate a higher degree of vertical mixing.

Adequate source and sink terms are sometimes difficult to obtain. Decay rates in highly polluted estuaries must be chosen with care because of possible inhibition effects (87, 85). Reaeration rates are particularly elusive for slowly moving, highly contaminated estuaries.

As Callaway has said (20), this is truly "the age of the great coefficient hunt."

CHAPTER XI

APPLICATION TO THE HOUSTON SHIP CHANNEL

An important aspect in proving the accuracy and usefulness of a model comes through testing the ability of the model to reproduce actual data. On May 24-26, 1971, a dye study was carried out on the Houston Ship Channel by the Estuarine Systems Projects of Texas A&M University during a moderate rainfall on the watershed surrounding the channel. This study provided sufficient data (31) to determine the ability of the model to reproduce the two-dimensional behavior of the channel during dynamic conditions.

Two hundred pounds of Rhodamine WT dye were released at Mile 16.5 on May 24, 1971, at 12 noon (hereafter called H-hour). Two-dimensional measurements for dye were taken periodically on the channel during the 44 hours following the release. The gauged runoff into the channel peaked at H + 7 hours (7:00 p.m.) on May 24 at a discharge of approximately 4750 cfs (31). The channel was partially stratified during the study period; for example, on the morning of May 25, the salinity at Mile 12 ranged from 12.6 ppt at the surface to 15.6 ppt at the bottom. The tide readings for the study period are shown in Figure 11-1.

The first two-dimensional data collection for dye fluorescence was carried out at H + 4 hours. The fluorescence contours for that sampling run are shown in Figure 11-2; the channel

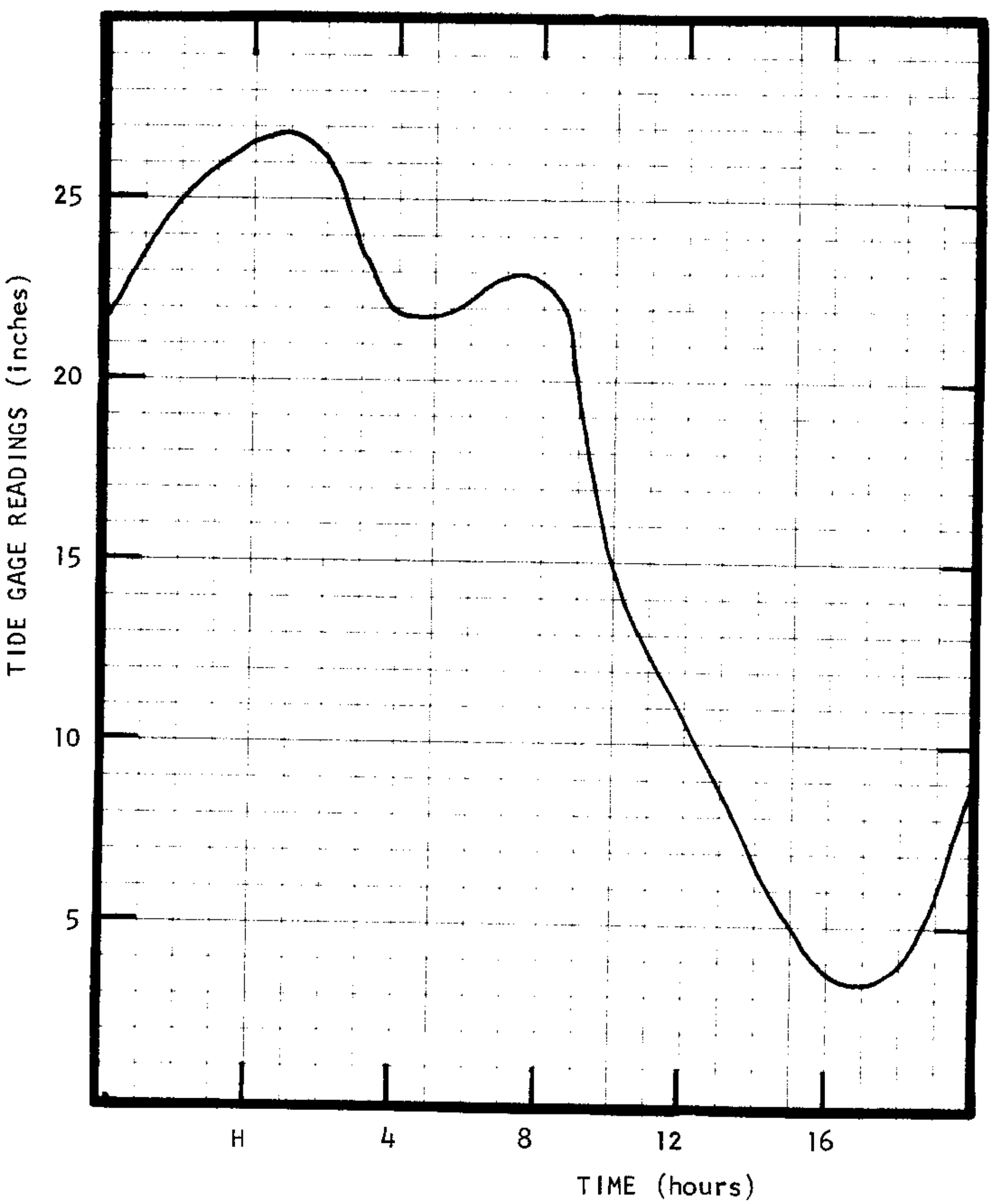


FIGURE 11-1. - TIDE READINGS

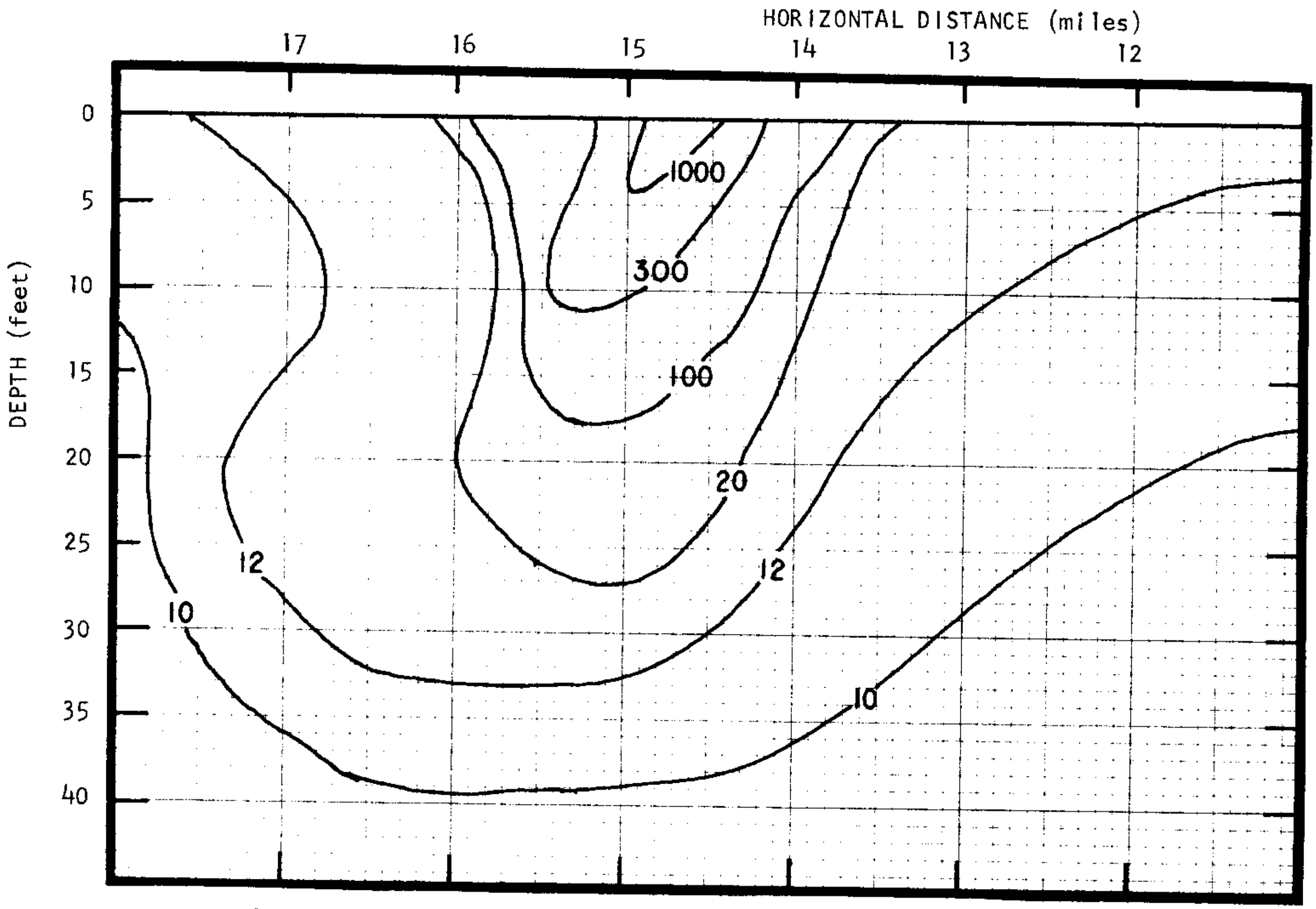


FIGURE 11-2. - MEASURED CONCENTRATIONS AT H + 4

had a background fluorescence of about 10 units.

The concentrations from this first sampling run were used as the input concentrations for the model. Concentrations were chosen at intervals of 0.25 miles in the longitudinal direction and 5.0 feet in the vertical. These initial concentration values are shown in Table 11-1.

Data was available also from a two-dimensional sampling run at H + 15 hours. The aim of the simulation was to compare the concentrations predicted by the model with the concentrations measured at H + 15 hours.

Average values were used in the model for velocity, dispersion, and decay during the 11-hour simulation period. Horizontal velocities were obtained by channel measurements and observation of fluorescence contours. Vertical velocity was considered to be insignificant and was set equal to zero. Horizontal dispersion was set at 450 ft^2/sec (1.39 $miles^2/day$) at all points. Vertical dispersion was considered to be proportional to horizontal velocity and had a maximum value of 0.004 ft^2/sec (1.24×10^{-5} $miles^2/day$). Values for horizontal velocity and vertical dispersion coefficients are shown in Table 11-2.

Widths were approximated from Corps of Engineer cross-section data and from work by Hutton (45, 46). These values are shown in Table 11-3 and represent widths at a 20-foot depth. The sampling data indicated that a considerable

TABLE 11-1.-INITIAL CONCENTRATIONS

Mile	0'	5'	10'	15'	20'	25'	30'	35'	40'
17.00	13.2	12.0	11.7	12.0	13.0	13.0	11.8	10.2	10.0
16.75	14.0	13.5	12.0	13.0	14.0	14.0	12.1	10.3	10.0
16.50	16.2	15.0	14.0	14.0	15.0	15.0	12.3	10.4	10.0
16.25	19.5	17.0	16.0	16.0	17.0	16.0	12.5	10.5	10.0
16.00	100.0	19.0	19.0	19.0	20.0	17.0	12.7	10.6	10.0
15.75	125.0	80.0	50.0	40.0	35.0	19.0	13.0	10.8	10.0
15.50	175.0	220.0	300.0	130.0	65.0	22.0	13.5	11.0	10.0
15.25	300.0	500.0	450.0	175.0	80.0	30.0	14.0	10.8	10.0
15.00	750.0	900.0	330.0	150.0	65.0	24.0	13.3	10.6	10.0
14.75	2000.0	550.0	225.0	100.0	40.0	20.0	12.6	10.4	10.0
14.50	1000.0	300.0	130.0	60.0	25.0	17.0	12.0	10.3	10.0
14.25	300.0	200.0	85.0	30.0	18.0	14.0	11.6	10.2	10.0
14.00	175.0	85.0	35.0	19.0	15.0	12.0	11.2	10.1	10.0
13.75	100.0	20.0	19.0	15.0	12.0	11.5	10.8	10.0	10.0
13.50	30.0	18.0	17.0	12.0	11.7	11.0	10.4	10.0	10.0
13.25	19.0	17.0	15.0	11.8	11.5	10.7	10.0	10.0	10.0
13.00	17.0	16.0	14.0	11.6	11.2	10.4	10.0	10.0	10.0
12.75	15.0	15.0	13.0	11.4	11.0	10.2	10.0	10.0	10.0
12.50	14.0	14.0	12.0	11.3	10.7	10.0	10.0	10.0	10.0
12.25	13.0	13.0	11.8	11.2	10.5	10.0	10.0	10.0	10.0
*12.00	12.5	12.0	11.6	11.1	10.2	10.0	10.0	10.0	10.0

*All concentrations not shown were equal to 10.0. The background fluorescence of 10.0 was subtracted from the concentrations shown in the table and the resulting values were used as the input data.

TABLE 11-2.-HORIZONTAL VELOCITIES AND VERTICAL DISPERSION COEFFICIENTS

Depth (ft.)	Horizontal Velocity (ft/sec)	Vertical Dispersion (ft^2/sec)
0	0.3667	.3333
5'	0.4200	.3820
10'	0.4400	.4000
15'	0.4400	.4000
20'	0.4333	.3940
25'	0.3893	.3540
30'	0.2733	.2485
35'	0.2333	.2120
40'	0.2333	.2120

TABLE 11-3.-WIDTHS AT 20-FOOT DEPTH

Mile	Width (feet)	Mile	Width (feet)
17.00	497		
16.75	510	12.75	872
16.50	524	12.50	931
16.25	537	12.25	863
16.00	550	12.00	796
15.75	562	11.75	728
15.50	575	11.50	661
15.25	562	11.25	657
15.00	550	11.00	654
14.75	537	10.75	650
14.50	525	10.50	647
14.25	568	10.25	633
14.00	611	10.00	619
13.75	654	9.75	605
13.50	697	9.50	591
13.25	756	9.25	586
13.00	814	9.00	582

amount of dye was lost by sediment adsorption and other influences. An exponential decay rate of 2.5 / day was found to adequately represent this loss of dye for the period being simulated.

The simulation run used a time increment of 600 seconds; the maximum allowable time increment for stability would have been 1155 seconds. The simulation required 66 iterations; the compile time plus the execution time was 2.5 minutes on an IBM 360/65 for MASSTRANS-I.

The measured dye concentration profiles for hour H + 15 are shown in Figure 11-3. The concentration profiles predicted by MASSTRANS-I are shown in Figure 11-4. Agreement between the profiles is very good considering that only average values were used as input data in the model.

Comparison between the dye concentrations and model predictions are shown at 5-foot increments in Figure 11-5. Peak values are accurately matched and the mass of dye appears to be conserved by the model. The skewing of the measured profiles was caused by local velocity fluctuations which were not included in the model input data.

Figure 11-6 shows a comparison between peak concentrations in the vertical direction for the model and the sampled data. This comparison demonstrates that the model can be especially accurate in representing the peak concentrations at the appropriate depths.

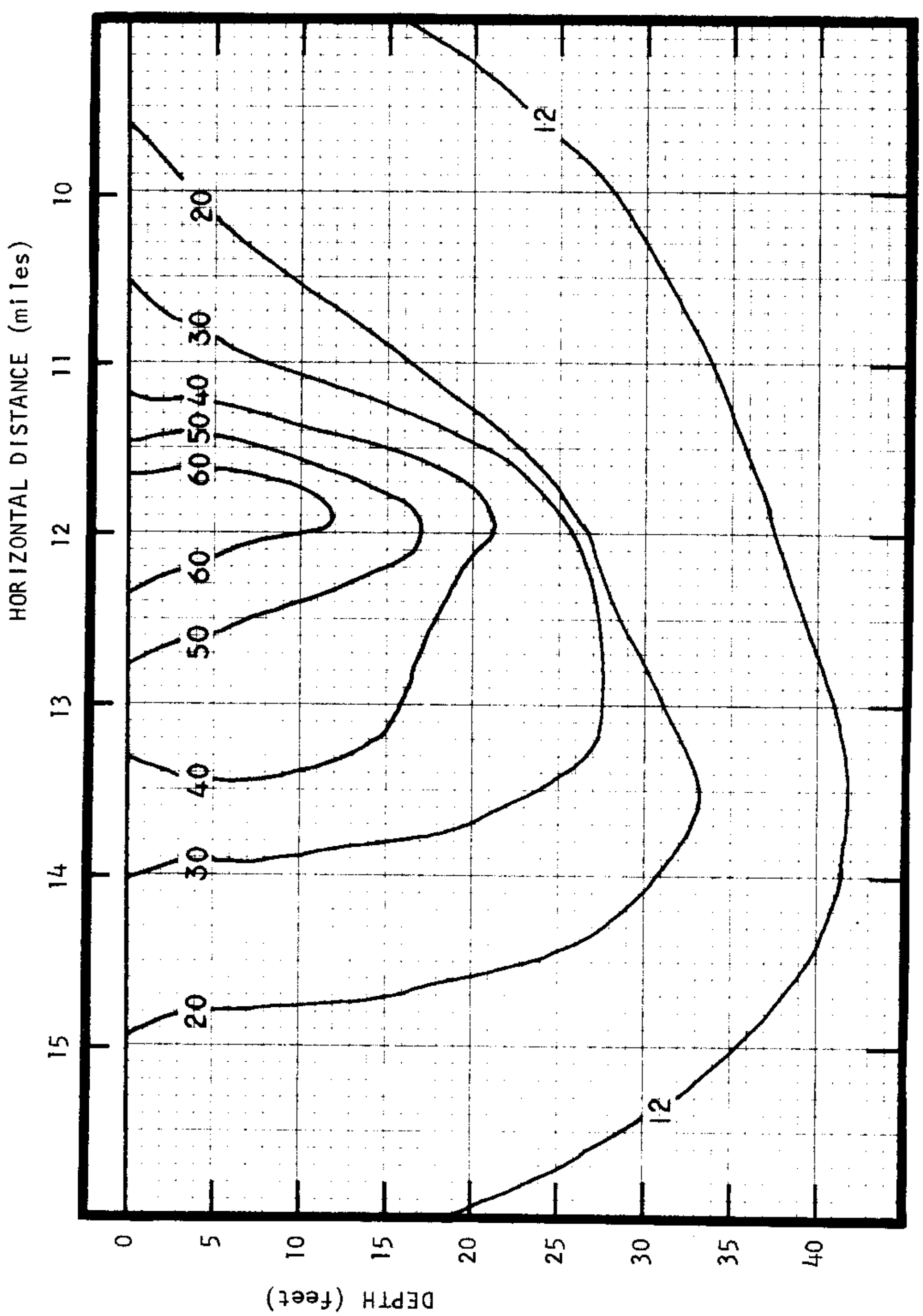


FIGURE 11-3. - MEASURED CONCENTRATIONS AT HOUR H + 15

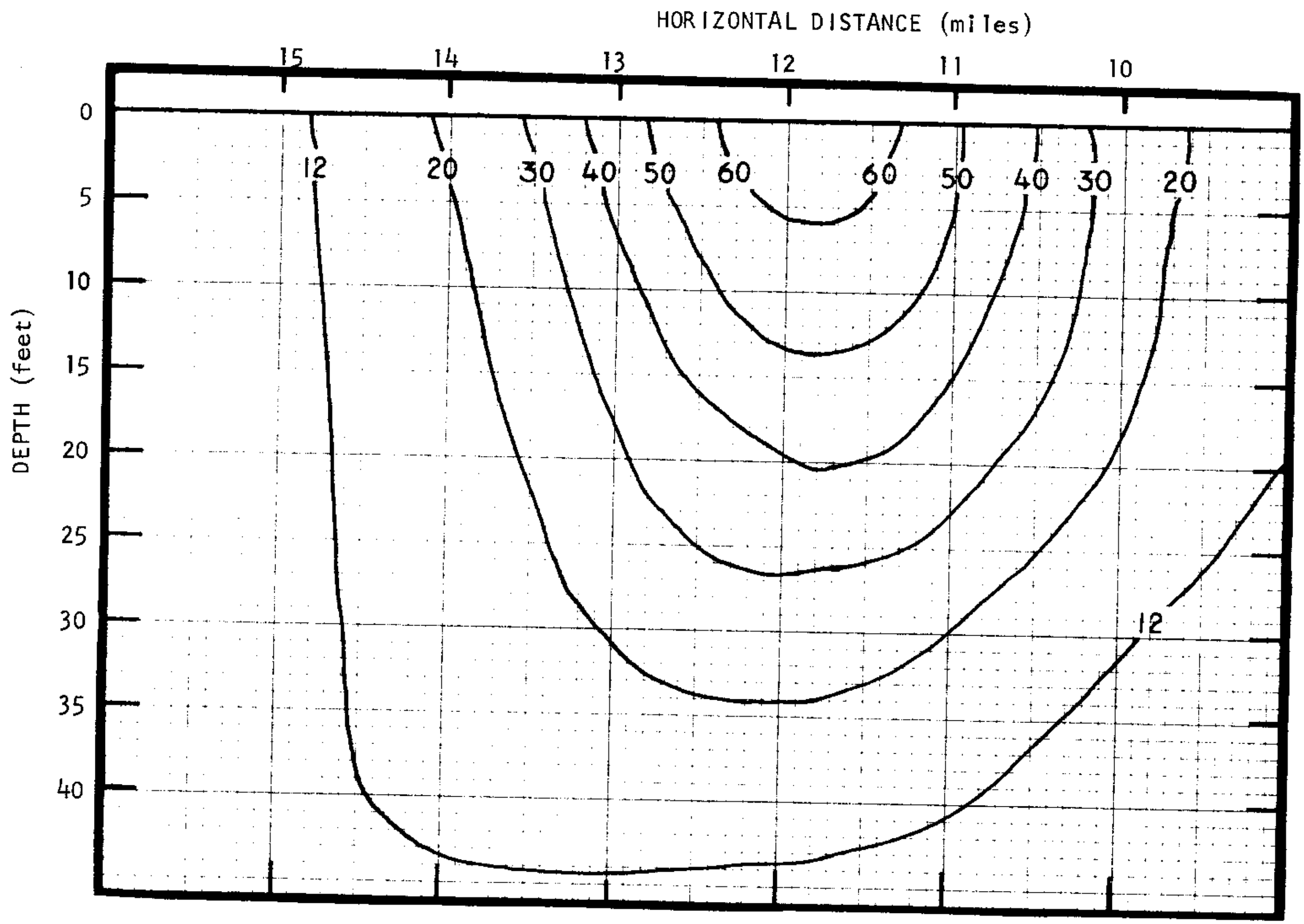


FIGURE 11-4. - CONCENTRATIONS PREDICTED BY MASSTRANS-1 FOR HOUR H + 15

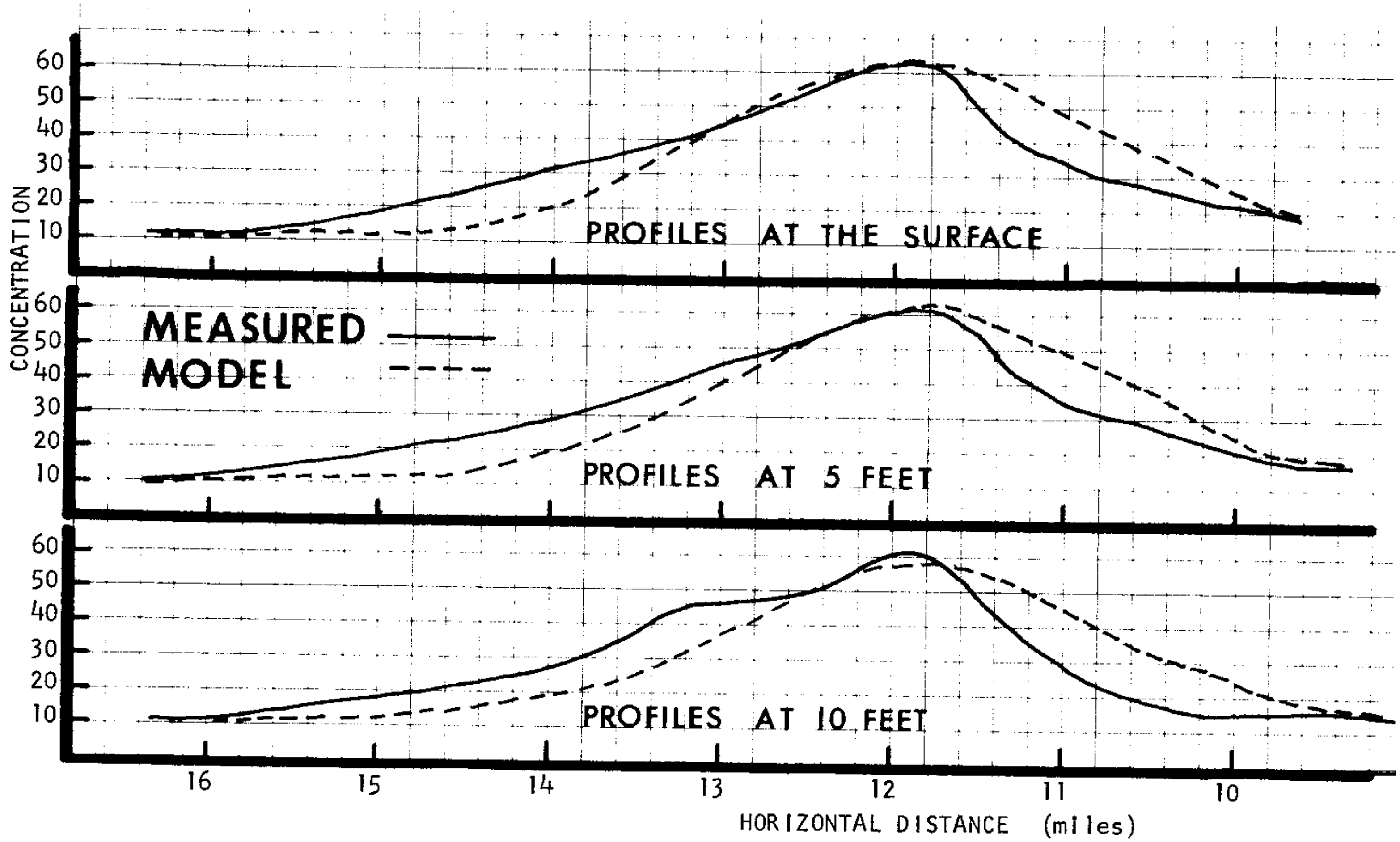
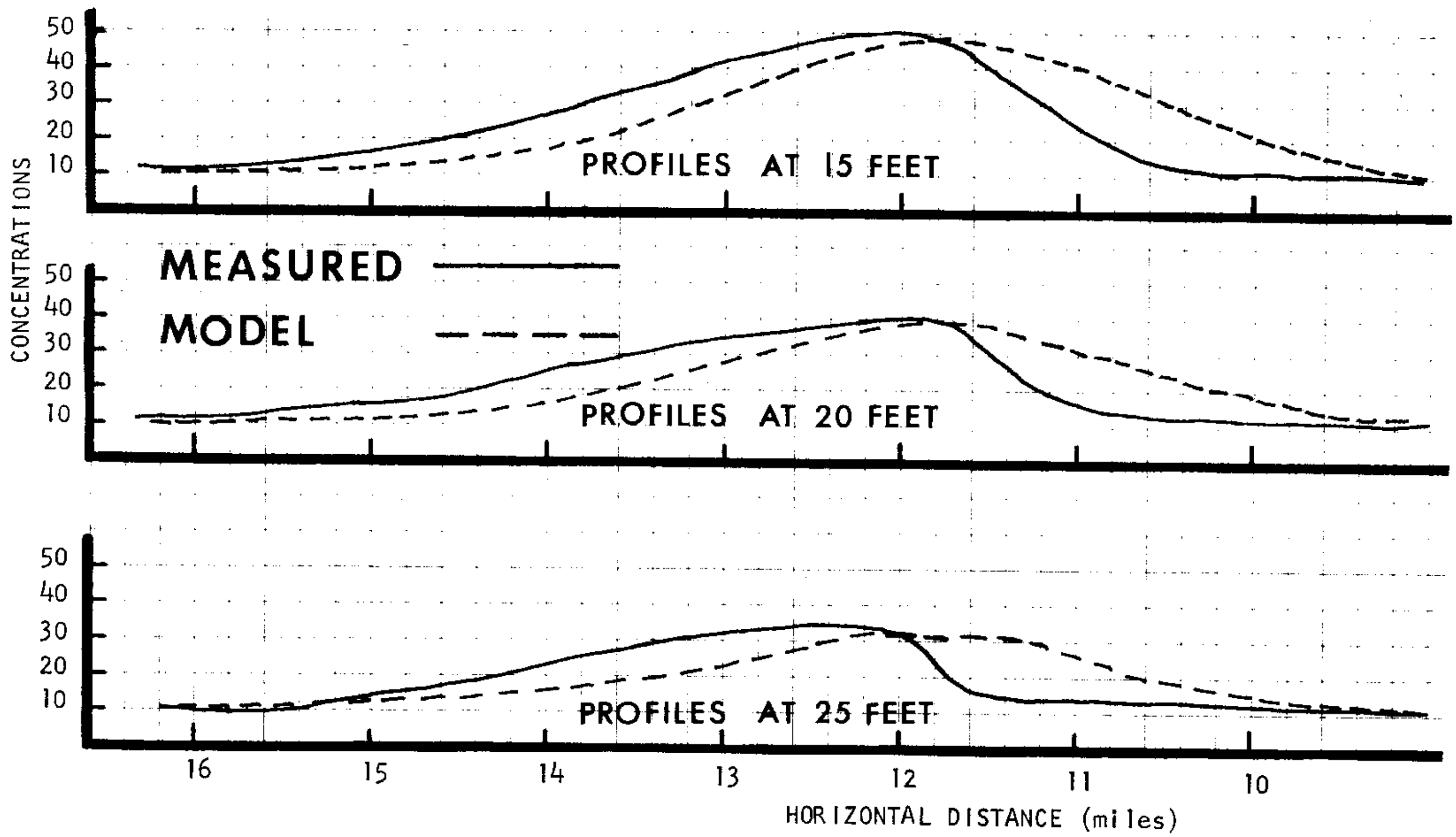


FIGURE 11-5. - COMPARISON OF MEASURED AND SIMULATED PROFILES

(Continued on next page)



(End of Figure 11-5)

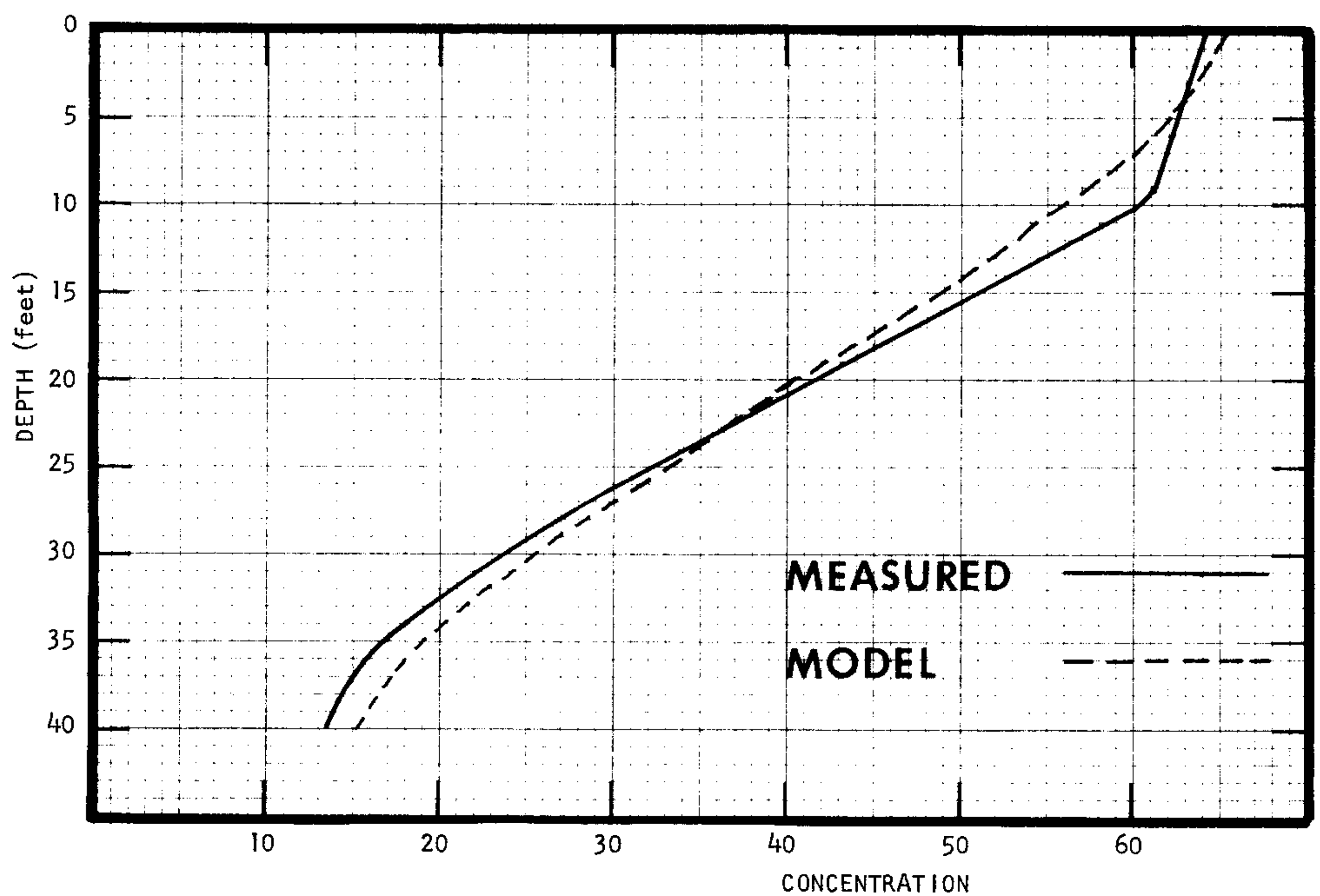


FIGURE 11-6. - COMPARISON OF PEAK CONCENTRATIONS

These comparisons demonstrate that the two-dimensional models can accurately represent dynamic conditions on the Houston Ship Channel, even when averaged input data is used. When more precise data for velocity, dispersion, and other parameters are fed into the model, more accurate simulations can be expected.

MODELING BOD AND DISSOLVED OXYGEN IN THE HOUSTON SHIP CHANNEL

A major use of OXTRANS-I will be to model the BOD and dissolved oxygen distributions in the Channel for rapidly changing conditions. The ability of the model to accomplish this task was demonstrated by applying the velocity and dispersion values which existed during the dye study to a hypothetical dissolved oxygen distribution. A simple initial distribution of dissolved oxygen and BOD was chosen in order that changes in the distribution would be more apparent. The hypothetical initial values for BOD_{20} and dissolved oxygen are shown in Figure 11-7; this figure shows a saturated value of 8 ppm of oxygen and no BOD below Mile 15. Above Mile 15, a BOD_{20} concentration of 13 ppm is shown along with a zero value for dissolved oxygen.

The velocity and dispersion values from the dye study simulation were applied to the initial DO-BOD values, and the simulation was run again until hour H + 15. The following parameters were used: aerobic decay rate (base e) = 0.25/day; anaerobic decay rate (base e) = 0.08/day; and reaeration rate (base e) =

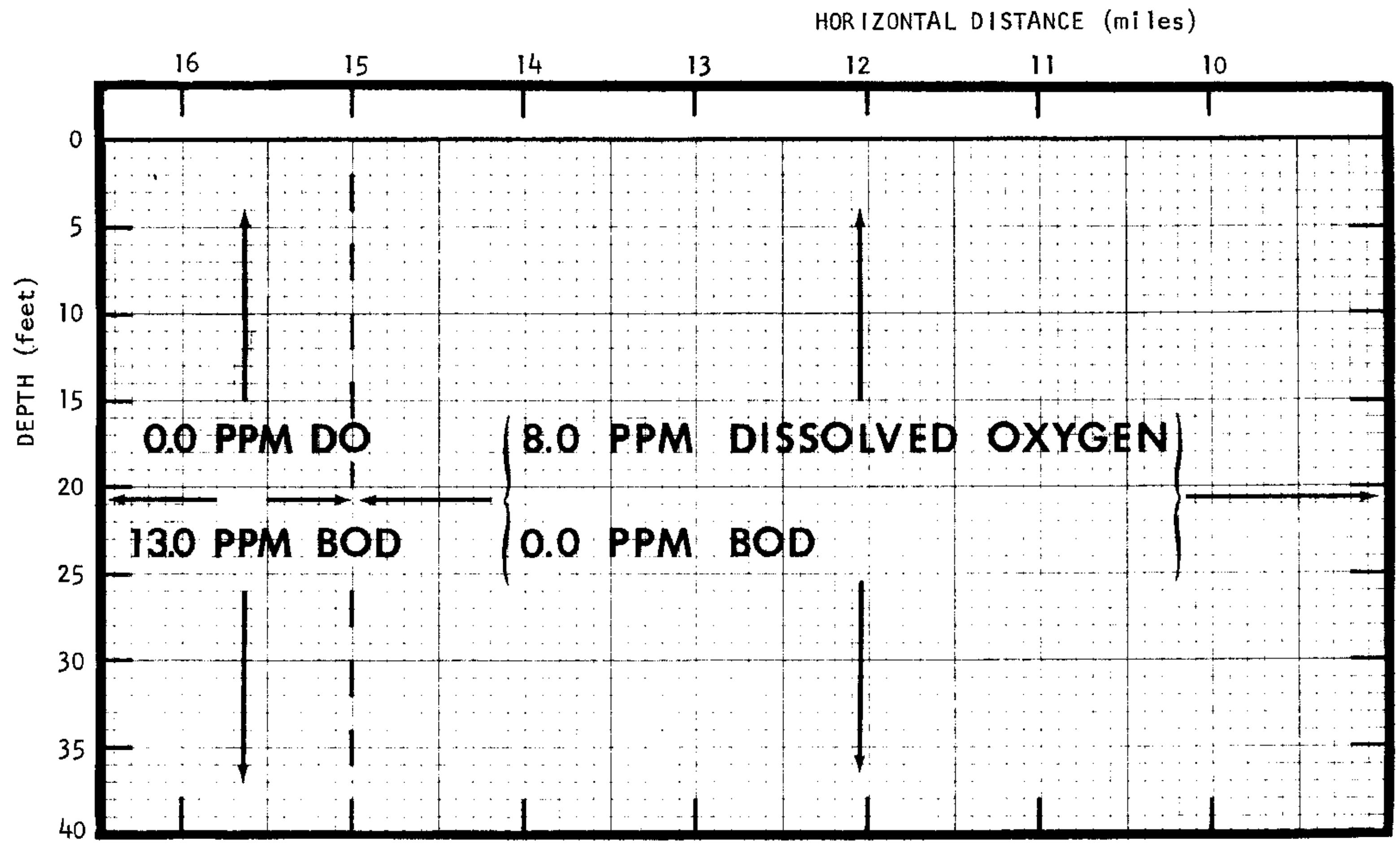


FIGURE 11-7. - INITIAL HYPOTHETICAL VALUES FOR BOD_{20} AND DO AT H + 4 HOURS

0.10/day. The results of this simulation are shown in Figure 11-8, for BOD_{20} and Figure 11-9, for dissolved oxygen. During the 11 hours of simulation, the saturated value of 8 ppm is pushed from Mile 15 to Mile 9.

This example demonstrates that the most pronounced change during the early stages of increased flow are due mainly to the downstream movement of the low oxygen conditions by velocity and dispersion. Several different initial BOD concentrations were tried in the model and the dissolved oxygen pattern at the end of the simulation was essentially the same as the values already shown in Figure 11-9. However, over a longer period of time, the persistence of the low oxygen conditions would obviously depend upon the amount of organic material that remained in the part of the channel being studied.

The conditions modeled in this simulation represent only moderate runoff and moderate pollution conditions. Extensive studies by Reynolds and Eckenfelder (85) during 1968 and 1969 show that the BOD_{20} in the vicinity of Mile 16 can rise to at least 38 ppm and the decay rate (base e) can range between .07/day and .44/day. Likewise, much higher runoff conditions are common to the channel.

APPLICABILITY OF THE MODELS

The examples of this chapter and the previous chapter demonstrate the reliability of the models when applied to the Houston

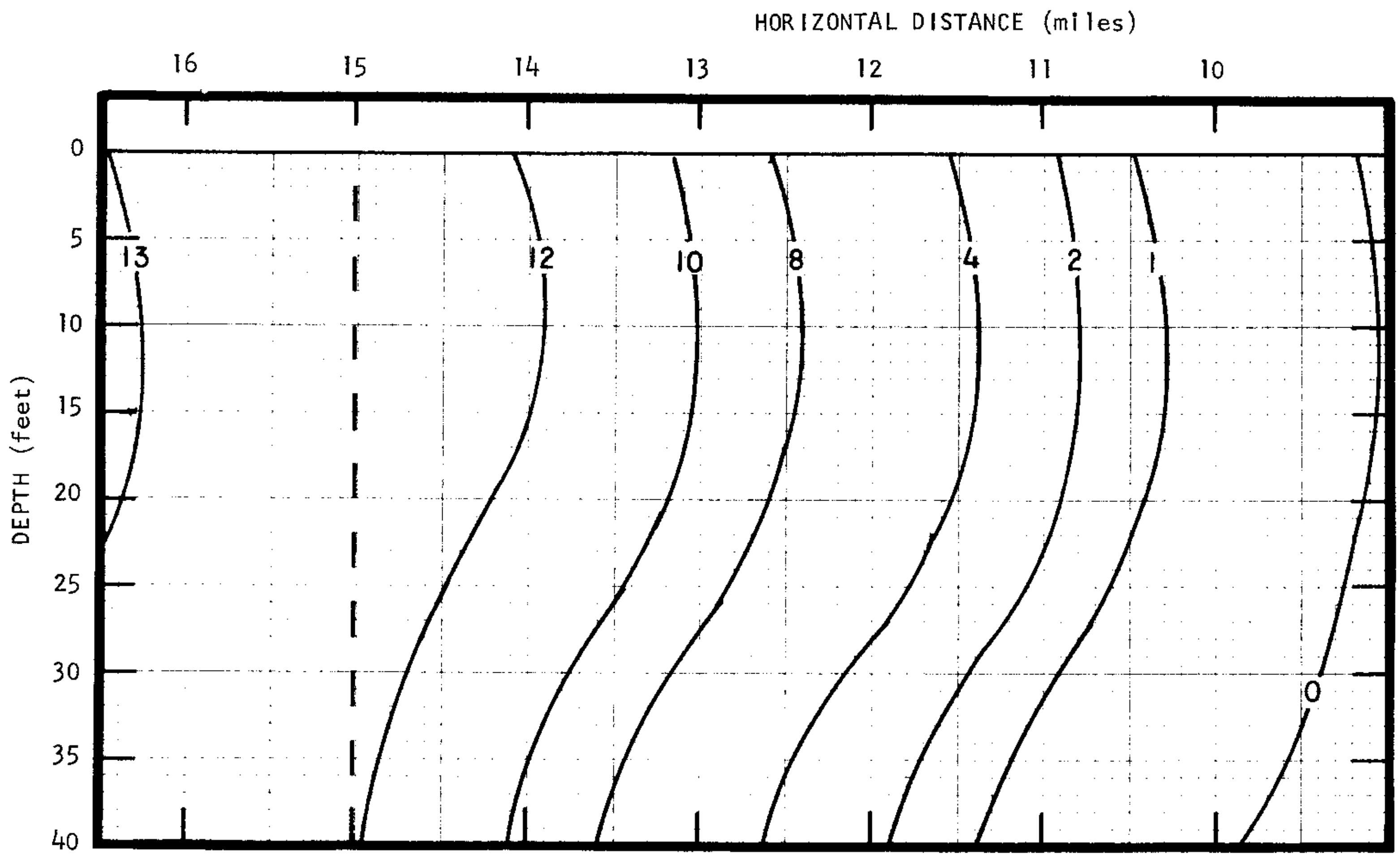


FIGURE 11-8. - BOD_{20} (ppm) AT H + 15 HOURS

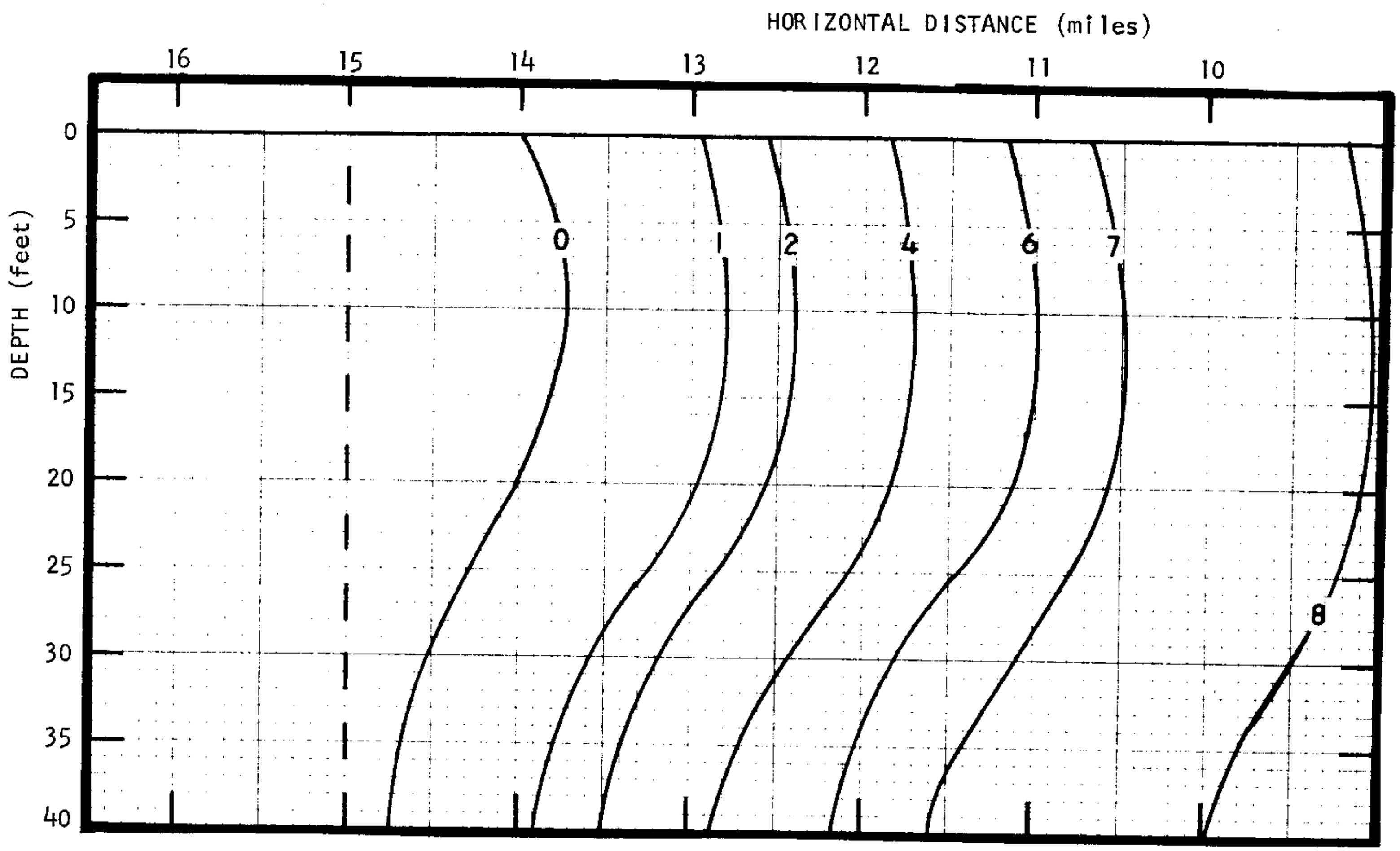


FIGURE 11-9. - DISSOLVED OXYGEN (ppm) AT H + 15 HOURS

Ship Channel and similar estuaries. These models can use approximate input parameters to provide estimates of concentration distributions in one or two dimensions; these approximations can be used for rough engineering estimates of conditions in the estuary for various hydrodynamic and loading conditions. The models can also accept input parameters which are much more exact; this type of approach is useful for simulation or predictive purposes.

CHAPTER XII

DISCUSSION

When modeling a partially stratified estuary, the following steps should be taken in utilizing the computer programs that were developed in this study. Representative parameter values for the estuary being studied should be used in the one-dimensional explicit and implicit models and the computed concentrations should be checked against exact solutions. Several time and distance steps should be tried until good accuracy is obtained for the range of conditions being studied. These initial computations should indicate the approximate increments to use in the two-dimensional models. In turn, the two-dimensional models should be run several times with different increments. The best approach is to use both the explicit and implicit models to cross-check each other. Special care should be taken when studying instantaneous releases; inaccurate initial conditions can invalidate the entire computation.

One of the real dangers in mathematical modeling is that almost any type of estuary model can generate profiles that appear to be reasonable. Some estuary models have gained acceptance through repeated use and promotion, not through repeated proof of their accuracy. For this reason, this study has taken great care to establish the accuracy of its finite difference models and has provided means of checking their accuracy when applied to parameters other than those used in this report. The models developed in this study include no weighting factors and depend solely on the estuary data provided as input; all of this data can be obtained independently of the models.

In this study, good accuracy was obtained for horizontal distance increments up to 0.25 miles. For finite difference methods, increasing the distance increment generally decreases the accuracy of the model and increases interference by numerical dispersion; this is especially true when studying time-changing behavior. Thus, a model whose grid dimensions can be changed easily has a great advantage: the model can be run several times with different distance increments until results of sufficient accuracy are obtained.

The computer programs that were developed in this study have numerous applications. The two-dimensional models can be applied directly to partially stratified estuaries of constant center-line depth; dredged channels in the Gulf Coast region are the most obvious examples of this type of estuary. The one-dimensional models can be applied directly to well-mixed estuaries.

The accuracy requirements of the input data for the models is determined by the needs of the user. Rough estimates of concentration profiles can be obtained by inputting a representative range of values for the estuary being studied. If more accurate results are required, data for the model should be obtained from dye releases, velocity measurements, reaeration rate measurements, decay rate studies, and benthal deposit studies. Examples of these types of investigations are available in several sources (2, 8, 40, 44, 46, 56, 71, 85, 103).

The usefulness of the present form of the computer programs is limited in several respects. The two-dimensional models require computers with large core storage; in addition, the computer programs require a compiler which accepts FORTRAN IV and unformatted READ and PRINT statements. These shortcomings can be overcome by straightforward programming techniques: discs and tapes can be used as alternate storage locations and format statement can be rewritten according to requirements of the computer being used.

The finite difference methods discussed in this study can be extended to estuaries with varying depths. Likewise, the finite difference equations can be expanded to three dimensions. However, in both these cases, the computational grids must be matched to the geometry of the particular estuary; this would require extensive additional programming and would result in models which were estuary-dependent. Additional work is needed on models of this type.

Additional verification of the present models is also needed. The ultimate usefulness of these methods can be determined only after extensive computations are made for a wide variety of inter-tidal and intra-tidal conditions.

CHAPTER XIII

SUMMARY AND CONCLUSIONS

The following goals were accomplished in this study: (a) computer models were developed which can calculate time-varying vertical and horizontal mass transport in partially stratified estuaries; (b) the accuracy and usefulness of explicit finite difference models were compared with that of Crank-Nicolson implicit finite difference models; (c) the applicability of finite difference models to the mass transport characteristics of the Houston Ship Channel was demonstrated; and (d) a summary was made of existing information on one- and two-dimensional mathematical models that have been applied to significant estuary problems.

Explicit and Crank-Nicolson finite difference models were developed for the one- and two-dimensional estuary equations with varying coefficients. The models were constructed to allow for the varying of parameters at any time at any grid point and were programmed in FORTRAN-IV computer language. Good accuracy was obtained by both types of models when proper time and distance increments were used. The Crank-Nicolson approach was found to be more accurate for a wider range of these increments. The concentration profiles for instantaneous releases and for steady-state conditions were analyzed. Accuracy was determined by comparison with analytical closed-form solutions.

Models also were developed to analyze the profiles for biochemical oxygen demand and dissolved oxygen under time-changing conditions. These models can analyze both aerobic and anaerobic conditions and included provisions for analyzing the effects of mechanical reaeration.

Applicability of these models to partially stratified estuaries was established by comparisons with dye study data from the Houston Ship Channel.

The Crank-Nicolson implicit method programmed in this study tends to take 1.75 times as much computer time as the explicit method in one-dimensional applications and 2.5 times as much time for two-dimensional applications when equal time and distance increments are used; however, the implicit method is sometimes the more economical choice since it remains stable and accurate for larger time increments than the explicit method.

The accuracy of the finite difference models is particularly sensitive to the distance increments. Horizontal grid increments of 0.25 miles or less can provide excellent accuracy when finite difference models are applied to instantaneous releases. Grid increments of 2.0 miles or less can provide excellent accuracy when finite difference models are applied to the downstream portions of steady-state profiles. Finite difference models should have grid sizes which can be varied easily to enable the choice of a distance increment which provides good accuracy for the conditions being studied.

The finite difference methods outlined in this study can be extended to partially stratified estuaries with varying depth and to three-dimensional estuary calculations; these applications would require extensive modifications to the existing programs. Further verification of the present two-dimensional models also is needed to determine their general applicability.

This study and similar studies demonstrate that techniques are presently available to model a wide range of hydrodynamic and mass transport conditions in estuaries. A major future task is to determine for two and three dimensions the appropriate values for parameters such as velocity, dispersion, decay, re-aeration, photosynthesis, and benthal demands.

APPENDIX I. - REFERENCES

1. Abbott, M. R., "Salinity Effects in Estuaries," Journal of Marine Resources, Vol. 18, No. 2, 1960, pp. 101-111.

2. Bailey, Thomas E., "Estuarine Oxygen Resources - Photosynthesis and Reaeration," Journal Sanitary Engineering Division, ASCE, Vol. 96, No. SA2, Proc. Paper 7215, April 1970, pp. 279-296.

3. Bella, David A., and Dobbins, William E., "Difference Modeling of Stream Pollution," Journal Sanitary Engineering Division, ASCE, Vol. 94, SA5, October 1968, pp. 995-1016.

4. Bird, R. Byron, Stewart, Warren E., and Lightfoot, Edwin N., Transport Phenomena, John Wiley & Sons, Inc., New York, 1960.

5. Bowden, K. F., "Circulation and Mixing in the Mersey Estuary," Publication No. 51, International Association of Scientific Hydrology, Commission of Surface Waters, 1960.

6. Bowden, K. F., "The Mixing Processes in a Tidal Estuary," in Advances in Water Pollution Research, E. A. Pearson, editor, Vol. 3, Pergamon Press, 1964, pp. 329-347.

7. Bowden, K. F., "Circulation and Diffusion," in Estuaries, Lauff, George H., ed., Publication No. 83, American Association for the Advancement of Science, Washington, D. C., 1967, 15-36.

8. Bunce, Ronald E., "A Discussion and Tabulation of Diffusion Coefficients for Tidal Waters Computed as a Function of Velocity," CB-SRBP Technical Paper No. 9, Middle Atlantic Region, Federal Water Pollution Control Administration, Charlottesville, Virginia, February 1967.

9. Bunce, Ronald E., and Hetling, Leo J., "A Steady State Segmented Estuary Model," CB-SRBP Technical Paper No. 11, Federal Water Pollution Control Administration, Middle Atlantic Region, Charlottesville, Virginia, 1968.

10. Callaway, R. J., Byram, K. V., and Ditsworth, G. R., "Mathematical Model of the Columbia River from the Pacific Ocean to Bonneville Dam, Part I," Federal Water Quality Administration Pacific Northwest Water Laboratory, Corvallis, Oregon, 1969.

11. Carnahan, Brice, Luther, H. A., and Wilkes, James O., Applied Numerical Methods, John Wiley & Sons, New York, 1969.

12. Churchill, M. A., Elmore, H. L., and Buckingham, R. A., "The Prediction of Stream Reaeration Rates," Journal Sanitary Engineering Division, ASCE, Vol. 88, No. SA4, Proc. Paper 3199, July 1962, pp. 1-46.

13. Cress, Paul, Dirksen, Paul, and Graham, J. Wesley, FORTRAN IV, With Watfor and Watfiv, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1970.

14. Daily, James W., and Harleman, Donald R. F., Fluid Dynamics, Addison-Wesley Publishing Co., 1966.

15. Dake, Jonas M. K., and Harleman, Donald R. F., "Thermal Stratification in Lakes: Analytical and Laboratory Studies," Water Resources Research, AGU, Vol. 5, No. 2, April 1969, pp. 484-495.

16. Dobbins, William E., "Diffusion and Mixing," Journal Boston Society of Civil Engineers, November 1963, pp. 108-127.

17. Douglas, J., "A Survey of Numerical Methods for Parabolic Differential Equations," Advances in Computers, Vol. 2, Academic Press, New York, 1961.

18. Douglas, J., and Gunn, J. E., "A General Formulation of Alternating Direction Methods, Part I, Parabolic and Hyperbolic Problems," Numerische Mathematik, Vol. 6, 1964, pp. 428-453.

19. Dronkers, J. J., Tidal Computations in Rivers and Coastal Waters, North-Holland Publishing Co., Amsterdam, 1964.

20. Environmental Protection Agency, "Estuarine Modeling: An Assessment," edited by TRACOR, Inc., Document No. 16070-DZV-02-71, U. S. Government Printing Office, Washington, D. C., February 1971.

21. Espey, W. H., Huston, R. J., Bergman, W. D., Stover, J. E., and Ward, G. H., "Galveston Bay Study, Phase I, Technical Report," Document No. 68-566-U, TRACOR, Inc., Austin, Texas, submitted to the Galveston Bay Study, Texas Water Quality Board, 1968.

22. Espey, W. H., Hays, A. J., Bergman, W. D., Buchner, J. P., Huston, R. J., and Ward, G. H., "Galveston Bay Project, Water Quality Modeling and Data Management, Phase II, Technical Progress Report," Document No. T70-AU-7636-U, TRACOR, Inc., Austin, Texas, submitted to the Galveston Bay Project, Texas Water Quality Board, July 1971.

23. Evett, Jack B., "A One-Dimensional Steady-State, Nonidealized Technique for Estuarine Analysis and Its Application to the Houston Ship Channel," thesis presented to Texas A&M University, at College Station, Texas, in January 1968, in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Department of Civil Engineering.

24. Federal Water Pollution Control Administration, "Delaware Estuary Comprehensive Study--Preliminary Report and Findings," Philadelphia, Pennsylvania, 1966.

25. Federal Water Pollution Control Administration, "Effects of the San Joaquin Master Drain on Water Quality of the San Francisco Bay and Delta," Central Pacific Basins Project, Southwest Region, San Francisco, California, January 1967.

26. Feigner, Kenneth D., and Harris, Howard S., "Documentation Report, FWQA Dynamic Estuary Model," Federal Water Quality Administration, U. S. Department of the Interior, July 1970.

27. Garder, A. O., Jr., Peaceman, D. W., and Pozzi, A. L., Jr., "Numerical Calculation of Multidimensional Miscible Displacement by the Method of Characteristics," Society of Petroleum Engineers Journal, AIME, Vol. 4, March 1964, pp. 26-36.

28. Gebhard, Thomas, Jr., "Numerical Solutions of the Two-Dimensional Convective Dispersion Equation," thesis presented to The University of Texas, at Austin, Texas, in May 1968, in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Department of Civil Engineering.

29. Hann, Roy W., "First Annual Report," Estuarine Systems Projects Technical Report No. 1, Civil Engineering Department, Texas A&M University, July 1969.

30. Hann, Roy W., Evett, Jack B., and Carpenter, Stewart B., "Version I: Steady-State Estuarine Model in Problem Oriented Language Form," Estuarine Systems Projects Technical Report No. 6, Civil Engineering Department, Texas A&M University, College Station, Texas, August 1969.

31. Hann, Roy W., Jr., "Selected Houston Ship Channel Studies," Estuarine Systems Projects Technical Report No. 21, Civil Engineering Department, Texas A&M University, June 1971.

32. Hansen, Donald V., and Rattray, Maurice, Jr., "Gravitational Circulation in Straits and Estuaries," Journal of Marine Research, Vol. 23, No. 2, 1965, pp. 104-122.

33. Hansen, Donald V., and Rattray, Maurice, Jr., "New Dimensions in Estuary Classification," Limnology and Oceanography, Vol. 11, No. 3, July 1966, pp. 319-326.

34. Hansen, Donald V., "Salt Balance and Circulation in Partially Mixed Estuaries," in Estuaries, Lauff, George H., ed., Publication No. 82, American Association for the Advancement of Science, Washington, D. C., 1967, pp. 45-51.

35. Harleman, Donald R. F., and Abraham, G., "One-Dimensional Analysis of Salinity Intrusion in the Rotterdam Waterway," Publication No. 44, Delft Hydraulics Laboratory, Delft, Netherlands, October 1966.

36. Harleman, D. R. F., and Ippen, A. T., "Two-Dimensional Aspects of Salinity Intrusion in Estuaries: Analysis of Salinity and Velocity Distributions," Technical Bulletin No. 13, Committee on Tidal Hydraulics, U. S. Army Corps of Engineers, Vicksburg, Mississippi, June 1967.

37. Harleman, Donald R. F., and Ippen, Arthur T., "Salinity Intrusion Effects in Estuary Shoaling," Journal Hydraulics Division, ASCE, Vol. 95, No. HY1, Proc. Paper 6340, January 1969, pp. 9-27.

38. Hays, Almond, Jr., "A Least-Cost Analysis for the Houston Ship Channel," thesis presented to The University of Texas, at Austin, Texas, in January 1970, in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Civil Engineering Department.

39. Hays, Almond J., Jr., and Gloyna, Ernest F., "Optimal Water Quality Management for Houston Ship Channel," Journal Sanitary Engineering Division, ASCE, Vol. 98, No. SA1, Proc. Paper 8715, February, 1972, pp. 195-214.

40. Hetling, Leo J., and O'Connell, Richard L., "Estimating Diffusion Characteristics in Tidal Waters," Water and Sewage Works, Vol. 112, No. 10, October 1965, pp. 378-380.

41. Hetling, Leo J., and O'Connell, Richard L., "Estimating Diffusion Characteristics in Tidal Waters," CB-SRBP Technical Paper No. 4, Middle Atlantic Region, FWPCA, Charlottesville, Virginia, 1966.

42. Hetling, Leo J., and O'Connell, Richard L., "A Study of Tidal Dispersion in the Potomac River," Water Resources Research, Vol. 2, No. 4, 1966, pp. 825-841.

43. Hildebrand, F. B., Introduction to Numerical Analysis, McGraw-Hill, New York, 1956.

44. Hung, Y. T., "Biological Reaction Rate Constants in Natural Water," thesis presented to The University of Texas, at Austin, Texas, in May 1970, in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Civil Engineering Department.

45. Hutton, Welford S., "A Study of Benthal Deposits in the Houston Ship Channel," thesis presented to Texas A&M University, at College Station, Texas, in 1970, in partial fulfillment of the requirements for the degree of Master of Science, Department of Civil Engineering.

46. Hutton, Welford S., Hann, Roy W., Jr., and Smith, Richard H., "A Quantitative and Qualitative Survey of Benthal Deposits Contained in the Houston Ship Channel," Estuarine Systems Projects Report No. 8, Civil Engineering Department, Texas A&M University, 1971.

47. International Business Machines Corporation, "Scientific Subroutine Package," 1967.

48. International Business Machines Corporation, "System/360 Scientific Subroutine Package, Version III," Document GH20-0205-4, Program No. 360A-CM-03X, Fifth edition, White Plains, New York, August 1970.

49. Ippen, Arthur T., Harleman, Donald R. F., and Lin, J. D., "Turbulent Diffusion and Gravitational Convection in an Idealized Estuary," Technical Report No. 38, Hydrodynamics Laboratory, Massachusetts Institute of Technology, March 1960.

50. Ippen, Arthur T., and Harleman, Donald R. F., "One-Dimensional Analysis of Salinity Intrusion in Estuaries," Technical Bulletin No. 5, Committee on Tidal Hydraulics, U. S. Army Corps of Engineers, Vicksburg, Mississippi, June 1961.

51. Ippen, Arthur T., ed., Estuary and Coastline Hydrodynamics, McGraw-Hill Book Co., New York, 1966.

52. Isaacs, Willie P., and Gaudy, A. F., Jr., "Atmospheric Oxygenation in a Simulated Stream," Journal Sanitary Engineering Division, ASCE, Vol. 94, No. SA2, April 1968, pp. 319-344.

53. Isaacs, Willie P., McCorkle, Melvin R., and Lorang, William F., "An Experimental Study of the Effects of Stream Channel Cross-Sectional Geometry on the Reaeration Rate," Proceedings Twenty-Fifth Purdue Industrial Waste Conference, May 1970.

54. Jeglic, John M., "DECS II, Digital Computer Program for Simulation of the Delaware Estuary's Behavior," Digital Computer Technology and Programming Analysis Memo No. 1027, Re-entry Systems Department, General Electric Co., Philadelphia, Pennsylvania, August 1965.

55. Jeglic, John M., "DECS III, Mathematical Simulation of the Estuarine Behavior," Digital Computer Technology and Programming Analysis Memo No. 1032, Revision A, Re-entry Systems Department, General Electric Co., Philadelphia, Pennsylvania, July 1967.

56. Juliano, David W., "Reaeration Measurements in an Estuary," Journal Sanitary Engineering Division, ASCE, Vol. 95, No. SA6, Proc. Paper 6987, December 1969, pp. 1165-1178.

57. Kaplan, Wilfred, Advanced Calculus, Addison-Wesley Publishing Co., Inc., Reading, Massachusetts, 1962.

58. Krenkel, P. A., and Orlob, G. T., "Turbulent Diffusion and the Reaeration Coefficient," Journal Sanitary Engineering Division, ASCE, Vol. 88, No. SA2, Proc. Paper 3079, March 1962, pp. 53-83.

59. Leendertse, Jan J., "Aspects of a Computational Model For Long Period Water Wave Propagation," RM-5294-PR, The Rand Corporation, 1967.

60. Leendertse, Jan J., "A Water Quality Simulation Model for Well-Mixed Estuaries and Coastal Seas: Volume I, Principles of Computation, RM-6230-RC, The Rand Corporation, 1970.

61. Masch, Frank D., Shankar, Nilakantan J., Jeffrey, M., Brandes, R. J., and White, W. A., "A Numerical Model for the Simulation of Tidal Hydrodynamics in Shallow Irregular Estuaries," Technical Report HYD-12-6901, Hydraulics Engineering Laboratories, The University of Texas, Austin, February 1969.

62. Masch, Frank D., and Shankar, Nilakantan J., "Mathematical Simulation of Two-Dimensional Horizontal Convective-Dispersion in Well-Mixed Estuaries," Proceedings Thirteenth Congress of the International Association for Hydraulic Research, Vol. 3, Paper C-32, Japan, September 1969.

63. Neumann, Gerhard, and Pierson, Willard J., Jr., Principles of Physical Oceanography, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1966.

64. O'Connor, Donald J., and Dobbins, W. E., "The Mechanism of Reaeration in Natural Streams," Journal Sanitary Engineering Division, ASCE, Vol. 82, No. SA6, Proc. Paper 1115, December 1956, pp. 1-30.

65. O'Connor, Donald J., "Oxygen Balance of an Estuary," Journal Sanitary Engineering Division, ASCE, Vol. 86, No. SA3, Proceedings Paper 2472, May 1960, pp. 35-55.

66. O'Connor, Donald J., "Organic Pollution of New York Harbor - Theoretical Considerations," Journal Water Pollution Control Federation, Vol. 34, No. 9, September 1962, pp. 905-919.

67. O'Connor, Donald J., "Estuarine Distribution of Nonconservative Substances," Journal Sanitary Engineering Division, ASCE, Vol. 91, No. SA1, Proc. Paper 4225, February 1965, pp. 23-42.

68. O'Connor, Donald J., "An Analysis of the Dissolved Oxygen Distribution in the East River," Journal Water Pollution Control Federation, Vol. 38, No. 11, November 1966, pp. 1813-1830.

69. O'Connor, Donald J., St. John, J. P., and DiToro, Dominic, "Water Quality Analysis of the Delaware River Estuary," Journal Sanitary Engineering Division, ASCE, Vol. 94, No. SA6, Proc. Paper 6318, December 1968, pp. 1225-1252.

70. Odum, H. T., "Primary Production in Flowing Waters," Limnology and Oceanography, Vol. 1, No. 2, 1956, pp. 102-117.

71. Odum, H. T., "Analysis of Diurnal Oxygen Curves for the Assay of Reaeration Rates and Metabolism in Polluted Marine Bays," Proceedings of the First International Conference on Waste Disposal in the Marine Environment, 1959, pp. 547-555.

72. Orlob, Gerald T., Shubinski, R. P., and Feigner, K. D., "Mathematical Modeling of Water Quality in Estuarial Systems," Proceedings National Symposium on Estuarine Pollution, ASCE, Stnaford University, August 1967, pp. 646-675.

73. Owens, M., Edward, R. W., and Gibbs, J. W., "Some Reaeration Studies in Streams," International Journal Air and Water Pollution, Vol. 8, Pergamon Press, Great Britain, 1964, pp. 469-486.

74. Peaceman, D. W., and Rachford, H. H., "The Numerical Solution of Parabolic and Elliptic Differential Equations," Journal Society Industrial Applied Mathematics, Vol. 3, pp. 28-41.

75. Pence, George D., Jeglic, John M., and Thomann, Robert V., "The Development and Application of a Time-Varying Dissolved Oxygen Model," National Symposium on Estuarine Pollution, ASCE, Stanford University, August 1967, pp. 537-585.

76. Pence, George D., Jeglic, John M., and Thomann, Robert V., "Time-Varying Dissolved-Oxygen Model," Journal Sanitary Engineering Division, ASCE, Vol. 94, No. SA2, Proc. Paper 5915, April 1968, pp. 281-402.

77. Pritchard, Donald W., "A Study of the Salt Balance in a Coastal Plain Estuary," Journal of Marine Research, Vol. 13, No. 1, 1954, pp. 133-144.

78. Pritchard, Donald W., "The Equations of Mass Continuity and Salt Continuity in Estuaries," Journal of Marine Research, Vol. 17, No. 3, 1958, pp. 412-423.

79. Pritchard, Donald W., "The Dynamic Structure of a Coastal Plain Estuary," Journal of Marine Research, Vol. 15, No. 1, 1956, pp. 34-42.

80. Pritchard, Donald W., "The Movement and Mixing of Contaminants in Tidal Es uaries," Proceedings, 1st Conference on Waste Disposal in the Marine Environment, E. A. Pearson, ed., Pergamon Press, New York, 1960.

81. Pritchard, Donald W., "Dispersion and Flushing of Pollutants in Estuaries," Journal Hydraulics Division, ASCE, Vol. 95, No. HY1, Proc. Paper 6344, 1969, pp. 115-124.

82. Rattray, Maurice, Jr., and Hansen, Donald V., "A Similarity Solution for Circulation in an Estuary," Journal Marine Research, Vol. 20, No. 2, 1962, pp. 121-133.

83, Rattray, Maurine, Jr., "Some Aspects of the Dynamics of Circulation in Fjords," in Estuaries, Lauff, George H., ed., Publication No. 83, American Association for the Advancement of Science, Washington, D. C., 1967, pp. 52-62.

84. Reid, Robert O., and Bodine, Bernie R., "Numerical Model for Storm Surges in Galveston Bay," Journal Waterways and Harbors Division, ASCE, Vol. 94, No. WW1, Proc. Paper 5805, February 1968, pp. 33-57.

85. Reynolds, Tom D., and Eckenfelder, W. Wesley, "Reaction Rates of Houston Ship Channel Waters," prepared for the Galveston Bay Study, Texas Water Quality Board, March 1970.

86. Sabersky, Rolf H., and Acosta, Allan J., Fluid Flow, The Macmillan Company, New York, 1964.

87. Schneider, Peter W., "Reaction Rates and Apparent Toxicity of Houston Ship Channel Water," thesis presented to Texas A&M University, at College Station, Texas, in May 1969, in partial fulfillment of the requirements for the degree of Master of Science, Department of Civil Engineering.

88. Shankar, Nelakantan J., and Masch, Frank D., "Influence of Tidal Inlets on Salinity and Related Phenomena in Estuaries," Center for Research in Water Resources, Department of Civil Engineering, The University of Texas at Austin, Texas, Technical Report HYD-16-7001, CRWR49, March 1970.

89. Shankar, Nelakantan J., "Influence of Tidal Inlets on Salinity and Related Phenomena in Estuaries," thesis presented to The University of Texas, at Austin, Texas, in January 1971, in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Department of Civil Engineering.

90. Shubinski, Robert P., McCarty, James C., and Lindorf, Marvin R., "Computer Simulation of Estuarial Networks," Journal Hydraulics Division, ASCE, Vol. 91, No. HY5, Proc. Paper 4470, September 1965, pp. 33-49.

91. Smith, G. D., Numerical Solution of Partial Differential Equations, Oxford University Press, New York, 1965.

92. Sparr, Ted M., Sprague, Charles R., and Hann, Roy W., "A Study of the Flushing Times of the Houston Ship Channel and Galveston Bay," Estuarine Systems Projects Technical Report No. 12, Civil Engineering Department, Texas A&M University, May 1970.

93. Stommel, Henry, "Computation of Pollution in a Vertically Mixed Estuary," Sewage and Industrial Wastes, Vol. 25, No. 9, September 1953, pp. 1065-1071.

94. Streeter, H. W., and Phelps, E. B., U. S. Public Health Bulletin 146, 1925.

95. Sverdrup, H. U., Johnson, Martin W., and Fleming, Richard H., The Oceans: Their Physics, Chemistry, and General Biology, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1942.

96. Texas Water Development Board, "HYD-I Program Description," Systems Engineering Division, Austin, Texas, report in preparation.

97. Thomann, Robert V., "The Use of Systems Analysis to Describe the Time Variation of Dissolved Oxygen in a Tidal Stream," thesis presented to New York University, at New York, New York, in 1963, in partial fulfillment of the requirements of the degree of Doctor of Philosophy, Department of Meteorology and Oceanography.

98. Thomann, Robert V., "Mathematical Model for Dissolved Oxygen," Journal Sanitary Engineering Division, ASCE, Vol. 89. No. SA5, October 1963, pp. 1-30.

99. Thomann, Robert V., and Sobel, Matthew J., "Estuarine Water Quality Management and Forecasting," Journal Sanitary Engineering Division, ASCE, Vol. 90, No. SA5, Proc. Paper 4116, October 1964, pp. 9-36.

100. Thomann, Robert V., "Recent Results from a Mathematical Model of Water Pollution Control in the Delaware Estuary," Water Resources Research, Vol. 1, No. 3, 1965, pp. 349-359.

101. Thomann, Robert V., and Marks, David H., "Results from a Systems Analysis Approach to the Optimum Control of Estuarine Water Quality," Third International Conference on Water Pollution Research, Munich, Germany, Sec. 3, Paper No. 2, 1966, pp. 1-14.

102. Thomann, Robert V., "Time-Series Analysis of Water Quality Data," Journal Sanitary Engineering Division, ASCE, Vol. 93, No. SA1, Proc. Paper 5108, February 1967, pp. 1-23.

103. Tsivoglou, E. C., Cohen, J. B., Scearer, S. O., and Godsil, P. J., "Tracer Measurements of Stream Reaeration. II. Field Studies," Journal Water Pollution Control Federation, Vol. 40, No. 2, Part I, February 1968, pp. 285-304.

104. Water Pollution Research Laboratory, "Effects of Polluting Discharges on the Thames Estuary," Water Pollution Research Technical Paper No. 11, Her Majesty's Stationery Office, London, 1964.

105. Water Resources Engineers, Inc., "A Water Quality Model of the Sacramento-San Joaquin Delta," Report of an Investigation for U. S. Public Health Service, Lafayette, California, June 1965.

106. Water Resources Engineers, Inc., "A Hydraulic-Water Quality Model of Suisun and San Pablo Bays," Report of an Investigation for the Federal Water Pollution Control Administration, Lafayette, California, March 1966.

107. Wicker, C. F., editor, "Evaluation of Present State of Knowledge of Factors Affecting Tidal Hydraulics and Related Phenomena," Report No. 3, Committee on Tidal Hydraulics, U. S. Army Corps of Engineers, Vicksburg, Mississippi, May 1965.

108. Wilson, Robert E., "A Study of the Dispersion and Flushing of Water-Borne Materials in the Northwest Branch of Baltimore Harbor, and the Relationship Between These Physical Processes and the Water Quality of the Inner Harbor," Chesapeake Bay Institute Technical Report No. 64, John Hopkins University, March 1970.

109. Withers, Richard E., and Hann, Roy W., "Salinity Patterns in the Houston Ship Channel," Estuarine Systems Projects Technical Report No. 2, Civil Engineering Department, Texas A&M University, August 1969.

110. Young, P. Jonathan, "Tidal Velocity Model and Stationing for the Houston Ship Channel," Estuarine System Projects Report No. 17, Civil Engineering Department, Texas A&M University, 1970.

APPENDIX II. - NOTATION

Dimensions are shown in brackets and are represented as follows: M = mass; L = length; and T = time.

a = abbreviated notation for a(x,z)

a(x,z) = left-hand boundary of estuary; lower limit of integration across the width, [L]

A = average top salinity (equation 4-8 only)

A_i = constant coefficient for Crank-Nicolson method

A_x = cross-sectional area at location x, $[L^2]$

A(X) = coefficient array for Crank-Nicolson method

b = abbreviated notation for b(x,z)

b(x,z) = right-hand boundary of estuary; upper limit of integration across the width, [L]

B = seaward distance, [L]

B = average bottom salinity (equation 4-8 only)

B_i = constant coefficient for Crank-Nicolson method

$B'_i = B_i - 2$

BOD_{20} = ultimate or 20-day BOD

B(X) = coefficient array for Crank-Nicolson method

BOD = concentration of biochemical oxygen demand

C = salinity of the ocean or bay outside the estuary being studied (equation 4-8 only)

C = concentration

C_6 = concentration in segment 6

C_A = concentration of constituent A

$\overline{C}_A$ = time averaged concentration of constituent A

C'_A = perturbation component of the concentration of constituent A

$C_{i,j}^{(k)}$ = concentration at location i,j, at time t (equation 4-17 only)

C_o = concentration at an outfall

$C_{x,t}$ = concentration at location x, at time t

$C_{x,z,t}$ = concentration at location x,z, at time t

$CSTAR_{x,z}$ = intermediate concentration value at location x,z, at completion of the first phase of the implicit alternating-direction method

C_{SAT} = saturation value for dissolved oxygen

D = molecular diffusion coefficient, $[L^2T^{-1}]$

D_A = molecular diffusion coefficient for constituent A, $[L^2T^{-1}]$

D_i = constant coefficient for Crank-Nicolson method

D'_o = apparent diffusion coefficient at x = 0, the mouth of the estuary, $[L^2T^{-1}]$

D_t = longitudinal coefficient of turbulent diffusion, $[L^2T^{-1}]$

D'_x = apparent diffusion coefficient, $[L^2T^{-1}]$

DO = dissolved oxygen concentration

DS = degree of stratification

D(X) = array for Crank-Nicolson method

e,exp = base of natural logarithms, equals 2.718..., usually raised to a power erfc = coerror function

E = dispersion coefficient, generally for the x-direction, $[L^2T^{-1}]$

E_i = turbulent diffusion component $[L^2T^{-1}]$

$E'_{k,k+1}$ = eddy exchange coefficient at interface between segment k and segment k+1, $[L^3T^{-1}]$

E_x, E_y, E_z = directional components of turbulent diffusion coefficient, $[L^2T^{-1}]$

$E_{x_{i,j}}$ = dispersion component in the x-direction at location i,j, $[L^2T^{-1}]$

$E_{y_{i,j}}$ = dispersion component in the y-direction at location i,j, $[L^2T^{-1}]$

EX = dispersion coefficient for the x-direction, $[L^2T^{-1}]$

$EX_{x,z}$ = dispersion coefficient for the x-direction at location x,z, at time t, $[L^2T^{-1}]$

$EXNEXT_{x,z}$ = dispersion coefficient for the x-direction at location x,z at time $t + \Delta t$, $[L^2T^{-1}]$

EXPON1 = exponent expression occurring in an extrapolation technique

EXPON2 = exponent expression occurring in an extrapolation technique

EZ = dispersion coefficient for the z-direction, $[L^2T^{-1}]$

$EZ_{x,z}$ = dispersion coefficient for the z-direction at location x,z, at time t, $[L^2T^{-1}]$

$EZNEXT_{x,z}$ = dispersion coefficient for the z-direction at location x,z, at time $t + \Delta t$, $[L^2T^{-1}]$

f = bed resistance coefficient, [dimensionless]

Fo = FROUDE number, [dimensionless]

g = acceleration of gravity, $[LT^{-2}]$

G = rate of energy dissipation per unit mass of fluid, $[L^2T^{-3}]$

$G(X)$ = coefficient array for Crank-Nicolson method

h = mean depth, [L]

H = tidal height, [L]

H = time zero, point in time at which tracer is released

H = water level elevation relative to the local mean sea level datum (equations 4-13 and 4-15 only), [L]

J = rate of gain of potential energy per unit mass of fluid, $[L^2T^{-3}]$

J_k = direct sources of biochemical oxygen demand in segment k; added concentration per unit time

K = frictional resistance term, $[L^{-1}]$

K = Van Dorn coefficient for wind stress (equations 4-13 and 4-14 only), [dimensionless]

K_d = decay rate; removal coefficient to the base e, $[T^{-1}]$

KD = decay rate to the base e, $[T^{-1}]$

KD_x = decay rate, to the base e, at location x, $[T^{-1}]$

L_k = concentration of ultimate carbonaceous biochemical oxygen demand in estuary segment k

m = amount per unit cross-sectional area for an instantaneous release into a one-dimensional estuary

m = amount per unit cross-sectional area per unit time for a continuous release into a one-dimensional estuary

m = amount per unit depth for an instantaneous release into a two-dimensional estuary

N = number of points

$\vec{N}$ = mass flux, $[ML^{-2}T^{-1}]$

$\vec{N}_A$ = mass flux of constituent A across a boundary, $[ML^{-2}T^{-1}]$

N_x, N_y, N_z = directional components of the mass flux, $[ML^{-2}T^{-1}]$

NX = number of points in the x-direction

NZ = number of points in the z-direction

ppm = parts per million parts

P_t = tidal prism, $[L^3]$

Q = flowrate, $[L^3T^{-1}]$

Q = flowrate of water in pounds per day (equations 8-6 and 8-11 only)

Q = vector average of transport per unit width (equations 4-13 and 4-14 only), $[L^2T^{-1}]$

Q_f = fresh water discharge, $[L^3T^{-1}]$

$Q_{k,k+1}$ = net flow across the interface between segment k and segment k+1, $[L^3T^{-1}]$

r = time rate of production, $[ML^{-3}T^{-1}]$

r_A = time rate of production of constituent A, $[ML^{-3}T^{-1}]$

R = rainfall rate, $[LT^{-1}]$

$\bar{s}_{lws}$ = local salinity at low water slack

s_o = ocean salinity

S_i = term defined for Gaussian elimination method

SLUG = point concentration used for an instantaneous release

t = time, [T]

T = tidal period, [T]

u = net velocity, $[LT^{-1}]$

u_o = maximum flood tide velocity at x = 0, $[LT^{-1}]$

U = advective velocity, generally in the x-direction, $[LT^{-1}]$

U = vertically integrated x-component of transport per unit width (equations 4-13 to 4-15 only), $[L^2T^{-1}]$

$U_{i,j}$ = velocity component in the x-direction at location i,j,$[LT^{-1}]$

V = advective velocity, generally in the y-direction, $[LT^{-1}]$

V = vertically integrated y-component of transport per unit width (equations 4-13 to 4-15 only), $[L^2T^{-1}]$

$\vec{V}$ = velocity vector, $[LT^{-1}]$

V_i = a time-averaged component of velocity, $[LT^{-1}]$

V'_i = perturbation component for turbulent velocity, $[LT^{-1}]$

$V_{i,j}$ = velocity component in the y-direction at location i,j, $[LT^{-1}]$

V'_x = perturbation component in the x-direction for turbulent velocity, $[LT^{-1}]$

V_x, V_y, V_z = directional velocity components, $[LT^{-1}]$

VX = velocity component in the x-direction, $[LT^{-1}]$

$VX_{x,z}$ = velocity component in the x-direction at location x,z, at time t, $[LT^{-1}]$

$VXNEXT_{x,z}$ = velocity component in the x-direction at location x,z, at time $t + \Delta t$, $[LT^{-1}]$

VZ = velocity component in the z-direction, $[LT^{-1}]$

$VZ_{x,z}$ = velocity component in the z-direction at location x,z, at time t, $[LT^{-1}]$

$VZNEXT_{x,z}$ = velocity component in the z-direction at location x,z, at time $t + \Delta t$, $[LT^{-1}]$

VOL_k = volume of segment k

W = b(x,z)-a(x,z); width of estuary at location x,y, [L]

W = wind speed 10 meters above the water (equations 4-13 and 4-14 only), $[LT^{-1}]$

W = loading rate in pounds per day (equations 8-6 to 8-11 only)

W_i = collection of terms for Crank-Nicolson method

W_x = width at location x, [L]

x = longitudinal direction or distance, [L]

y = lateral direction or distance, [L]

z = vertical direction or distance, [L]

Z = depth of water, [L]

Symbols

α_i = term defined for Gaussian elimination method

Δt = time increment, [T]

Δx = x-increment, [L]

Δy = y-increment, [L]

Δz = z-increment, [L]

Θ = weighting factor for finite difference equation

$\xi_{k,k+1}$ = advective coefficient dependent upon the ratio of dispersion to advective forces at interface between segment k and segment k+1, [dimensionless]

Π = 3.14159.....

ρ = mass density, $[ML^{-3}]$

ρ_A = mass density of a mixture with constituent A, $[ML^{-3}]$

Ψ = angle between the wind velocity vector and the x-axis

Subscripts

1,2, etc. = location

A = constituent A

d = decay

f = fresh water

i = a general index

i = location i

j = location j

k = segment designation in a segmented estuary

NX = value at location NX, the final grid point in the x-direction

NZ = value at location NZ, the final grid point in the z-direction

o = origin; x = o

o = outfall

o = ocean (equation 4-12 only)

t = tide (equation 4-10 only)

t = time

t = turbulence

x = the x-component; in the longitudinal direction

y = the y-component; in the lateral direction

z = the z-component; in the vertical direction

Superscripts

(k) = at time t

(k+1) = at time $t + \Delta t$

' = a perturbation component resulting from turbulent fluctuations

' = apparent value

' = dispersion term for Thomann method

APPENDIX III. - COMPUTER PROGRAMS AND DATA

IDEAL-I

IDEAL-II

MASSTRANS-I

MASSTRANS-II

OXTRANS-I

STABLE-I

STABLE-II

EXACT-I

EXACT-II

PROFILE-I

PROFILE-II

COMPUTER PROGRAM FOR

IDEAL-I

Object Code	=	7912 bytes
Array Area	=	1616 bytes
Total	=	9528 bytes

```
C
C                   *****************************************
C                   *                                       *
C                   *               IDEAL-I                 *
C                   *                                       *
C                   *  ONE DIMENSIONAL MASS TRANSPORT MODEL *
C                   *                                       *
C                   *   EXPLICIT FINITE DIFFERENCE METHOD   *
C                   *                                       *
C                   *         CONSTANT COEFFICIENTS         *
C                   *                                       *
C                   *---------------------------------------*
C                   *                                       *
C                   *    LOGIC AND PROGRAMMING BY           *
C                   *                                       *
C                   *        JONATHAN YOUNG                 *
C                   *        ENVIRONMENTAL ENG. DIVISION    *
C                   *        CIVIL ENGINEERING DEPT.        *
C                   *        TEXAS A&M UNIVERSITY           *
C                   *        COLLEGE STATION, TEXAS         *
C                   *                                       *
C                   *****************************************
C
C
C                        THIS PROGRAM USES AN EXPLICIT FINITE
C         DIFFERENCE METHOD TO ANALYZE MASS TRANSPORT IN A ONE DIMEN-
C         SIONAL ESTUARY.  THE MODEL IS USEFUL IN EVALUATING TRANSPORT IN
C         A FLUME OR WELL-MIXED ESTUARY WHERE VELOCITY, DISPERSION,
C         DECAY, AND CROSS-SECTIONAL AREA CAN BE CONSIDERED CONSTANT
C         THROUGHOUT.
C
```

```
C           THIS PROGRAM WAS WRITTEN PRIMARILY TO TEST THE ACCURACY
C        OF A FINITE DIFFERENCE METHOD AS COMPARED WITH ANALYTICAL,
C        CLOSED FORM SOLUTIONS FOR THE ESTUARY MASS TRANSPORT EQUATION.
C        REPRESENTATIVE VALUES FOR DISPERSION,VELOCITY, AND DECAY RATES
C        SHOULD BE CHOSEN FOR THE ESTUARY BEING STUDIED.  THESE VALUES
C        CAN BE USED SEVERAL TIMES IN THIS MODEL TO DETERMINE WHICH
C        TIME AND DISTANCE INCREMENTS GIVE SATISFACTORY ACCURACY.
C        STABILITY CRITERIA IS PRINTED BY THE PROGRAM AND SEVERE
C        INACCURACIES WILL RESULT IF THESE LIMITS ARE EXCEEDED.
C
C           THE PROGRAM CAN ANALYZE INSTANTANEOUS RELEASES, CONTINUOUS
C        RELEASES, OR STEADY-STATE PROFILES.  EXPONENTIAL EXTRAPOLATION
C        AT THE BOUNDARIES IS USED FOR INSTANTANEOUS RELEASES.  FOR
C        OTHER TYPES OF LOADING, A CONSTANT SLOPE EXTRAPOLATION IS USED.
C
C           INFORMATION REGARDING THIS COMPUTER PROGRAM CAN BE
C        OBTAINED FROM JONATHAN YOUNG AT HYDROSCIENCE,INC., WESTWOOD,
C        NEW JERSEY.
C
C***********************************************************************
```

```
C***********************************************************************
C
C
C          ***INPUT VARIABLES***
C
C        N         = NUMBER OF POINTS
C        DELT      = TIME INCREMENT (DAYS)
C        DELX      = DISTANCE INCREMENT (MILES)
C        E         = DISPERSION COEFFICIENT (MI**2/DAY)
C        XKD       = DECAY RATE ( /DAY)
C        U         = VELOCITY (MILES/DAY)
C        NC        = NUMBER OF INITIAL CONCENTRATIONS
C        NS        = NUMBER OF SOURCES ( MAXIMUM OF 2 )
C        LTYPE     = TYPE OF LOADING
C                      1 MEANS CONSTANT CONCENTRATION AT OUTFALL
C                        ( CAN BE USED FOR STEADY-STATE CALCULATIONS )
C                      2 MEANS AN INSTANTANEOUS RELEASE
C                      3 MEANS CONTINUOUS LOADING
C        NDTPR     = PRINTING INCREMENT FOR TIME STEPS (INTEGER)
C        NSKIP     = PRINTING INCREMENT FOR DISTANCE STEPS (INTEGER)
C        TBEGIN    = TIME OF FIRST CALCULATION (DAYS)
C        TSTOP     = TIME OF FINAL CALCULATION (DAYS)
C        JJ        = POINT OF AN INITIAL CONCENTRATION
C        LOAD(JJ)  = INITIAL CONCENTRATION
C        MASS      = MASS RATE BEING ADDED AT OUTFALL (OPTIONAL)
C
C*******************************************************************
C
```

```
C
C           ***OTHER VARIABLES***
C
C        MSU        = POINT OF PEAK OF UPPERMOST SOURCE
C        MSL        = POINT OF PEAK CF LOWERMOST SOURCE
C        UPPER      = MILEAGE AT MSU
C        LOWER      = MILEAGE AT MSL
C        UMAX       = CONCENTRATION AT MSU
C        LMAX       = CONCENTRATION AT MSL
C        NSEG       = NUMBER OF SEGMENTS (=N-1)
C        COUNT      = A COUNTER TO COMPARE WITH PRINTING INCREMENT
C        X(IJ)      = DISTANCES TO BE PRINTED
C      NN1,IJ,K,L   = DO LOOP INDICES
C        QUIT       = NUMBER OF TIME INCREMENTS
C        NOMORE     = DO LOOP PARAMETER (=QUIT+1)
C        T          = TIME (DAYS)
C        COEF1      = DISPERSION TERM COEFFICIENT
C        COEF2      = VELOCITY TERM COEFFICIENT
C        COEF3      = DECAY TERM COEFFICIENT
C        CT(K)      = CONCENTRATION AT PREVIOUS TIME INCREMENT
C        CT1(K)     = CONCENTRATION BEING CALCULATED
C        EXPON      = EXPONENT FOR BOUNDARY CONDITION CALCULATIONS
C        XUP        = ORIGINAL LOCATION OF UPPERMOST PEAK OR OUTFALL
C        XLOW       = ORIGINAL LOCATION OF LOWERMOST PEAK OR OUTFALL
C        STABLX     = MAXIMUM X INCREMENT TO INSURE STABILITY
C        STABLT     = MAXIMUM TIME INCREMENT TO INSURE STABILITY
C        CNTINU     = SUBROUTINE TO CALCULATE CONTINUOUS LOADING VALUES
C        NCALC      = APPROXIMATE NUMBER OF PROGRAM CALCULATIONS
C        INDEX      = N*NOMORE
C
C*********************************************************************
```

```
      REAL LOAD,LOWER,LMAX,MASS
      INTEGER COUNT
      DIMENSION CT(101),CT1(101),LOAD(101),X(101)
C
C-------------------------------------------------------------------
C     ***INITIALIZE TERMS AND READ INPUT DATA***
C-------------------------------------------------------------------
      DATA CT,CT1,LOAD,X/404*0.0/
      READ ,N,DELT,DELX,E,XKD,U,NC,NS,LTYPE,NDTPR,NSKIP,TBEGIN,TSTOP
      PRINT,N,DELT,DELX,E,XKD,U,NC,NS,LTYPE,NDTPR,NSKIP,TBEGIN,TSTOP
      DO 3 J=1,NC
      READ, JJ, LOAD(JJ)
      CT(JJ)=LOAD(JJ)
    3 CONTINUE
      IF(LTYPE.NE.3) GO TO 4
      READ, MASS
    4 CONTINUE
      NSEG=N-1
      MSU=1
      MSL=N
      UPPER=0.0
      LOWER=(N-1)*DELX
      UMAX=LOAD(1)
      LMAX=LOAD(N)
C
C-------------------------------------------------------------------
C     ***ESTABLISH THE LOCATIONS OF THE UPPERMOST
C         AND LCWERMOST WASTE SOURCES***
C-------------------------------------------------------------------
```

```
      DO 6 I=1,NSEG
      IF(LOAD(I+1).LT.LOAD(I)) GO TO 7
      MSU=I+1
      UPPER=(I)*DELX
      UMAX=LOAD(I+1)
    6 CONTINUE
    7 DO 8 I=1,NSEG
      IF(LOAD(N-I).LT.LOAD(N+1-I)) GO TO 9
      MSL=N-I
      LOWER=(MSL-1)*DELX
      LMAX=LOAD(N-I)
    8 CONTINUE
    9 CONTINUE
C
C-----------------------------------------------------------------------
C     ***DETERMINE THE INITIAL VALUE OF COUNT***
C-----------------------------------------------------------------------
      COUNT=(TBEGIN/DELT + 0.01)
      COUNT=COUNT-(COUNT/NDTPR)*NDTPR
C
C-----------------------------------------------------------------------
C     ***DETERMINE STABILITY CRITERIA***
C-----------------------------------------------------------------------
      STABLX=DELX
      STABLT=DELT
      IF(E.GT.0.0.AND.U.GT.0.0) STABLX=2.*E/U
      TERM=2.*E+DELX*DELX*XKD
      IF(TERM.NE.0.0) STABLT=(DELX*DELX)/TERM
```

```
C
C-----------------------------------------------------------------------
C      ***PRINT DATA VALUES***
C-----------------------------------------------------------------------
      PRINT 10
   10 FORMAT(1H1)
      PRINT 11
   11 FORMAT(1H1,////, 15X, '*****DISPERSION CALCULATIONS BY THE EXPLICI
     $T METHOD*****', ///)
      PRINT 12, N
   12 FORMAT(22X, ' NUMBER OF POINTS = ',I3,/)
      SECS=DELT*86400.
      PRINT 13, DELT,SECS
   13 FORMAT(25X, 'TIME INCREMENT = ',F7.4, ' DAYS ( ', F8.2,' SECONDS )
     $',/)
      FEET=DELX*5280.
      PRINT 14, DELX, FEET
   14 FORMAT(21X, 'DISTANCE INCREMENT = ', F7.3, ' MILES ( ', F8.2, ' FE
     $ET )',/)
      EFEET=E*5280.*5280./86400.
      PRINT 15, E,EFEET
   15 FORMAT(17X, 'DISPERSION COEFFICIENT = ',F7.2, ' MILES SQUARED/DAY
     $( ', F7.2, ' FEET SQUARED/SEC )',/)
      PRINT 16, XKD
   16 FORMAT(22X, 'DECAY COEFFICIENT = ', F7.2, ' PER DAY',/)
      UF=U*5280./86400.
      PRINT 17, U,UF
   17 FORMAT(31X, 'VELOCITY = ', F7.2, ' MILES/DAY ( ', F6.3, ' FEET/SEC
     $ )',/)
      PRINT 18, NC
   18 FORMAT(1X, 'NUMBER OF KNOWN INITIAL CONCENTRATIONS = ', I3,/)
```

```
      PRINT 185, NS
  185 FORMAT(22X, 'NUMBER OF SOURCES = ', I3,/)
      PRINT 19, LTYPE
   19 FORMAT( 16X, 'TYPE OF LOADING (LTYPE) = ', I2,/)
      DTPR=NDTPR*DELT
      WRITE(6,20)DTPR
   20 FORMAT(16X, 'PRINTOUT TIME INCREMENT = ', F7.4, ' DAYS',/)
      DXPR=NSKIP*DELX
      WRITE(6,22) DXPR
   22 FORMAT(12X, 'PRINTOUT DISTANCE INCREMENT = ', F7.4, ' MILES',/)
      PRINT 221, STABLX
  221 FORMAT( 13X, 'MAXIMUM STABLE X INCREMENT = ', F7.3,' MILES',/)
      PRINT 222, STABLT
  222 FORMAT( 13X, 'MAXIMUM STABLE T INCREMENT = ', F7.4, ' DAYS',/)
      PRINT 23, TBEGIN, TSTOP
   23 FORMAT(26X, 'RANGE OF TIME = ', F5.3, ' TO ', F8.3,' DAYS',/,1H1)
      T=TBEGIN
      WRITE(6,24)
   24 FORMAT(1X, 119('*'),/)
C
C-----------------------------------------------------------------------
C     ***CALCULATE DISTANCES TO BE PRINTED***
C-----------------------------------------------------------------------
      DO 25 IJ=1,N,NSKIP
      X(IJ)=(IJ-1)*DELX
   25 CONTINUE
      NN1=1+NSKIP
C
C-----------------------------------------------------------------------
C     ***PRINT DISTANCES***
C-----------------------------------------------------------------------
```

```
      WRITE(6,27) (X(IJ), IJ=NN1,N,NSKIP)
   27 FORMAT( ' X VALUES IN MILES', 10F10.2,/,1(18X,10F10.2))
      WRITE(6,24)
C
C-----------------------------------------------------------------------
C     ***PRINT INITIAL CONCENTRATIONS***
C-----------------------------------------------------------------------
      WRITE(6,32)
   32 FORMAT(4X, 'TIME',45X, 'CONCENTRATIONS (PPM)',/)
      WRITE(6,35)T,(CT(J),J=1,N,NSKIP)
   35 FORMAT(F8.2,F10.4,10F10.4,/,1(18X,10F10.4))
C
C-----------------------------------------------------------------------
C     ***CALCULATE CONSTANT COEFFICIENTS FOR
C          THE FINITE DIFFERENCE EQUATION***
C-----------------------------------------------------------------------
      COEF1=E*DELT/(DELX**2.)
      COEF2=U*DELT/(2.*DELX)
      COEF3=XKD*DELT
      QUIT=(TSTOP-TBEGIN)/DELT
      NOMORE=QUIT+1
C
C-----------------------------------------------------------------------
C     ***BEGIN LOOP WHICH IS REPEATED FOR EACH TIME INCREMENT***
C-----------------------------------------------------------------------
      DO 200 JJ=1,NOMORE
      T=T+DELT
      COUNT=COUNT+1
```

```
C
C-----------------------------------------------------------------------
C     ***APPLY EXPLICIT FINITE DIFFERENCE EQUATION
C          TO EACH INTERNAL POINT***
C-----------------------------------------------------------------------
      DO 50 K=2,NSEG
      CT1(K)=COEF1*(CT(K+1)-2*CT(K)+CT(K-1))-COEF2*(CT(K+1)-CT(K-1))-COE
     2F3*CT(K)+CT(K)
   50 CONTINUE
C
C-----------------------------------------------------------------------
C     ***DETERMINE IF CURVE IS TO BE EXTENDED UPSTREAM AND
C          APPLY THE APPROPRIATE BOUNDARY CONDITIONS***
C-----------------------------------------------------------------------
      IF(MSU.EQ.1.AND.LTYPE.EQ.1) GO TO 55
      IF(LTYPE.NE.2) GO TO 53
      XUP=UPPER-U*TBEGIN
      EXPON= (+2.*(-XUP  +DELX) * DELX-(DELX**2.) - 2.*DELX*U*T) /
     $   (4.*E*T)
      CT1(1)=CT1(2)*EXP(EXPON)
      GO TO 54
   53 CONTINUE
      CT1(1)=2.*CT1(2)-CT1(3)
   54 CONTINUE
      IF(CT1(3).LE.0.001) GO TO 55
      IF(CT1(1).LT.0.0) CT1(1)=CT1(2)*CT1(2)/CT1(3)
   55 CONTINUE
      IF(CT1(1).LT.0.0) CT1(1)=0.0
C
C-----------------------------------------------------------------------
C     ***DETERMINE IF CURVE IS TO BE EXTENDED DOUNSTREAM AND
C          APPLY THE APPROPRIATE BOUNDARY CONDITIONS***
C-----------------------------------------------------------------------
```

```
      IF(MSL.EQ.N.AND.LTYPE.EQ.1) GO TO 59
      IF(LTYPE.NE.2) GC TO 57
      XLOW=LOWER-U*TBEGIN
      EXPON= (-2.*((N-2)*DELX-XLCW ) * DELX- (DELX**2.) + 2.*DELX*U*T)/
     $     (4.*E*T)
      CT1(N)=CT1(N-1)*EXP(EXPON)
      GO TO 58
   57 CONTINUE
      CT1(N)=2.*CT1(N-1)-CT1(N-2)
   58 CONTINUE
      IF(CT1(N-2).LE.0.001) GO TO 59
      IF(CT1(N).LT.0.0) CT1(N)=CT1(N-1)*CT1(N-1)/CT1(N-2)
   59 CONTINUE
      IF(CT1(N).LT.0.0) CT1(N)=0.0
C
C-----------------------------------------------------------------------
C         ***APPLY APPROPRIATE LOADING CONDITIONS***
C-----------------------------------------------------------------------
      IF(LTYPE.EQ.1) GC TO 60
      IF(LTYPE.EQ.3) CALL CNTINU(E,T,MASS,CT1(MSU))
      GO TO 70
   60 CONTINUE
      CT1(MSU)=UMAX
      CT1(MSL)=LMAX
   70 CONTINUE
      IF(COUNT.GE.NDTPR) GO TO 75
      GO TO 100
C
C-----------------------------------------------------------------------
C      ***PRINT CONCENTRATIONS FOR EACH DESIRED TIME INCREMENT***
C-----------------------------------------------------------------------
   75 WRITE(6,80) T,(CT1(M),M=1,N,NSKIP)
   80 FORMAT(/, F8.3, F10.4, 10F10.4, /, 1(18X, 10F10.4))
```

```
      COUNT=0
  100 DO 150 L=1,N
      IF(CT1(L).LT.0.001)  CT1(L)=0.0
  150 CT(L)=CT1(L)
  200 CONTINUE
 1000 CONTINUE
C
C-----------------------------------------------------------------------
C     ***DETERMINE THE NUMBER OF CALCULATIONS MADE BY THE PROGRAM
C-----------------------------------------------------------------------
      NCALC=3*NS + 10*NSEG +  2*N/NSKIP + NOMORE*(5*NSEG + 32)
      INDEX=N*NOMORE
      PRINT 205, N,NOMORE,NCALC, INDEX
  205 FORMAT(/////, 1X, 'N=', I3, 5X, 'NOMORE=', I5, 5X, 'NCALC=', I7,
     $ 5X, 'INDEX=', I6,////)
      STOP
      END
C-----------------------------------------------------------------------
      SUBROUTINE CNTINU(EX,TIME,XMASS,CONCIN)
C
C             THIS SUBROUTINE SHOULD BE WRITTEN ACCORDING TO THE
C        PARTICULAR LOADING RATE OF THE OUTFALL BEING STUDIED.
C        THE SUBROUTINE WRITTEN BELOW IS VALID ONLY IN THE CASE OF
C        ONE OUTFALL AND WHERE NO VELOCITY IS PRESENT.
C
C          THIS SUBROUTINE IS USED ONLY IF LTYPE EQUALS 3.
C
C
C-----------------------------------------------------------------------
      RAD=3.14159*EX
      CONCIN=(XMASS*TIME**.5)/SQRT(RAD)
      RETURN
      END
```

Input Data for IDEAL-I

Each line represents a new card unless single spaced.

51, 0.002, 0.10, 1., 0., 5., 1, 1, 2, 25, 1, 0., 0.6

11

18939.4

COMPUTER PROGRAM FOR

IDEAL-II

Object Code =	10,968 bytes
Array Area =	2,828 bytes
Total =	13,796 bytes

```
C
C                  ****************************************
C                  *                                      *
C                  *               IDEAL-II               *
C                  *                                      *
C                  *  ONE DIMENSIONAL MASS TRANSPORT MODEL *
C                  *                                      *
C                  *   IMPLICIT FINITE DIFFERENCE METHOD  *
C                  *                                      *
C                  *        CONSTANT COEFFICIENTS         *
C                  *                                      *
C                  *--------------------------------------*
C                  *                                      *
C                  *    LOGIC AND PRCGRAMMING BY          *
C                  *                                      *
C                  *        JCNATHAN YCUNG                *
C                  *        ENVIRONMENTAL ENG. DIVISION   *
C                  *        CIVIL ENGINEERING DEPT.       *
C                  *        TEXAS A&M UNIVERSITY          *
C                  *        CCLLEGE STATION, TEXAS        *
C                  *                                      *
C                  ****************************************
C
C            THIS PROGRAM USES AN CRANK-NICOLSON IMPLICIT FINITE
C        DIFFERENCE METHOD TO ANALYZE MASS TRANSPORT IN A ONE DIMEN-
C        SIONAL ESTUARY.  THE MODEL IS USEFUL IN EVALUATING TRANSPORT IN
C        A FLUME OR WELL-MIXED ESTUARY WHERE VELOCITY, DISPERSION,
C        DECAY, AND CROSS-SECTIONAL AREA CAN BE CONSIDERED CONSTANT
C        THROUGHOUT.
C
```

```
C             THIS PROGRAM WAS WRITTEN PRIMARILY TO TEST THE ACCURACY
C         OF A FINITE DIFFERENCE METHOD AS COMPARED WITH ANALYTICAL,
C         CLOSED FORM SOLUTIONS FOR THE ESTUARY MASS TRANSPORT EQUATION.
C         REPRESENTATIVE VALUES FOR DISPERSION,VELOCITY, AND DECAY RATES
C         SHOULD BE CHOSEN FOR THE ESTUARY BEING STUDIED.  THESE VALUES
C         CAN BE USED SEVERAL TIMES IN THIS MODEL TO DETERMINE WHICH
C         TIME AND DISTANCE INCREMENTS GIVE SATISFACTORY ACCURACY.
C         STABILITY CRITERIA IS PRINTED BY THE PROGRAM AND SEVERE
C         INACCURACIES MAY  RESULT IF THESE LIMITS ARE EXCEEDED.
C
C             THE PROGRAM CAN ANALYZE INSTANTANEOUS RELEASES, CONTINUOUS
C         RELEASES, OR STEADY-STATE PROFILES.  EXPONENTIAL EXTRAPOLATION
C         AT THE BOUNDARIES IS USED FOR INSTANTANEOUS RELEASES.  FOR
C         OTHER TYPES OF LOADING, A CCNSTANT SLOPE EXTRAPOLATION IS USED.
C
C             INFORMATICN REGARDING THIS COMPUTER PROGRAM CAN BE
C         OBTAINED FROM JONATHAN YOUNG AT HYDROSCIENCE,INC., WESTWOOD,
C         NEW JERSEY.
C
C***********************************************************************
```

```
C**********************************************************************
C
C
C          ***INPUT VARIABLES***
C
C         N         = NUMBER OF POINTS
C         DELT      = TIME INCREMENT (DAYS)
C         DELX      = DISTANCE INCREMENT (MILES)
C         E         = DISPERSICN COEFFICIENT (MI**2/DAY)
C         XKD       = DECAY RATE ( /DAY)
C         U         = VELOCITY (MILES/DAY)
C         NC        = NUMBER OF INITIAL CONCENTRATIONS
C         NS        = NUMBER OF SOURCES ( MAXIMUM OF 2 )
C         LTYPE     = TYPE OF LOADING
C                       1 MEANS CONSTANT CONCENTRATION AT OUTFALL
C                         ( CAN BE USED FOR STEADY-STATE CALCULATIONS )
C                       2 MEANS AN INSTANTANEOUS RELEASE
C                       3 MEANS CONTINUOUS LOADING
C         NDTPR     = PRINTING INCREMENT FOR TIME STEPS (INTEGER)
C         NSKIP     = PRINTING INCREMENT FOR DISTANCE STEPS (INTEGER)
C         TBEGIN    = TIME OF FIRST CALCULATION (DAYS)
C         TSTOP     = TIME OF FINAL CALCULATION (DAYS)
C         JJ        = POINT OF AN INITIAL CONCENTRATION
C         LOAD(JJ)  = INITIAL CONCENTRATION
C         MASS      = MASS RATE BEING ADDEC AT OUTFALL (OPTIONAL)
C
C****************************************************************
C
```

```
C          ***OTHER VARIABLES***
C
C        MSU        = POINT OF PEAK OF UPPERMOST SOURCE
C        MSL        = POINT OF PEAK OF LOWERMOST SOURCE
C        UPPER      = MILEAGE AT MSU
C        LOWER      = MILEAGE AT MSL
C        UMAX       = CONCENTRATION AT MSU
C        LMAX       = CONCENTRATION AT MSL
C        NSEG       = NUMBER OF SEGMENTS (=N-1)
C        COUNT      = A COUNTER TO COMPARE WITH PRINTING INCREMENT
C        X(IJ)      = DISTANCES TO BE PRINTED
C        QUIT       = NUMBER OF TIME INCREMENTS
C        NOMORE     = DO LOOP PARAMETER (=QUIT+1)
C        T          = TIME (DAYS)
C        A,B*,D     = COEFFICIENTS FOR SOLUTION MATRIX
C        W(M)       = RIGHT SIDE ARRAY FOR SOLUTION MATRIX
C        ALPHA(M)   = INTERMEDIATE ARRAY FOR MATRIX SOLUTION
C        S(M)       = INTERMEDIATE ARRAY FOR MATRIX SOLUTION
C        CT(K)      = CONCENTRATION AT PREVIOUS TIME INCREMENT
C        CT1(K)     = CONCENTRATION BEING CALCULATED
C        EXPON      = EXPONENT FOR BOUNDARY CONDITION CALCULATIONS
C        XUP        = ORIGINAL LOCATION OF UPPERMOST PEAK OR OUTFALL
C        XLOW       = ORIGINAL LOCATION OF LOWERMOST PEAK OR OUTFALL
C        STABLX     = MAXIMUM X INCREMENT TO INSURE STABILITY
C        STABLT     = MAXIMUM TIME INCREMENT TO INSURE STABILITY
C        CNTINU     = SUBROUTINE TO CALCULATE CONTINUOUS LOADING VALUES
C        NCALC      = APPROXIMATE NUMBER OF PROGRAM CALCULATIONS
C        INDEX      = N*NOMORE
C
C******************************************************************
```

```
      REAL LOAD,LOWER,LMAX,MASS
      INTEGER COUNT
      DIMENSION CT(101),CT1(101),LOAD(101),X(101),W(101),ALPHA(101),
     $ S(101)
C
C-----------------------------------------------------------------------
C     ***INITIALIZE TERMS AND READ INPUT DATA***
C-----------------------------------------------------------------------
      DATA CT,CT1,LOAD,X,W,ALPHA,S/707*0.0/
      READ ,N,DELT,DELX,E,XKD,U,NC,NS,LTYPE,NDTPR,NSKIP,TBEGIN,TSTOP
      PRINT,N,DELT,DELX,E,XKD,U,NC,NS,LTYPE,NDTPR,NSKIP,TBEGIN,TSTOP
      DO 3 J=1,NC
      READ, JJ, LOAD(JJ)
      CT(JJ)=LOAD(JJ)
    3 CONTINUE
      IF(LTYPE.NE.3) GO TO 4
      READ, MASS
    4 CONTINUE
      NSEG=N-1
      MSU=1
      MSL=N
      UPPER=0.0
      LOWER=(N-1)*DELX
      UMAX=LOAD(1)
      LMAX=LOAD(N)
C
C-----------------------------------------------------------------------
C     ***ESTABLISH THE LOCATICNS OF THE UPPERMOST
C          AND LCWERMOST WASTE SOURCES***
C-----------------------------------------------------------------------
```

```
      DO 6 I=1,NSEG
      IF(LOAD(I+1).LT.LOAD(I)) GO TO 7
      MSU=I+1
      UPPER=(I)*DELX
      UMAX=LOAD(I+1)
    6 CONTINUE
    7 DO 8 I=1,NSEG
      IF(LOAD(N-I).LT.LOAD(N+1-I)) GO TO 9
      MSL=N-I
      LOWER=(MSL-1)*DELX
      LMAX=LOAD(N-I)
    8 CONTINUE
    9 CONTINUE
C
C-----------------------------------------------------------------------
C     ***DETERMINE THE INITIAL VALUE OF COUNT***
C-----------------------------------------------------------------------
      COUNT=(TBEGIN/DELT + 0.01)
      COUNT=COUNT-(COUNT/NDTPR)*NDTPR
C
C-----------------------------------------------------------------------
C     ***DETERMINE STABILITY CRITERIA***
C-----------------------------------------------------------------------
      STABLX=DELX
      STABLT=DELT
      IF(E.GT.0.0.AND.U.GT.0.0) STABLX=2.*E/U
      TERM=2.*E+DELX*DELX*XKD
      IF(TERM.NE.0.0) STABLT=2.*(DELX*DELX)/TERM
C
```

```
C-----------------------------------------------------------------------
C      ***PRINT DATA VALUES***
C-----------------------------------------------------------------------
       PRINT 10
    10 FORMAT(1H1)
       PRINT 11
    11 FORMAT(1H1,////, 15X, '*****DISPERSION CALCULATIONS BY THE CRANK-N
      $ICOLSON IMPLICIT METHOD*****',///)
       PRINT 12, N
    12 FORMAT(22X, ' NUMBER OF PCINTS = ',I3,/)
       SECS=DELT*86400.
       PRINT 13, DELT,SECS
    13 FORMAT(25X, 'TIME INCREMENT = ',F7.4, ' DAYS ( ', F8.2,' SECONDS )
      $',/)
       FEET=DELX*5280.
       PRINT 14, DELX, FEET
    14 FORMAT(21X, 'DISTANCE INCREMENT = ', F7.3, ' MILES ( ', F8.2, ' FE
      $ET )',/)
       EFEET=E*5280.*5280./86400.
       PRINT 15, E,EFEET
    15 FORMAT(17X, 'DISPERSION COEFFICIENT = ',F7.2, ' MILES SQUARED/DAY
      $( ', F7.2, ' FEET SQUARED/SEC )',/)
       PRINT 16, XKD
    16 FORMAT(22X, 'DECAY COEFFICIENT = ', F7.2, ' PER DAY',/)
       UF=U*5280./86400.
       PRINT 17, U,UF
    17 FORMAT(31X, 'VELOCITY = ', F7.2, ' MILES/DAY ( ', F6.3, ' FEET/SEC
      $ )',/)
       PRINT 18, NC
    18 FORMAT(1X, 'NUMBER OF KNOWN INITIAL CONCENTRATIONS = ', I3,/)
```

```
      PRINT 185, NS
  185 FORMAT(22X, 'NUMBER OF SOURCES = ', I3,/)
      PRINT 19, LTYPE
   19 FORMAT( 16X, 'TYPE OF LOADING (LTYPE) = ', I2,/)
      DTPR=NDTPR*DELT
      WRITE(6,20)DTPR
   20 FORMAT(16X, 'PRINTOUT TIME INCREMENT = ', F7.4, ' DAYS',/)
      DXPR=NSKIP*DELX
      WRITE(6,22) DXPR
   22 FORMAT(12X, 'PRINTOUT DISTANCE INCREMENT = ', F7.4, ' MILES',/)
      PRINT 221, STABLX
  221 FORMAT( 13X, 'MAXIMUM STABLE X INCREMENT = ', F7.3,' MILES',/)
      PRINT 222, STABLT
  222 FORMAT( 13X, 'MAXIMUM STABLE T INCREMENT = ', F7.4, ' DAYS',/)
      PRINT 23, TBEGIN, TSTOP
   23 FORMAT(26X, 'RANGE OF TIME = ', F5.3, ' TO ', F8.3,' DAYS',/,1H1)
      T=TBEGIN
      WRITE(6,24)
   24 FORMAT(1X, 119('*'),/)
C
C-----------------------------------------------------------------------
C     ***CALCULATE DISTANCES TO BE PRINTED***
C-----------------------------------------------------------------------
      DO 25 IJ=1,N,NSKIP
      X(IJ)=(IJ-1)*DELX
   25 CONTINUE
      NN1=1+NSKIP
C
C-----------------------------------------------------------------------
C     ***PRINT DISTANCES***
C-----------------------------------------------------------------------
```

```
      WRITE(6,27) (X(IJ), IJ=NN1,N,NSKIP)
   27 FORMAT( ' X VALUES IN MILES', 10F10.2,/,1(18X,10F10.2))
      WRITE(6,24)
C
C-----------------------------------------------------------------------
C     ***PRINT INITIAL CONCENTRATIONS***
C-----------------------------------------------------------------------
      WRITE(6,32)
   32 FORMAT(4X, 'TIME',45X, 'CONCENTRATICNS (PPM)',/)
      WRITE(6,35)T,(CT(J),J=1,N,NSKIP)
   35 FORMAT(F8.2,F10.4,10F10.4,/,1(18X,10F10.4))
C
C-----------------------------------------------------------------------
C     ***CALCULATE CONSTANT COEFFICIENTS FOR
C          THE FINITE DIFFERENCE EQUATION***
C-----------------------------------------------------------------------
      A=E*DELT/(2.*DELX**2.)+U*DELT/(4.*DELX)
      B1=E*DELT/(DELX**2.)+XKD*DELT/2.+1.
      B2=B1-2.
      D=E*DELT/(2.*DELX**2.)-U*DELT/(4.*DELX)
      QUIT=(TSTOP-TBEGIN)/DELT
      NOMORE=QUIT+1
C
C-----------------------------------------------------------------------
C     ***BEGIN LCOP WHICH IS REPEATED FOR EACH TIME INCREMENT***
C-----------------------------------------------------------------------
      DO 200 JK=1,NOMORE
      T=T+DELT
      COUNT=COUNT+1
C
```

```
C-----------------------------------------------------------------------
C       ***SET THE UPPER BOUNDARY CONDITIONS***
C
C       STATEMENT NUMBER 36 FOR EXPONENTIAL EXTRAPOLATION
C       STATEMENT NUMBER 37 FOR CONSTANT SLOPE EXTRAPOLATION
C       STATEMENT NUMBER 39 FOR CONSTANT CONCENTRATION AT THE BOUNDARY
C-----------------------------------------------------------------------
        IF(MSU.EQ.1.AND.LTYPE.EQ.1) GO TO 39
        IF(LTYPE.NE.2) GO TO 37
     36 CONTINUE
        XUP=UPPER-U*TBEGIN
        EXPON= (+2.*(-XUP  +DELX) * DELX-(DELX**2.) - 2.*DELX*U*T) /
       $   (4.*E*T)
        IF(EXPON.LT.-170.) EXPCN=-170.
        ALPHA(1)=-A*EXP(EXPON)+B1
        W(1)=A*CT(1) -B2*CT(2)+D*CT(3)
        ALPHA(2)=B1-A*D/ALPHA(1)
        GO TO 41
     37 CONTINUE
        ALPHA(1)=-2.*A+B1
        W(1)=A*CT(1)-B2*CT(2)+D*CT(3)
        ALPHA(2)=B1-A*(D-A)/ALPHA(1)
        GO TO 41
     39 CONTINUE
        ALPHA(1)=B1
        W(1)=A*2.*CT(1)-B2*CT(2)+D*CT(3)
        ALPHA(2)=B1-A*D/ALPHA(1)
     41 CONTINUE
C
```

```
C------------------------------------------------------------------------
C          ***BUILD ARRAYS FOR MATRIX SOLUTION***
C------------------------------------------------------------------------
         MOST=NSEG-2
         DO 40 M=2,MOST
         W(M)=A*CT(M)-B2*CT(M+1)+D*CT(M+2)
      40 CONTINUE
C
C------------------------------------------------------------------------
C          ***SOLVE TRIDIAGONAL MATRIX***
C------------------------------------------------------------------------
         S(1)=W(1)
         S(2)=W(2)+A*S(1)/ALPHA(1)
         DO 42 MM=3,MOST
         ALPHA(MM)=B1-A*D/ALPHA(MM-1)
         S(MM)=W(MM)+A*S(MM-1)/ALPHA(MM-1)
      42 CONTINUE
C
C------------------------------------------------------------------------
C        ***SET LOWER BOUNDARY CONDITIONS***
C
C        STATEMENT NUMBER 421 FOR EXPONENTIAL EXTRAPOLATION
C        STATEMENT NUMBER 43 FOR CONSTANT CONCENTRATION AT THE BOUNDARY
C        STATEMENT NUMBER 44 FOR CONSTANT SLOPE EXTRAPOLATION
C------------------------------------------------------------------------
         IF(LTYPE.EQ.1.AND.MSL.EQ.N) GO TO 43
         IF(LTYPE.NE.2) GO TO 44
     421 CONTINUE
         XLOW=LOWER-U*TBEGIN
```

```
      EXPON= (-2.*((N-2)*DELX-XLOW ) * DELX- (DELX**2.) + 2.*DELX*U*T)/
     $    (4.*E*T)
      IF(EXPON.LT.-170.) EXPON=-170.
      B1F=B1-D*EXP(EXPON)
      ALPHA(NSEG-1)=B1F-A*D/ALPHA(NSEG-2)
      W(NSEG-1)=A*CT(NSEG-1)-B2*CT(NSEG)+D*CT(NSEG+1)
      S(NSEG-1)=W(NSEG-1)+A*S(NSEG-2)/ALPHA(NSEG-2)
      GO TO 45
   43 CONTINUE
      ALPHA(NSEG-1)=B1-A*D/ALPHA(NSEG-2)
      W(NSEG-1)=A*CT(NSEG-1)-B2*CT(NSEG)+2.*D*CT(NSEG+1)
      S(NSEG-1)=W(NSEG-1)+A*S(NSEG-2)/ALPHA(NSEG-2)
      GO TO 45
   44 CONTINUE
      ALPHA(NSEG-1)=(B1-2.*D)+(D-A)*D/ALPHA(NSEG-2)
      W(NSEG-1)=A*CT(NSEG-1)-B2*CT(NSEG)+D*CT(NSEG+1)
      S(NSEG-1)=W(NSEG-1)-(D-A)*S(NSEG-2)/ALPHA(NSEG-2)
   45 CONTINUE
      CT1(NSEG)=S(NSEG-1)/ALPHA(NSEG-1)
      NFINAL=NSEG-2
      DO 46 II=1,NFINAL
      CT1(NSEG-II)=(S(NSEG-II-1)+D*CT1(NSEG-II+1))/ALPHA(NSEG-II-1)
   46 CONTINUE
C
C-----------------------------------------------------------------------
C     ***DETERMINE IF CURVE IS TO BE EXTENDED UPSTREAM AND
C            APPLY THE APPROPRIATE BOUNDARY CONDITIONS***
C-----------------------------------------------------------------------
      IF(MSU.EQ.1.AND.LTYPE.EQ.1) GO TO 55
      IF(LTYPE.NE.2) GO TO 53
      XUP=UPPER-U*TBEGIN
```

```
      EXPON= (+2.*(-XUP  +DELX) * DELX-(DELX**2.) - 2.*DELX*U*T) /
     $   (4.*E*T)
      CT1(1)=CT1(2)*EXP(EXPON)
      GO TO 55
   53 CONTINUE
      CT1(1)=2.*CT1(2)-CT1(3)
   55 CONTINUE
C
C-----------------------------------------------------------------------
C     ***DETERMINE IF CURVE IS TO BE EXTENDED DOWNSTREAM AND
C         APPLY THE APPROPRIATE BOUNDARY CONDITIONS***
C-----------------------------------------------------------------------
      IF(MSL.FQ.N.AND.LTYPE.EQ.1) GO TO 59
      IF(LTYPE.NE.2) GO TO 57
      XLOW=LOWER-U*TBEGIN
      EXPON= (-2.*((N-2)*DELX-XLOW ) * DELX- (DELX**2.) + 2.*DELX*U*T)/
     $   (4.*E*T)
      CT1(N)=CT1(N-1)*EXP(EXPCN)
      GO TO 59
   57 CONTINUE
      CT1(N)=2.*CT1(N-1)-CT1(N-2)
   59 CONTINUE
C
C-----------------------------------------------------------------------
C        ***APPLY APPROPRIATE LOADING CONDITIONS***
C-----------------------------------------------------------------------
      IF(LTYPE.EQ.1) GO TO 6C
      IF(LTYPE.EQ.3) CALL CNTINU(E,T,MASS,CT1(MSU))
      GO TO 70
   60 CONTINUE
```

```
      CT1(MSU)=UMAX
      CT1(MSL)=LMAX
   7C CONTINUE
      DO 73 L=1,N
      IF(CT1(L).LT.0.001) CT1(L)=0.0
      CT(L)=CT1(L)
   73 CONTINUE
      IF(COUNT.GE.NDTPR) GO TC 75
      GO TO 200
C
C-----------------------------------------------------------------------
C     ***PRINT CONCENTRATIONS FOR EACH DESIRED TIME INCREMENT***
C-----------------------------------------------------------------------
   75 WRITE(6,80) T,(CT1(M),M=1,N,NSKIP)
   80 FORMAT(/, F8.3, F10.4, 10F10.4, /, 1(18X, 10F10.4))
      COUNT=0
  200 CONTINUE
 1000 CONTINUE
C
C-----------------------------------------------------------------------
C     ***DETERMINE THE NUMBER OF CALCULATIONS MADE BY THE PROGRAM
C-----------------------------------------------------------------------
      NCALC=3*NS + 10*NSEG +  2*N/NSKIP + NOMORE*(10*NSEG+46)
      INDEX=N*NOMORE
      PRINT 205, N,NOMORE,NCALC, INDEX
  205 FORMAT(/////, 1X, 'N=', I3, 5X, 'NCMORE=', I5, 5X, 'NCALC=', I7,
     $ 5X, 'INDEX=', I6,////)
      STOP
      END
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CNTINU(EX,TIME,XMASS,CONCIN)
C
C             THIS SUBROUTINE SHOULD BE WRITTEN ACCORDING TO THE
C        PARTICULAR LOADING RATE OF THE OUTFALL BEING STUDIED.
C        THE SUBROUTINE WRITTEN BELOW IS VALID ONLY IN THE CASE OF
C        ONE OUTFALL AND WHERE NO VELOCITY IS PRESENT.
C
C          THIS SUBROUTINE IS USED ONLY IF LTYPE EQUALS 3.
C
C
C-----------------------------------------------------------------------
      RAD=3.14159*EX
      CONCIN=(XMASS*TIME**.5)/SQRT(RAD)
      RETURN
      END
```

Input Data for IDEAL-II

Each line represents a new card unless single spaced.

15, 0.002, 0.1, 1., .23, 5.0, 2, 2, 1, 1, 1, 0.0, 0.022

4

1000.

15

1000.

COMPUTER PROGRAM FOR

MASSTRANS-I

Object Code	=	54,728 bytes
Array Area	=	40,196 bytes
Total	=	94,924 bytes

```
C
C
C
C
C
C     ***************************************************
C     *                                                 *
C     *                 MASSTRANS-I                     *
C     *                                                 *
C     *         ESTUARY MASS TRANSPORT MODEL            *
C     *                                                 *
C     * TWO DIMENSIONAL NUMERICAL ANALYSIS - EXPLICIT METHOD *
C     *                                                 *
C     ******-------------------------------------*******
C          *                                       *
C          *  LOGIC AND PROGRAMMING BY             *
C          *                                       *
C          *       P. JONATHAN YOUNG               *
C          *   C/O ENVIRONMENTAL ENGINEERING DIVISION *
C          *       CIVIL ENGINEERING DEPARTMENT    *
C          *       TEXAS A&M UNIVERSITY            *
C          *       COLLEGE STATION, TEXAS 77840    *
C          *                                       *
C          *****************************************
C
```

```
C
C           THIS COMPUTER PROGRAM WAS DESIGNED PRIMARILY TO ANALYZE THE
C       MASS TRANSPORT OF DISSCLVED MATERIALS IN A PARTIALLY STRATIFIED
C       ESTUARY.  THESE MATERIALS MAY BE CONSERVATIVE OR NON-CONSERVATIVE.
C       AN EXPLICIT FINITE DIFFERENCE APPRCXIMATION TO THE ESTUARY MASS
C       TRANSPORT EQUATION FORMS THE BASIS OF THE COMPUTATIONS.
C
C           THIS PROGRAM WAS DEVELOPED TO ANALYZE ESTUARIES WHOSE
C       CHARACTERISTICS DO NOT VARY SIGNIFICANTLY WITH WIDTH.  THE WIDTH
C       OF THE ESTUARY MAY BE VARIED THROUGHOUT BUT THE CONCENTRA-
C       TIONS OF DISSOLVED MATERIALS AT EACH CROSS SECTION ARE CONSIDERED
C       TO BE UNCHANGING IN THE LATERAL CIRECTION.  ALL PHYSICAL AND
C       HYDRODYNAMIC CHARACTERISTICS MAY VARY WITH TIME IN THE LONGITU-
C       DINAL AND VERTICAL DIRECTIONS.  THE PROGRAM CAN BE APPLIED WITH
C       EQUAL EASE TO THE TWO HCRIZONTAL DIRECTIONS, ALLOWING DEPTH TO
C       VARY RATHER THAN WIDTH.
C
C           CONCENTRATION PROFILES CAN BE CALCULATED BY THIS PROGRAM FOR
C       CONTINUOUS OR INSTANTANEOUS RELEASES.  INPUT DATA MAY INCLUDE GRID
C       DIMENSIONS, DISTANCE INCREMENTS, TIME INCREMENTS, WIDTHS, LOADING
C       PARAMETERS, VELOCITIES, DISPERSION COEFFICIENTS, DECAY RATES,
C       BENTHAL DEMANDS, AND OTHER SOURCE OR SINK TERMS.
C
C           THE TIME INCREMENTS CAN BE INCREASED OR DECREASED AT ANY TIME
C       DURING THE CALCULATION OF THE CONCENTRATION PROFILE.  THE DIS-
C       TANCES BETWEEN GRID POINTS CAN BE INCREASED AT ANY TIME, AND A
C       ROUTINE WITHIN THE PROGRAM WILL CHOOSE THE APPROPRIATE VALUES FROM
C       THE PREVIOUSLY CALCULATED PROFILE AND WILL PLACE THESE VALUES IN
C       THE DESIRED LOCATIONS IN THE NEW GRID SYSTEM.  THE NUMBER OF GRID
C       POINTS MAY BE INCREASED OR DECREASED AT ANY TIME.  LIKEWISE, AT
C       ANY TIME DURING THE CALCULATION, A NEW SET OF DATA FOR PHYSICAL
C       AND HYDRODYNAMIC CONDITIONS MAY BE READ INTO THE PROGRAM.
```

```
C          A USER OF THIS PROGRAM MUST BE FAMILIAR WITH THE LIMITATIONS
C     ON ACCURACY AND STABILITY INHERENT IN THE TYPE OF NUMERICAL PROCE-
C     DURE USED IN THESE CALCULATIONS.  A SUBROUTINE WITHIN THE PROGRAM
C     PRINTS OUT THE PROPER INCREMENTS TO INSURE STABILITY AND TERMI-
C     NATES THE PROGRAM IF THIS CRITERIA IS VIOLATED BY THE INPUT PARA-
C     METERS.  ANOTHER SUBROUTINE EXTRAPOLATES CONCENTRATIONS AT THE
C     BOUNDARIES--SEVERAL METHODS CAN BE USED FOR THESE EXTRAPOLATIONS
C     DEPENDING ON THE TYPE OF PROFILE BEING ANALYZED AND THE CHOICE OF
C     THE USER.  A SUBROUTINE IS ALSO INCLUDED WHICH PRINTS OUT ERROR
C     MESSAGES AND TERMINATES THE PROGRAM IF CERTAIN INCONSISTENCIES
C     OCCUR IN THE INPUT DATA.
C
C          THIS PROGRAM WAS DEVELOPED PRIMARILY TO ANALYZE PARTIALLY
C     STRATIFIED ESTUARIES WHICH HAVE BEEN DREDGED OUT TO A FAIRLY
C     CONSTANT DEPTH AT THE CENTERLINE OF THE CHANNEL--THESE ESTUARIES
C     ARE COMMON IN THE GULF COAST REGION.  APPLICATION OF THIS PROGRAM
C     TO PARTIALLY STRATIFIED ESTUARIES WITH VARIABLE DEPTHS WOULD
C     REQUIRE MODERATE REVISIONS TO THE PROGRAM AND WOULD MAKE THE
C     PROGRAM ESTUARY-DEPENDENT.
C
C          THIS COMPUTER PROGRAM CAN ALSO BE APPLIED TO ESTUARIES WHICH
C     ARE WELL-MIXED IN THE VERTICAL DIRECTION.  THIS OPTION ALLOWS FOR
C     VARYING WIDTH OR VARYING DEPTH AND USES MOST OF THE ROUTINES
C     AVAILABLE TO THE TWO-DIMENSIONAL ANALYSIS.
C
C          QUESTIONS REGARDING THIS PROGRAM MAY BE REFERRED TO
C     JONATHAN YOUNG AT HYDROSCIENCE, INC, 363 OLD HOOK ROAD,
C     WESTWOOD, NEW JERSEY  07675   PHONE 201 / 666-2600
C
C-----------------------------------------------------------------------
```

```
C-----------------------------------------------------------------------
C          ***INPUT VARIABLES***
C
C        SET            = DATA SET NUMBER
C                         --CAN BE EQUAL TO 1 IF NO NEW GRID DIMENSIONS
C                           ARE TO BE READ IN--MUST BE GREATER THAN 1 IF
C                           GRID DIMENSIONS ARE TO BE CHANGED
C        **************NOTE--CPTIONAL GRID ADJUSTMENT PARAMETERS ARE
C                           INSERTED HERE.  THESE ARE EXPLAINED BELOW
C        NX             = NUMBER OF GRID POINTS IN THE X DIRECTION
C        NZ             = NUMBER OF GRID POINTS IN THE Z DIRECTION
C        DELMX          = X INCREMENT ( MILES )
C        DELTAZ         = Z INCREMENT ( FEET )
C        DELT1          = FIRST TIME INCREMENT ( DAYS )
C        ITER1          = NUMBER OF ITERATIONS USING DELT1
C        DELT2          = SECOND TIME INCREMENT ( DAYS )
C        ITER2          = NUMBER OF ITERATIONS USING DELT2
C        NC             = NUMBER OF GRID POINTS IN THE X DIRECTION WHERE
C                         CONCENTRATIONS ARE ADDED WITH THE DATA SET
C        NS             = NUMBER OF POINT SOURCES
C        XSTART         = MILEAGE AT UPPERMOST GRID POINT
C        TSTART         = TIME TO BE ADDED TO THE TOTAL TIME AT THE FIRST
C                         ITERATION OF THE DATA SET BEING PROCESSED( DAYS )
C        XPRINT         = PRINTING INTERVAL FOR X DIRECTION ( INTEGER )
C        TPRINT         = PRINTING INTERVAL FOR TIME ( INTEGER )
C        NPPAGE         = NUMBER OF GRIDS PRINTED PER PAGE OF OUTPUT
```

```
C         LTYPE        = TYPE CF LOADING ( INSTANTANEOUS OR CONTINUOUS )
C                          1 MEANS CONSTANT CONCENTRATION AT OUTFALL
C                            (THIS CAN BE USED FOR STEADY-STATE PROFILES)
C                          2 MEANS AN INSTANTANEOUS RELEASE
C                          3 MEANS CONTINUOUS LOADING
C         OPTION       = OPTION FOR BOUNDARY CONDITIONS
C                          1 MEANS NO TRANSFER ACROSS Z BOUNDARIES AND
C                            EXPONENTIAL EXTRAPOLATION IN X DIRECTION
C                          2 MEANS NO TRANSFER ACROSS Z BOUNDARIES AND
C                            CONSTANT SLOPE EXTRAPOLATION IN X DIRECTION
C                          3 MEANS A SYMMETRICAL DISTRIBUTION WITH
C                            EXPONENTIAL EXTRAPOLATION
C                          4 MEANS A CONSTANT SLOPE EXTRAPOLATION IN
C                            BOTH DIRECTIONS
C                          5 MEANS A SPECIAL CASE DEFINED BY THE USER
C                          6 MEANS NO TRANSFER ACROSS Z BOUNDARIES AND
C                            EXTRAPOLATION IN X DIRECTION BY INVERTED
C                            DIFFERENCES (ACFI)
C                          7 MEANS EXTRAPOLATION IN BOTH DIRECTIONS BY
C                            INVERTED DIFFERENCES (ACFI)
C         VX(X,Z)      = HORIZONTAL VELOCITY ( FT/SEC )
C         EX(X,Z)      = HORIZONTAL DISPERSION ( FT**2/SEC )
C         W(X)         = WIDTH ( FEET ) FOR THE TWO DIMENSIONAL CASE OR
C                        CROSS SECTIONAL AREA (FT**2) FOR THE ONE
C                        DIMENSIONAL CASE
C         KDAYS(X)     = DECAY RATE ( PER DAY )
C         DEMAND(X,Z)  = OTHER SOURCES AND SINKS (CONCENTRATION UNITS)
C         INSECT(I)    = SECTIONS INTO WHICH ARE ADDED LOADS
C                        OR CONCENTRATIONS
C         CONCIN(I,J)  = INITIAL CONCENTRATIONS ASSOCIATED WITH INSECT(I)
C
C         ****************NOTE--INITIAL CONCENTRATIONS MAY ALSO BE INPUT
C                          THROUGH THE BLOCK DATA SUBPROGRAM
```

```
C          **************NOTE--VALUES FOR VZ AND EZ ARE TO BE INCLUDED
C                              WITH THE DATA ONLY IF NZ IS GREATER THAN 1
C          VZ(X,Z)      = VERTICAL VELOCITIES ( FT/SEC )
C          EZ(X,Z)      = VERTICAL DISPERSION ( FT**2/SEC )
C
C          FMT1(I)      = FORMAT FOR HEADING FCR COMPUTED PROFILES
C          FMT2(I)      = FORMAT FOR DEPTH NOTATION FOR PROFILES
C          FMT3(I)      = FORMAT FOR OUTPUT CF COMPUTED PROFILES
C
C
C-----------------------------------------------------------------------
C
C            *** GRID ADJUSTMENT PARAMETERS ***
C
C
C          NXSKIP       = NUMBER OF X INTERVALS SKIPPED IN OLD GRID
C          NZSKIP       = NUMBER OF Z INTERVALS SKIPPED IN OLD GRID
C          NXTAKE       = LOWEST GRID NUMBER IN X DIRECTION IN THE OLD GRID
C                         FROM WHICH A CONCENTRATION IS TAKEN
C          NZTAKE       = LOWEST GRID NUMBER IN Z DIRECTION IN THE OLD GRID
C                         FROM WHICH A CONCENTRATION IS TAKEN
C          NXPUT        = LOWEST GRID NUMBER IN THE X DIRECTION IN THE NEW
C                         GRID INTO WHICH A CONCENTRATION IS PLACED
C          NZPUT        = LOWEST GRID NUMBER IN THE Z DIRECTION IN THE NEW
C                         GRID INTO WHICH A CONCENTRATION IS PLACED
C-----------------------------------------------------------------------
```

```
C-----------------------------------------------------------------------
C
C              ************************
C              ***** MAIN PROGRAM *****
C              ************************
C
C          THE MAIN PROGRAM CALLS FOR THE DATA TO BE READ IN,THEN CALLS
C     FOR THE STABILITY CONDITIONS TO BE EVALUATED,AND NEXT CALLS FOR
C     THE PRINTING OF A SUMMARY OF INPUT DATA.  THE NUMERICAL ANALYSIS
C     SUBROUTINE IS THEN CALLED FOR EITHER THE ONE-DIMENSIONAL OR TWO-
C     DIMENSIONAL CASE.  IF THE TIME INCREMENT IS TO BE CHANGED DURING
C     THIS PART OF THE PROGRAM,THE APPROPRIATE PARAMETERS ARE ADJUSTED.
C     STABILITY IS AGAIN CHECKED AND THE NUMERICAL ANALYSIS IS
C     CONTINUED.  NEXT,A NEW DATA SET MAY BE READ TO CONTINUE THE
C     ANALYSIS.  ANY NUMBER OF DATA SETS MAY BE PROCESSED.  WHEN NO MORE
C     DATA SETS ARE AVAILABLE,THE PROGRAM IS TERMINATED.
C
C          THE INPUT AND OUTPUT OF DATA CAN BE CONTROLLED IN THREE
C     WAYS--THE INPUT DATA ITSELF, THE BLOCK DATA SUBROUTINE, AND THE
C     SUBROUTINE PRINT2.  NO OTHER PART OF THE PROGRAM SHOULD REQUIRE
C     CHANGING.
C
C          THE FORTRAN USED IN THIS PROGRAM USES SEVERAL OF THE OPTIONS
C     AVAILABLE IN WATFOR AS DESCRIBED IN THE FOLLOWING TEXT**FORTRAN IV
C     WITH WATFOR AND WATFIV**BY CRESS,DIRKSEN,&GRAHAM,PRENTICE-HALL,
C     INC.,1970. THE PROGRAM CAN BE RUN ON A COMPUTER WITH A WATFOR OR
C     WATFIV COMPILER.  THE SYNTAX HAS BEEN KEPT COMPATIBLE WITH
C     FORTRAN G EXCEPT FOR THE UNFORMATTED READ AND PRINT STATEMENTS.
C
C-----------------------------------------------------------------------
```

```
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
   10 CALL DATA(&1000)
      CALL STABLE
      CALL PRINT1
      CALL PRINT2
      CALL ERROR(&1000)
   25 IF(NZ.EQ.1) GO TO 310
      CALL TWODEX
      GO TO 315
  310 CALL ONEDEX
  315 IF(ITER2.EQ.0) GO TO 10
      ITER=ITER2
      ITER2=0
      TPRINT=TPRINT*(DELT1/DELT2+.00001)
      DELTAT=DELT2*86400.
      CALL STABLE
      IF(ITER.EQ.0) CALL ERROR(&1000)
      GO TO 25
 1000 CONTINUE
      STOP
      END
```

```
      C---------------------------------------------------------------------
25          BLOCK DATA
      C             **********************
      C             ***** BLOCK DATA *****
      C             **********************
      C                THIS ROUTINE INITIALIZES SELECTED VARIABLES IN THE
      C          COMMON BLOCKS.  THIS SUBPROGRAM MAY BE USED TO ESTABLISH
      C          INITIAL CONCENTRATION PROFILES IF THE USER DESIRES.
      C
      C---------------------------------------------------------------------
26          INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
27          REAL KD,KDMAX,KDMIN,KDAYS,LOWER
28          COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
29          COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
      C
      C---------------------------------------------------------------------
      C           ESTABLISH INITIAL CONCENTRATION IF DESIRED
      C---------------------------------------------------------------------
30          DATA C/2142*0.0/
31          DATA VX,VZ,EX,EZ,CONCIN,DEMAND/6426*0.0/
32          DATA W,KDAYS,KD/153*0.0/
33          DATA EXMAX,EZMAX,VXMAX,VZMAX,WMAX,KDMAX,DEMAX,TIME/8*0.0/
34          DATA EXMIN,EZMIN,VXMIN,VZMIN,WMIN,KDMIN,DEMIN/7*1000000./
35          DATA INSECT/51*0/
36          DATA IPAGE/1/
37          DATA T/1/,TNEXT/2/
38          END
```

```
C-----------------------------------------------------------------------
      SUBROUTINE DATA(*)
C
C              ***************************
C              ***** SUBROUTINE DATA *****
C              ***************************
C
C           THIS SUBROUTINE READS IN THE APPROPRIATE DATA SET AND ADJUSTS
C      THE GRID SIZE IF NECESSARY.  THE LOCATIONS OF THE UPPERMOST AND
C      LOWERMOST PEAKS ARE DETERMINED, AND THE MINIMUM AND MAXIMUM VALUES
C      FOR THE INPUT PARAMETERS ARE CALCULATED.  ALL OF THE INPUT DATA IS
C      PRINTED OUT UNFORMATTED.  THE FORMAT IS READ IN FOR THE PRINTING
C      OUT OF THE CALCULATED CONCENTRATIONS.
C
C-----------------------------------------------------------------------
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
      READ(5,12,END=13) SET
   12 FORMAT(F10.0)
      GO TO 14
   13 CONTINUE
      RETURN1
   14 CONTINUE
      IF(SET.EQ.1.)  GO TO 400
```

```
C
C-----------------------------------------------------------------------
C           CHOOSE CONCENTRATIONS FROM THE OLD GRID TO PUT INTO THE
C           APPROPRIATE PLACES IN THE NEW GRID .  THE GRID CAN ONLY BE
C           CHANGED IF LTYPE EQUALS 2.  OTHERWISE, THE DATA CARD IS READ
C           BUT  IS NOT IMPLEMENTED.
C-----------------------------------------------------------------------
      READ ,NXSKIP,NZSKIP,NXTAKE,NZTAKE,NXPUT,NZPUT
      IF(LTYPE.NE.2) GO TO 400
      NXSTOP=(NX-1)/NXSKIP+1
      NZSTOP=(NZ-1)/NZSKIP+1
      IF(NZ.EQ.1) NZSTOP=1
      DO 350 I=1,NXSTOP
      X=NXPUT-1+I
      II=NXTAKE+NXSKIP*(I-1)
      DO 350 J=1,NZSTOP
      Z=NZPUT-1+J
      JJ=NZTAKE+NZSKIP*(J-1)
      C(X,Z,TNEXT)=C(II,JJ,T)
  350 CONTINUE
  400 CONTINUE
C
C
C-----------------------------------------------------------------------
C           READ AND PRINT INPUT DATA FOR HORIZONTAL DIRECTION
C-----------------------------------------------------------------------
```

```
      READ,NX,NZ
      READ,DELMX,DELTAZ
      READ,DELT1,ITER1
      READ,DELT2,ITER2
      READ,NC,NS
      READ,XSTART,TSTART,XPRINT,TPRINT,NPPAGE
      READ,LTYPE,OPTION
      PRINT 1001
 1001 FORMAT( 1H1,10X,'INPUT DATA',///,
     1   1X,'NX,NZ,    DELMX,DELTAZ,   DELT1,ITER1')
      PRINT,NX,NZ,DELMX,DELTAZ,DELT1,ITER1
      PRINT 1002
 1002 FORMAT(//,1X,'DELT2,ITER2,    NC,NS,    XSTART,TSTART')
      PRINT, DELT2,ITER2,NC,NS,XSTART,TSTART
      PRINT 1003
 1003 FORMAT(//,1X,'XPRINT,TPRINT,NPPAGE,    LTYPE,OPTION')
      PRINT, XPRINT,TPRINT,NPPAGE,LTYPE,OPTION
      IF(SET.EQ.1.) GO TO 500
      DO 375 Z=1,NZ
      DO 375 X=1,NX
      C(X,Z,T)=C(X,Z,TNEXT)
      C(X,Z,TNEXT)=0.0
  375 CONTINUE
  500 CONTINUE
      NXM1=NX-1
      NXM2=NX-2
      NZM1=NZ-1
      NZM2=NZ-2
      DELTAT=DELT1*86400.
      DELTAX=DELMX*5280.
```

```
      XSF=XSTART*5280.
      ITER=ITER1
      TIME=TIME+TSTART*86400.
      COUNT=(TSTART/DELTAT+0.01)
      COUNT=COUNT-(COUNT/TPRINT)*TPRINT
      IF(SET.EQ.2) RETURN
      READ,((VX(X,Z),X=1,NX),Z=1,NZ)
      PRINT 2
    2 FORMAT(//, 1X, '((VX(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((VX(X,Z),X=1,NX),Z=1,NZ)
      READ,((EX(X,Z),X=1,NX),Z=1,NZ)
      PRINT 3
    3 FORMAT(//, 1X, '((EX(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((EX(X,Z),X=1,NX),Z=1,NZ)
      READ,(W(X),X=1,NX)
      PRINT 4
    4 FORMAT(//, 1X, '(W(X),X=1,NX)')
      PRINT, (W(X),X=1,NX)
      READ,(KDAYS(X),X=1,NX)
      PRINT 5
    5 FORMAT(//, 1X, '(KDAYS(X),X=1,NX)')
      PRINT, (KDAYS(X),X=1,NX)
      READ,((DEMAND(X,Z),X=1,NX),Z=1,NZ)
      PRINT 6
    6 FORMAT(//, 1X, '((DEMAND(X,Z),X=1,NX),Z=1,NZ)')
      PRINT, ((DEMAND(X,Z),X=1,NX),Z=1,NZ)
      IF(NC.EQ.0) GO TO 24
      DO 10 I=1,NC
      READ,INSECT(I)
      PRINT 7
```

```
    7 FORMAT(//, 1X, 'INSECT(I)')
      PRINT,INSECT(I)
      READ, (CONCIN(I,J), J=1,NZ)
      PRINT 8
    8 FORMAT(//, 1X, '(CONCIN(I,J),J=1,NZ)')
      PRINT,(CONCIN(I,J),J=1,NZ)
   10 CONTINUE
      DO 15 I=1,NC
      DO 15 Z=1,NZ
      C(INSECT(I),Z,T)=CONCIN(I,Z) +C(INSECT(I),Z,T)
   15 CONTINUE
C
C-----------------------------------------------------------------------
C           DETERMINE LOCATIONS OF UPPFRMOST AND LOWERMOST
C           PEAK CONCENTRATIONS
C-----------------------------------------------------------------------
      AMAX1=0.0
      PEAKU=0.0
      AMAX2=0.0
      PEAKL=0.0
      UPPER=XSF
      LOWER=XSF
      MSU=1
      MSL=NX
      DO 17 I=1,NC
      DO 16 J=1,NZ
      IF(CONCIN(I,J).LT.PEAKU) GO TO 16
      PEAKU=CONCIN(I,J)
   16 CONTINUE
      IF(PEAKU.LE.AMAX1) GO TO 20
```

```
      AMAX1=PEAKU
      MSU=INSECT(I)
      UPPER=(MSU-1)*DELTAX +XSF
   17 CONTINUE
   20 CONTINUE
      DO 22 II=1,NC
      I=NC+1-II
      DO 21 J=1,NZ
      IF(CONCIN(I,J).LT.PEAKL) GO TO 21
      PEAKL=CONCIN(I,J)
   21 CONTINUE
      IF(PEAKL.LE.AMAX2) GO TO 24
      AMAX2=PEAKL
      MSL=INSECT(I)
      LOWER=(MSL-1)*DELTAX +XSF
   22 CONTINUE
   24 CONTINUE
C
C-----------------------------------------------------------------------
C            DETERMINE MAXIMUM AND MINIMUM VALUES FOR INPUT DATA
C-----------------------------------------------------------------------
      DO 25 X=1,NX
      KD(X)=KDAYS(X)/86400.
      IF(W(X).GT.WMAX)WMAX=W(X)
      IF(W(X).LT.WMIN)WMIN=W(X)
      IF(KD(X).GT.KDMAX)KDMAX=KD(X)
      IF(KD(X).LT.KDMIN)KDMIN=KD(X)
      DO 25 Z=1,NZ
      ABSVX=ABS(VX(X,Z))
      IF(ABSVX.GT.VXMAX)VXMAX=ABSVX
```

```
      IF(ABSVX.LT.VXMIN)VXMIN=ABSVX
      IF(EX(X,Z).GT.EXMAX)EXMAX=EX(X,Z)
      IF(EX(X,Z).LT.EXMIN)EXMIN=EX(X,Z)
      IF(DEMAND(X,Z).GT.DEMAX)DEMAX=DEMAND(X,Z)
      IF(DEMAND(X,Z).LT.DEMIN)DEMIN=DEMAND(X,Z)
   25 CONTINUE
      IF(NZ.EQ.1)GO TO 200
C
C-----------------------------------------------------------------------
C           READ AND PRINT DATA FOR VERTICAL DIRECTION
C-----------------------------------------------------------------------
      READ,((VZ(X,Z),X=1,NX),Z=1,NZ)
      PRINT 30
   30 FORMAT(//, 1X, '((VZ((X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((VZ(X,Z),X=1,NX),Z=1,NZ)
      READ,((EZ(X,Z),X=1,NX),Z=1,NZ)
      PRINT 32
   32 FORMAT(//, 1X, '((EZ(X,Z),X=1,NX),Z=1,NZ)')
      PRINT, ((EZ(X,Z),X=1,NX),Z=1,NZ)
C
C-----------------------------------------------------------------------
C           DETERMINE MAXIMUM AND MINIMUM VALUES FOR INPUT DATA
C-----------------------------------------------------------------------
      DO 120 X=1,NX
      DO 120 Z=1,NZ
      ABSVZ=ABS(VZ(X,Z))
      IF(ABSVZ.GT.VZMAX)VZMAX=ABSVZ
      IF(ABSVZ.LT.VZMIN)VZMIN=ABSVZ
      IF(EZ(X,Z).GT.EZMAX)EZMAX=EZ(X,Z)
      IF(EZ(X,Z).LT.EZMIN)EZMIN=EZ(X,Z)
  120 CONTINUE
  200 CONTINUE
```

```
C
C--------------------------------------------------------------------
C          READ AND PRINT FORMATS FOR OUTPUT
C--------------------------------------------------------------------
      READ 205, (FMT1(I), I=1,40)
      PRINT 202
  202 FORMAT(//,1X, 'FORMAT FOR OUTPUT')
      PRINT 205, (FMT1(I),I=1,40)
  205 FORMAT (20A4,/,20A4)
      READ 205,(FMT2(I),I=1,40)
      PRINT 205, (FMT2(I), I=1,40)
      READ 207,(FMT3(I), I=1,20)
      PRINT 207, (FMT3(I), I=1,20)
  207 FORMAT(20A4)
      PRINT 259
  259 FORMAT(1H1)
      RETURN
      END
```

```
    C-----------------------------------------------------------------------
213       SUBROUTINE STABLE
    C
    C            ******************************
    C            ***** SUBROUTINE STABLE *****
    C            ******************************
    C
    C          THE RELATIONSHIP BETWEEN DISTANCE INCREMENTS,TIME INCREMENTS,
    C     DISPERSION COEFFICIENTS, AND VELOCITIES IS NEEDED TO DETERMINE
    C     THE STABILITY OF THE EXPLICIT FINITE-DIFFERENCE PROCEDURE.  THIS
    C     SUBROUTINE CALCULATES THE MAXIMUM ALLOWABLE INCREMENTS FOR TIME
    C     AND DISTANCE.  THE PROGRAM IS TERMINATED IF THE INPUT PARAMETERS
    C     VIOLATE THIS CRITERIA.
    C
    C-----------------------------------------------------------------------
214       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
215       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
216       COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
         1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
         2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
217       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
         1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
         2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
         3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
         4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
         5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
218       STABLX = DELTAX
219       STABLZ = DELTAZ
220       STABLT = DELTAT
```

```
C
C-----------------------------------------------------------------------
C          CALCULATE STABILITY CRITERIA FOR ONE DIMENSIONAL CASE
C-----------------------------------------------------------------------
      IF(NZ.NE.1) GO TO 500
      IF(EXMIN.GT.0.0.AND.VXMAX.GT.0.0) STABLX=2.*EXMIN/VXMAX
      IF(STABLX.LT.DELTAX) ITER=0
      TERM=2.*EXMAX+DELTAX**2.*KDMAX
      IF(TERM.NE.0.0) STABLT=(DELTAX**2.)/TERM
      IF(STABLT.LT.DELTAT) ITER=0
      R1=DELTAT/(DELTAX**2.)
      R2=0.
      GO TO 800
  500 CONTINUE
C
C-----------------------------------------------------------------------
C          CALCULATE STABILITY CRITERIA FOR TWO DIMENSIONAL CASE
C-----------------------------------------------------------------------
      IF(EXMIN.GT.0.0.AND.VXMAX.GT.0.0) STABLX=2.*EXMIN/VXMAX
      IF(STABLX.LT.DELTAX) ITER=0
      IF(EZMIN.GT.0.0.AND.VZMAX.GT.0.0) STABLZ=2.*EZMIN/VZMAX
      IF(STABLZ.LT.DELTAZ) ITER=0
      TERM=2.*(EXMAX*DELTAZ**2.+EZMAX*DELTAX**2.)+KDMAX*(DELTAX**2)*
     $  (DELTAZ**2)
      IF(TERM.NE.0.0) STABLT=(DELTAZ**2.)*(DELTAX**2.)/TERM
      IF(STABLT.LT.DELTAT) ITER=0
      R1=DELTAT/(DELTAX**2.)
      R2=DELTAT/(DELTAZ**2.)
  800 CONTINUE
      RETURN
      END
```

```
    C-----------------------------------------------------------------
243       SUBROUTINE ONEDEX
    C
    C             *****************************
    C             ***** SUBROUTINE ONEDEX *****
    C             *****************************
    C
    C          THIS SUBROUTINE CALCULATES THE ONE DIMENSIONAL CONCENTRATION
    C     PROFILE FOR THE ASSIGNED GRID AND THE ASSIGNED NUMBER OF ITERA-
    C     TIONS.  A FINITE-DIFFERENCE,EXPLICIT SCHEME IS USED TO SOLVE THE
    C     PARTIAL DIFFERENTIAL EQUATION FOR MASS TRANSPORT.
    C
    C-----------------------------------------------------------------
244       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
245       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
246       COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
         1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
         2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
247       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
         1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
         2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
         3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
         4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
         5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
```

```
      Z=1
      NXM1=NX-1
C
C-----------------------------------------------------------------------
C           CALCULATE TERMS WHICH ARE USED REPEATEDLY
C-----------------------------------------------------------------------
      TWOXSQ=2.*DELTAX**2.
      TWOX=2.*DELTAX
      TWOZSQ=2.*DELTAZ**2.
      TWOZ=2.*DELTAZ
C
C-----------------------------------------------------------------------
C           CALCULATE NEW CONCENTRATION AT EACH POINT
C-----------------------------------------------------------------------
      DO 300 IT=1,ITER
      TIME=TIME+DELTAT
      DO 200 X=2,NXM1
      TERM1A=W(X)*VX(X,Z)/TWOX
      TERM1B=(W(X)*EX(X,Z)+W(X-1)*EX(X-1,Z))/TWOXSQ
      TERM2B = (W(X+1)*EX(X+1,Z) + W(X)*EX(X,Z)) / TWOXSQ
     $         +(W(X)*EX(X,Z) + W(X-1)*EX(X-1,Z)) / TWOXSQ
      TERM3A=(-W(X)*VX(X,Z))/TWOX
      TERM3B=(W(X+1)*EX(X+1,Z)+W(X)*EX(X,Z))/TWOXSQ
      TERM1=C(X-1,Z,T)*(TERM1A+TERM1B)
      TERM2=-C(X,Z,T)*(TERM2B+KD(X)*W(X))
      TERM3=C(X+1,Z,T)*(TERM3A+TERM3B)
      C(X,Z,TNEXT)=C(X,Z,T)+(DELTAT/W(X))*(TERM1+TERM2+TERM3)
     $  -DEMAND(X,Z)
      IF(C(X,Z,TNEXT).LT.0.0) C(X,Z,TNEXT) = 0.0
  200 CONTINUE
```

```
C
C-----------------------------------------------------------------------
C           CALCULATE BOUNDARY VALUES
C-----------------------------------------------------------------------
      CALL BOUND
      Z=1
      DO 250 X=1,NX
      C(X,Z,T)=C(X,Z,TNEXT)
      C(X,Z,TNEXT)=0.0
  250 CONTINUE
      COUNT=COUNT+1
      IF(COUNT.GE.TPRINT) GO TO 292
      GO TO 300
C
C-----------------------------------------------------------------------
C           PRINT CONCENTRATIONS AT APPROPRIATE INTERVALS
C-----------------------------------------------------------------------
  292 CONTINUE
      COUNT=0
      IPAGE=IPAGE+1
      IF(IPAGE.GE.NPPAGE) GO TO 295
      GO TO 297
  295 PRINT 296
  296 FORMAT(1H1)
      IPAGE = 0
  297 CALL PRINT2
  300 CONTINUE
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
289         SUBROUTINE TWODEX
      C
      C              ****************************
      C              ***** SUBROUTINE TWODEX *****
      C              ****************************
      C
      C          THIS SUBROUTINE CALCULATES THE TWO-DIMENSIONAL CONCENTRATION
      C     PROFILE FOR THE ASSIGNED GRID AND THE ASSIGNED NUMBER OF ITERA-
      C     TIONS.  A FINITE-DIFFERENCE,EXPLICIT SCHEME IS USED TO SOLVE THE
      C     PARTIAL DIFFERENTIAL EQUATION FOR MASS TRANSPORT.
      C
      C-----------------------------------------------------------------------
290         INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
291         REAL KD,KDMAX,KDMIN,KDAYS,LOWER
292         COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
293         COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
294         NXM1=NX-1
295         NZM1=NZ-1
296         NZ1=2
```

```
C
C-----------------------------------------------------------------------
C           CALCULATE TERMS WHICH ARE USED REPEATEDLY
C-----------------------------------------------------------------------
      TWOXSQ=2.*DELTAX**2.
      TWOX=2.*DELTAX
      TWOZSQ=2.*DELTAZ**2.
      TWOZ=2.*DELTAZ
C
C-----------------------------------------------------------------------
C           CALCULATE NEW CONCENTRATION AT EACH POINT
C-----------------------------------------------------------------------
      DO 300 IT=1,ITER
      TIME=TIME+DELTAT
      DO 200 X=2,NXM1
      DO 100 Z=NZ1,NZM1
      TERM1A=W(X)*VX(X,Z)/TWOX
      TERM1B=(W(X)*EX(X,Z)+W(X-1)*EX(X-1,Z))/TWOXSQ
      TERM2B = (W(X+1)*EX(X+1,Z) + W(X)*EX(X,Z)) / TWOXSQ
     $        +(W(X)*EX(X,Z) + W(X-1)*EX(X-1,Z)) / TWOXSQ
      TERM3A=(-W(X)*VX(X,Z))/TWOX
      TERM3B=(W(X+1)*EX(X+1,Z)+W(X)*EX(X,Z))/TWOXSQ
      TERM4A=W(X)*VZ(X,Z)/TWCZ
      TERM4B=(W(X)*EZ(X,Z)+W(X)*EZ(X,Z-1))/TWOZSQ
      TERM5A=(-W(X)*VZ(X,Z))/TWOZ
      TERM5B=(W(X)*EZ(X,Z+1)+W(X)*EZ(X,Z))/TWOZSQ
      TERM6B = (W(X) * EZ(X,Z+1) + W(X)*EZ(X,Z))/ TWOZSQ
     $        +(W(X)*EZ(X,Z)+ W(X)*EZ(X,Z-1)) / TWOZSQ
```

```
      TERM2=-C(X,Z,T)*(TERM2B+TERM6B+KD(X)*W(X))
      TERM1=C(X-1,Z,T)*(TERM1A+TERM1B)
      TERM3=C(X+1,Z,T)*(TERM3A+TERM3B)
      TERM4=C(X,Z-1,T)*(TERM4A+TERM4B)
      TERM5=C(X,Z+1,T)*(TERM5A+TERM5B)
      C(X,Z,TNEXT)=C(X,Z,T)+(DELTAT/W(X))*(TERM1+TERM2+TERM3+TERM4+TERM5
     $  )-DEMAND(X,Z)
      IF(C(X,Z,TNEXT).LT.0.0) C(X,Z,TNEXT) = 0.0
  100 CONTINUE
  200 CONTINUE
C
C-----------------------------------------------------------------------
C           CALCULATE BOUNDARY VALUES
C-----------------------------------------------------------------------
      CALL BOUND
      DO 250 Z=1,NZ
      DO 250 X=1,NX
      C(X,Z,T)=C(X,Z,TNEXT)
      C(X,Z,TNEXT)=0.0
  250 CONTINUE
```

```
      COUNT=COUNT+1
      IF(COUNT.GE.TPRINT) GO TO 292
      GO TO 300
C
C-----------------------------------------------------------------------
C               PRINT CONCENTRATIONS AT APPROPRIATE INTERVALS
C-----------------------------------------------------------------------
  292 CONTINUE
      COUNT=0
      IPAGE=IPAGE+1
      IF(IPAGE.GE.NPPAGE) GO TO 295
      GO TO 297
  295 PRINT 296
  296 FORMAT(1H1)
      IPAGE = 0
  297 CALL PRINT2
  300 CONTINUE
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
345         SUBROUTINE BOUND
      C
      C              ****************************
      C              ***** SUBROUTINE BOUND *****
      C              ****************************
      C
      C           SUBROUTINE BOUND PROVIDES FOR EXTRAPOLATION OF CONCENTRATIONS
      C       AT THE BOUNDARIES.  THE TYPE OF EXTRAPOLATION DEPENDS ON THE VALUE
      C       OF THE VARIABLE OPTION.
      C
      C-----------------------------------------------------------------------
346         INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
347         REAL KD,KDMAX,KDMIN,KDAYS,LOWER
348         COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
349         COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
350         IF(NZ.EQ.1) NZM1=1
351         ZUP=UPPER
352         ZLOW=LOWER
353         ZSF=XSF
```

```
      GO TO (1001,1002,1003,1004,10C5,1006,1006), OPTION
C
C-----------------------------------------------------------------
C         EXPONENTIAL EXTRAPOLATICN
C-----------------------------------------------------------------
 1003 CONTINUE
      IF(NZ.EQ.1) GO TC 1001
      DO 140 X=2,NXM1
      IF(C(X,2,T+1).LE.0.0) GC TO 140
      VZ1=VZ(X,1)
      ZZ=-ZUP+DELTAZ+ZSF+VZ1*TSTART*86400.
      EZ1=EZ(X,1)
      C2=4.*EZ1*TIME
      EXPON=( 2.*ZZ*DELTAZ-(CELTAZ*DELTAZ)-2.*DELTAZ*VZ1*TIME)/C2
      C(X,1,T+1)=C(X,2,T+1)*EXP(EXPON)
  140 CONTINUE
      DO 190 X=2,NXM1
      IF(C(X,NZM1,T+1).LE.0.0) GO TO 190
      VZNZ=VZ(X,NZ)
      ZZ=(NZ-2)*DELTAZ-ZLOW+ZSF+VZNZ*TSTART*86400.
      EZNZ=EZ(X,NZ)
      C2=4.*EZNZ*TIME
      EXPON=(-2.*ZZ*DELTAZ-(DELTAZ*DELTAZ)+2.*DELTAZ*VZNZ*TIME)/C2
      C(X,NZ,T+1)=C(X,NZM1,T+1)*EXP(EXPON)
  190 CONTINUE
 1001 CONTINUE
      IF(LTYPE.EQ.1.AND.MSU.EQ.1) GO TO 350
      DO 340 Z=1,NZ
      IF(C(2,Z,T+1).LE.0.0) GO TO 340
      VX1=VX(1,Z)
      XX=-UPPER+DELTAX+XSF+VX1*TSTART*86400.
      EX1=EX(1,Z)
```

```
      C2=4.*EX1*TIME
      EXPON=( 2.*XX*DELTAX-(DELTAX*DELTAX)-2.*DELTAX*VX1*TIME)/C2
      C(1,Z,T+1)=C(2,Z,T+1)*EXP(EXPON)
  340 CONTINUE
  350 CONTINUE
      IF(LTYPE.EQ.1.AND.MSL.EQ.NZ) GO TO 400
      DO 390 Z=1,NZ
      IF(C(NX-1,Z,T+1).LE.0.0) GO TO 390
      VXNX=VX(NX,Z)
      XX=(NX-2)*DELTAX-LOWER+XSF+VXNX*TSTART*86400.
      EXNX=EX(NX,Z)
      C2=4.*EXNX*TIME
      EXPON=(-2.*XX*DELTAX-(DELTAX*DELTAX)+2.*DELTAX*VXNX*TIME)/C2
      C(NX,Z,T+1)=C(NX-1,Z,T+1)*EXP(EXPON)
  390 CONTINUE
  400 CONTINUE
      GO TO 2000
C
C---------------------------------------------------------------------
C         CONSTANT SLOPE EXTRAPOLATION
C---------------------------------------------------------------------
 1004 CONTINUE
      IF(NZ.EQ.1) GO TO 1002
      DO 540 X=2,NXM1
      C(X,1,T+1)=2.*C(X,2,T+1)-C(X,3,T+1)
      IF(C(X,1,T+1).LT.0.0) C(X,1,T+1)=0.0
      C(X,NZ,T+1)=2.*C(X,NZM1,T+1)-C(X,NZM2,T+1)
      IF(C(X,NZ,T+1).LT.0.0) C(X,NZ,T+1)=0.0
  540 CONTINUE
```

```
 1002 CONTINUE
      DO 590 Z=1,NZ
      C(1,Z,T+1)=2.*C(2,Z,T+1)-C(3,Z,T+1)
      IF(C(1,Z,T+1).LT.0.0) C(1,Z,T+1)=0.0
      C(NX,Z,T+1)=2.*C(NXM1,Z,T+1)-C(NXM2,Z,T+1)
      IF(C(NX,Z,T+1).LT.0.0) C(NX,Z,T+1)=0.0
  590 CONTINUE
      GO TO 2000
 1005 CONTINUE
C
C-----------------------------------------------------------------------
C        THIS OPTION ALLOWS THE PROGRAM USER TO SUBSTITUTE HIS OWN
C        EXTRAPOLATION ROUTINE
C-----------------------------------------------------------------------
      ZZ=-40.
      EZ1=0.025
      VZ1=0.
      C2=4.*EZ1*TIME
      EXPON=(+2.*ZZ*DELTAZ-(DELTAZ**2.)-2*DELTAZ*VZ1*TIME)/C2
      DO 240 X=1,NX
      C(X,1,T+1)=C(X,2,T+1)*EXP(EXPON)
      C(X,NZ,T+1)=C(X,1,T+1)
  240 CONTINUE
      GO TO 1001
C
C-----------------------------------------------------------------------
C           INVERTED DIFFERENCES EXTRAPOLATION (ACFI)
C-----------------------------------------------------------------------
 1006 CALL EXTRAP
      GO TO 2000
 2000 CONTINUE
```

```
      IF(NZ.EQ.1) GO TO 2005
      IF(OPTION.EQ.3.OR.OPTION.EQ.4.OR.OPTION.EQ.5)  GO TO 2005
      DO 700 X=1,NX
      C(X,1,T+1)=C(X,2,T+1)
      C(X,NZ,T+1)=C(X,NZM1,T+1)
  700 CONTINUE
 2005 CONTINUE
C
C-----------------------------------------------------------------------
C        IF LTYPE EQUALS 1, SET CONCENTRATIONS AT SOURCE POINTS.
C        IF LTYPE EQUALS 3, THE USER CAN INCLUDE A SPECIAL SUBROUTINE
C        OR SET OF CALCULATIONS IN THIS PART OF THE PROGRAM.
C-----------------------------------------------------------------------
      IF(LTYPE.EQ.1.AND.NS.LE.2) GO TO 275
      IF(LTYPE.EQ.1.AND.NS.GT.2) GO TO 281
      GO TO 290
  275 CONTINUE
      DO 280 Z=1,NZ
      C(MSU,Z,T+1)=C(MSU,Z,T)
      C(MSL,Z,T+1)=C(MSL,Z,T)
  280 CONTINUE
      GO TO 290
  281 CONTINUE
      DO 285 I=1,NC
      DO 285 Z=1,NZ
      C(INSECT(I),Z,T+1)=CONCIN(I,Z)
  285 CONTINUE
  290 CONTINUE
      RETURN
      END
```

```
C-----------------------------------------------------------------------
      SUBROUTINE EXTRAP
C
C             ***************************
C             ***** SUBROUTINE EXTRAP ****
C             ***************************
C
C          THIS SUBROUTINE EXTRAPOLATES THE CONCENTRATION PROFILE BY
C     USING A CONTINUED FRACTIONS AND INVERTED DIFFERENCES SCHEME. IT IS
C     A MODIFIED VERSION OF THE IBM SCIENTIFIC SUBPROGRAM ACFI.
C
C          IF THE EXTRAPOLATED CONCENTRATION IS GREATER THAN THE
C     ADJACENT CONCENTRATION OR IF IT IS NEGATIVE, THEN THE BOUNDARY
C     CONCENTRATION IS EXTRAPOLATED ACCORDING TO THE PROPORTION
C     BETWEEN THE CONCENTRATICNS AT THE TWO ADJACENT INTERNAL POINTS
C     (I.E.  C1/C2 = C2/C3 ).
C
C-----------------------------------------------------------------------
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
      DIMENSION  ARG(10), VALY(10)
```

```
      NZ1=2
      IF(NZ.EQ.1) NZ1=1
      IF(NZ.EQ.1)NZM1=1
      IF(MSU.EQ.1.AND.LTYPE.EQ.1) GO TO 228
C
C-----------------------------------------------------------------------
C           EXTRAPOLATE THE PROFILE IN THE X DIRECTION
C-----------------------------------------------------------------------
      DO 225 Z=NZ1,NZM1
      IF(C(2, Z, T+1) . LE. C.00) GO TO 220
C
C-----------------------------------------------------------------------
C           CHOOSE THE NUMBER OF POINTS (NDIM) TO BE USED IN THE
C           EXTRAPOLATION.
C-----------------------------------------------------------------------
      NDIM=3
      NSTOP=NDIM+1
C
C-----------------------------------------------------------------------
C           PLACE CONCENTRATICNS IN PROPER ORDER FOR SUBROUTINE ACFI
C-----------------------------------------------------------------------
      DO 210 X=2,NSTOP
      ARG(X - 1) = FLOAT(X)
      VALY(X-1) = C(X, Z, T+1)
      IF(VALY(X-1).LE.0.0) GC TO 205
      GO TO 210
  205 NDIM = X-1
      GO TO 211
  210 CONTINUE
```

```
  211 CONTINUE
      EPS = VALY(1)/1000.
      CALL ACFI   ( 1.0, ARG, VALY, Y, NDIM, EPS, IER, J )
      C(1, Z, T+1)=Y
C
C-----------------------------------------------------------------------
C            IF THE EXTRAPOLATED VALUE IS TOO HIGH OR IS NEGATIVE,THEN USE
C            A PROPORTION EQUATION.
C-----------------------------------------------------------------------
      IF(Y.LE.0.0.OR.Y.GT.C(2,Z,T+1)) C(1,Z,T+1)=C(2,Z,T+1)*C(2,Z,T+1)
     $  /C(3,Z,T+1)
      GO TO 225
  220 C(1, Z, T+1) = 0.0
  225 CONTINUE
  228 CONTINUE
      IF(LTYPE.EQ.1.AND.MSL.EQ.NX) GO TO 251
      DO 250 Z=NZ1,NZM1
      IF ( C(NX-1, Z, T+1) . LE. 0.00) GO TO 245
      NDIM=3
      NSTOP=NDIM
      DO 235 I=1,NSTOP
      IBACK = NX - I
      ARG(I) = FLOAT(I+1)
      VALY(I) = C( IBACK, Z, T+1)
      IF( VALY(I) . LE. 0.00) GO TO 230
      GO TO 235
  230 NDIM = I
      GO TO 236
  235 CONTINUE
  236 CONTINUE
      EPS = VALY(1)/1000.
```

```
      CALL ACFI   ( 1.0, ARG, VALY, Y, NDIM, EPS, IER, J )
      C( NX, Z, T+1) = Y
      IF(Y.LE.0.0.OR.Y.GT.C(NX-1,Z,T+1))C(NX,Z,T+1)=C(NXM1,Z,T+1)*
     $  C(NXM1,Z,T+1)/C(NXM2,Z,T+1)
      GO TO 250
  245 C(NX, Z, T+1) = 0.0
  250 CONTINUE
  251 CONTINUE
      IF(NZ.EQ.1.OR.OPTION.EQ.6) RETURN
C
C----------------------------------------------------------------------
C            EXTRAPOLATE THE PROFILE IN THE Z DIRECTION
C----------------------------------------------------------------------
      DO 275 X = 1, NX
      IF(C(X, 2, T+1) . LE. 0.00) GO TO 270
      NDIM=3
      NSTOP=NDIM
      DO 260 Z = 1, NSTOP
       ARG(Z) = FLOAT(Z+1)
      VALY(Z) = C(X, Z+1, T+1)
      IF( VALY(Z) . LE. 0.00) GO TO 255
      GO TO 260
  255 NDIM = Z
      GO TO 261
  260 CONTINUE
  261 CONTINUE
      EPS = VALY(1)/1000.
      CALL ACFI   ( 1.0, ARG, VALY, Y, NDIM, EPS, IER, J )
      C ( X, 1, T+1) = Y
```

```
      IF(Y.LE.0.0.OR.Y.GT.C(X,2,T+1))C(X,1,T+1)=C(X,2,T+1)*C(X,2,T+1)
     $  /C(X,3,T+1)
      IF(C(X,1,T+1).LT.0.0) C(X,1,T+1)=0.0
      GO TO 275
  270 C(X, 1, T+1) = 0.00
  275 CONTINUE
      DO 295 X = 1, NX
      IF (C(X, NZ-1, T+1) . LE. 0.00) GO TO 292
      NDIM=3
      NSTOP=NDIM
      DO 285 I = 1, NSTOP
      ARG(I) = FLOAT(I+1)
      IBACK = NZ - I
      VALY(I) = C(X, IBACK, T+1)
      IF(VALY(I). LE. 0.00) GO TO 280
      GO TO 285
  280 NDIM = I
      GO TO 286
  285 CONTINUE
  286 CONTINUE
      EPS = VALY(1)/ 1000.
      CALL ACFI   ( 1.0, ARG, VALY, Y, NDIM, EPS, IER, J )
      C(X, NZ, T+1) = Y
      IF(Y.LE.0.0.OR.Y.GT.C(X,NZ-1,T+1)) C(X,NZ,T+1)=C(X,NZM1,T+1)*
     $  C(X,NZM1,T+1)/C(X,NZM2,T+1)
      GO TO 295
  292 C(X, NZ, T+1)=0.0
  295 CONTINUE
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
552         SUBROUTINE ACFI   ( X, ARG, VALY, Y, NDIM, EPS, IER, J )
      C
      C             ****************************
      C             ****** SUBROUTINE ACFI *****
      C             ****************************
      C
      C     ...................................................................
      C
      C        PURPOSE
      C           TO INTERPOLATE FUNCTION VALUE Y FOR A GIVEN ARGUMENT VALUE
      C           X USING A GIVEN TABLE (ARG,VAL) OF ARGUMENT AND FUNCTION
      C           VALUES.
      C
      C        DESCRIPTION OF PARAMETERS
      C           X      - THE ARGUMENT VALUE SPECIFIED BY INPUT.
      C           ARG    - THE INPUT VECTOR (DIMENSION NDIM) OF ARGUMENT
      C                    VALUES OF THE TABLE (POSSIBLY DESTROYED).
      C           VAL    - THE INPUT VECTOR (DIMENSION NDIM) OF FUNCTION
      C                    VALUES OF THE TABLE (DESTROYED).
      C           Y      - THE RESULTING INTERPOLATED FUNCTION VALUE.
      C           NDIM   - AN INPUT VALUE WHICH SPECIFIES THE NUMBER OF
      C                    POINTS IN TABLE (ARG,VAL).
      C           EPS    - AN INPUT CONSTANT WHICH IS USED AS UPPER BOUND
      C                    FOR THE ABSOLUTE ERROR.
      C           IER    - A RESULTING ERROR PARAMETER.
      C
```

```
C        REMARKS
C           (1) TABLE (ARG,VAL) SHOULD REPRESENT A SINGLE-VALUED
C               FUNCTION AND SHOULD BE STORED IN SUCH A WAY, THAT THE
C               DISTANCES ABS(ARG(I)-X) INCREASE WITH INCREASING
C               SUBSCRIPT I. TO GENERATE THIS ORDER IN TABLE (ARG,VAL),
C               SUBROUTINES ATSG, ATSM OR ATSE COULD BE USED IN A
C               PREVIOUS STAGE.
C           (2) NO ACTION BESIDES ERROR MESSAGE IN CASE NDIM LESS
C               THAN 1.
C           (3) INTERPOLATION IS TERMINATED EITHER IF THE DIFFERENCE
C               BETWEEN TWO SUCCESSIVE INTERPOLATED VALUES IS
C               ABSOLUTELY LESS THAN TOLERANCE EPS, OR IF THE ABSOLUTE
C               VALUE OF THIS DIFFERENCE STOPS DIMINISHING, OR AFTER
C               (NDIM-1) STEPS (THE NUMBER OF POSSIBLE STEPS IS
C               DIMINISHED IF AT ANY STAGE INFINITY ELEMENT APPEARS IN
C               THE DOWNWARD DIAGONAL OF INVERTED-DIFFERENCES-SCHEME
C               AND IF IT IS IMPOSSIBLE TO ELIMINATE THIS INFINITY
C               ELEMENT BY INTERCHANGING OF TABLE POINTS).
C               FURTHER IT IS TERMINATED IF THE PROCEDURE DISCOVERS TWO
C               ARGUMENT VALUES IN VECTOR ARG WHICH ARE IDENTICAL.
C               DEPENDENT ON THESE FOUR CASES, ERROR PARAMETER IER IS
C               CODED IN THE FOLLOWING FORM
C                IER=0 - IT WAS POSSIBLE TO REACH THE REQUIRED
C                        ACCURACY (NO ERROR).
C                IER=1 - IT WAS IMPOSSIBLE TO REACH THE REQUIRED
C                        ACCURACY BECAUSE OF ROUNDING ERRORS.
C                IER=2 - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE
C                        NDIM IS LESS THAN 2, OR THE REQUIRED ACCURACY
C                        COULD NOT BE REACHED BY MEANS OF THE GIVEN
C                        TABLE. NDIM SHOULD BE INCREASED.
C                IER=3 - THE PROCEDURE DISCOVERED TWO ARGUMENT VALUES
C                        IN VECTOR ARG WHICH ARE IDENTICAL.
```

```
C
C        METHOD
C           INTERPOLATION IS DONE BY CONTINUED FRACTIONS AND INVERTED-
C           DIFFERENCES-SCHEME. ON RETURN Y CONTAINS AN INTERPOLATED
C           FUNCTION VALUE AT POINT X, WHICH IS IN THE SENSE OF REMARK
C           (3) OPTIMAL WITH RESPECT TO GIVEN TABLE. FOR REFERENCE, SEE
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,
C           MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.395-406.
C
C     ..................................................................
C
C
C
      DIMENSION ARG(10), VALY(10), VAL(10)
      DO 100 M = 1,NDIM
      VAL(M) = VALY(M)
  100 CONTINUE
      IER=2
      IF(NDIM)20,20,1
    1 Y=VAL(1)
      DELT2=0.
      IF(NDIM-1)20,20,2
C
C     PREPARATIONS FOR INTERPOLATICN LOOP
    2 P2=1.
      P3=Y
      Q2=0.
      Q3=1.
C
C
```

```
C     START INTERPOLATICN LOCP
      DO 16 I=2,NDIM
      II=0
      P1=P2
      P2=P3
      Q1=Q2
      Q2=Q3
      Z=Y
      DELT1=DELT2
      JEND=I-1
C
C     COMPUTATION OF INVERTED DIFFERENCES
    3 AUX=VAL(I)
      DO 10 J=1,JEND
      H=VAL(I)-VAL(J)
      IF(ABS(H)-1.E-6*ABS(VAL(I)))4,4,9
    4 IF(ARG(I)-ARG(J))5,17,5
    5 IF(J-JEND)8,6,6
C
C     INTERCHANGE ROW I WITH ROW I+II
    6 II=II+1
      III=I+II
      IF(III-NDIM)7,7,19
    7 VAL(I)=VAL(III)
      VAL(III)=AUX
      AUX=ARG(I)
      ARG(I)=ARG(III)
      ARG(III)=AUX
      GOTO 3
C
```

```
C     COMPUTATION OF VAL(I) IN CASE VAL(I)=VAL(J) AND J LESS THAN I-1
    8 VAL(I)=1.E75
      GOTO 10
C
C     COMPUTATION OF VAL(I) IN CASE VAL(I) NOT EQUAL TO VAL(J)
    9 VAL(I)=(ARG(I)-ARG(J))/H
   10 CONTINUE
C     INVERTED DIFFERENCES ARE COMPUTED
C
C     COMPUTATION OF NEW Y
      P3=VAL(I)*P2+(X-ARG(I-1))*P1
      Q3=VAL(I)*Q2+(X-ARG(I-1))*Q1
      IF(Q3)11,12,11
   11 Y=P3/Q3
      GOTO 13
   12 Y=1.E75
   13 DELT2=ABS(Z-Y)
      IF(DELT2-EPS)19,19,14
   14 IF(I-8)16,15,15
   15 IF(DELT2-DELT1)16,18,18
   16 CONTINUE
C     END OF INTERPOLATICN LCOP
C
C
      RETURN
C
C     THERE ARE TWO IDENTICAL ARGUMENT VALUES IN VECTOR ARG
   17 IER=3
      RETURN
C
```

```
C      TEST VALUE DELT2 STARTS OSCILLATING
    18 Y=Z
       IER=1
       RETURN
C
C      THERE IS SATISFACTORY ACCURACY WITHIN NDIM-1 STEPS
    19 IER=0
    20 RETURN
       END
```

```
C-----------------------------------------------------------------------
      SUBROUTINE PRINT1
C
C             *****************************
C             ***** SUBRCUTINE PRINT1 *****
C             *****************************
C
C          THIS SUBROUTINE CALCULATES THE CONVERSION VALUES FOR MANY OF
C     THE INPUT PARAMETERS AND PRINTS OUT A SUMMARY OF THE INPUT DATA
C     AND STABILITY CRITERIA.
C
C-----------------------------------------------------------------------
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
      UNITS1 = 86400./5280.
      UNITS2 = 86400./(5280.*5280.)
      DELDAY = DELTAT/86400.
      SECS1=DELT1*86400.
      SECS2=DELT2*8640C.
      DELMX=DELTAX/5280.
      DELMZ=DELTAZ/5280.
```

```
      TOTALX=(NX-1)*DELTAX
      TOTMX=TOTALX/5280.
      TOTALZ=(NZ-1)*DELTAZ
      TOTMZ=TOTALZ/5280.
      VX1=VXMAX*UNITS1
      VX2=VXMIN*UNITS1
      VZ1=VZMAX*UNITS1
      IF(VZMIN.EQ.1000000.) VZMIN=0.
      VZ2=VZMIN*UNITS1
      EX1=EXMAX*UNITS2
      EX2=EXMIN*UNITS2
      EZ1=EZMAX*UNITS2
      IF(EZMIN.EQ.1000000.) EZMIN=0.
      EZ2=EZMIN*UNITS2
      XKD1=KDMAX*86400.
      XKD2=KDMIN*86400.
      SXM=STABLX/5280.
      SZM=STABLZ/5280.
      STD=STABLT/86400.
      PRINT 500
      PRINT 505
      PRINT 507
      PRINT 510
      PRINT 507
      IF(NZ.EQ.1) PRINT 515
      IF(NZ.NE.1) PRINT 520
      PRINT 507
      IF(LTYPE.EQ.1) GO TO 7
      IF(LTYPE.EQ.2) GO TO 5
```

```
      PRINT 530
      GO TO 10
    5 PRINT 525
      GO TO 10
    7 PRINT 523
   10 CONTINUE
      PRINT 507
      PRINT 535
      PRINT 507
      PRINT 505
      PRINT 540, DELTAX, DELMX
      PRINT 545, DELTAZ,DELMZ
      PRINT 550, SECS1, DELT1
      PRINT 551, ITER1
      PRINT 552, SECS2, DELT2
      PRINT 553, ITER2
      PRINT 555,NX
      PRINT 560, NZ
      PRINT 565, VXMAX, VX1
      PRINT 570, VXMIN, VX2
      PRINT 572,VZMAX,VZ1
      PRINT 573,VZMIN,VZ2
      PRINT 575,EXMAX,EX1
      PRINT 580,EXMIN,EX2
      PRINT 585, EZMAX,EZ1
      PRINT 590, EZMIN,EZ2
      PRINT 600,WMAX
      PRINT 605,WMIN
      PRINT 610,XKD1,KDMAX
      PRINT 615,XKD2,KDMIN
      PRINT 620,DEMAX
      PRINT 630, NC
```

```
C
C-----------------------------------------------------------------------
C          XUPEAK AND XLPEAK ARE CALCULATED ASSUMING DISTANCES ARE
C       DECREASING IN THE DOWNSTREAM DIRECTION
C-----------------------------------------------------------------------
        XUPEAK = XSTART - (MSU-1)*DELMX
        XLPEAK = XSTART - (MSL-1)*DELMX
        PRINT 635, XUPEAK,MSU
        PRINT 637, XLPEAK,MSL
        PRINT 500
        PRINT 505
        PRINT 502
        PRINT 640
        PRINT 645,STABLX,SXM
        PRINT 650,STABLZ,SZM
        PRINT 655,STABLT,STD
        PRINT 660,R1
        PRINT 665, R2
        PRINT 505
        IF(NC.EQ.0) GO TO 100
        PRINT 500
        PRINT 700
        IF(NZ.EQ.1) GO TO 50
        DO 25 I=1,NC
C
C-----------------------------------------------------------------------
C            WHERE IS CALCULATED ASSUMING DISTANCES ARE DECREASING IN
C       THE DOWNSTREAM DIRECTION
C-----------------------------------------------------------------------
```

```
      WHERE=XSTART-(INSECT(I)-1)*DELMX
      PRINT 710, INSECT(I),WHERE,DELTAZ
      DO 25 Z=1,NZ
      PRINT 715,INSECT(I),Z,CONCIN(I,Z)
   25 CONTINUE
      GO TO 100
   50 DO 75 I=1,NC
      WHERE=XSTART-(INSECT(I)-1)*DELMX
      PRINT 705, INSECT(I), WHERE, CCNCIN(I,1)
      PRINT 502
   75 CONTINUE
  100 CONTINUE
      PRINT 500
  500 FORMAT(1H1)
  502 FORMAT(/)
  505 FORMAT(T45, 38('*'))
  507 FORMAT(T45,'*',T82,'*')
  510 FORMAT(T45,'*',T56,'ESTUARY SIMULATION', T82, '*')
  515 FORMAT(T45, '*', T48, 'ONE DIMENSIONAL EXPLICIT METHOD',T82,'*')
  520 FORMAT(T45, '*', T48, 'TWO DIMENSIONAL EXPLICIT METHOD',T82,'*')
  523 FORMAT(T45, '*', T53, 'CONSTANT CONCENTRATION',T82,'*')
  525 FORMAT(T45, '*', T54, 'INSTANTANEOUS RELEASE', T82,'*')
  530 FORMAT(T45, '*', T56, 'CONTINUOUS LOADING', T82, '*')
  535 FORMAT(T45, '*', T51, 'PROGRAMMER -- JONATHAN YOUNG', T82,'*')
  540 FORMAT(   /, T53, 'X INCREMENT =', F7.0, ' FEET (', F8.5, ' MILES
     $)',/)
  545 FORMAT(T53, 'Z INCREMENT =', F7.0, ' FEET (', E10.3, ' MILES )'/)
  550 FORMAT(T42, 'INITIAL TIME INCREMENT =',F7.0, ' SECONDS (', F6.4,
     $  ' DAYS )'/)
  551 FORMAT(T44,'NUMBER OF ITERATIONS = ',I5,/)
```

```
552 FORMAT( T42, 'REVISED TIME INCREMENT = ', F7.0, ' SECONDS(',F6.4,
   $  ' DAYS )'/)
553 FORMAT(T44,'NUMBER OF ITERATIONS = ',I5,/)
555 FORMAT(T37, 'NUMBER OF HORIZONTAL POINTS =', I4,/)
560 FORMAT(T39, 'NUMBER OF VERTICAL POINTS =',I4,/)
565 FORMAT(T37, 'MAXIMUM HORIZONTAL VELOCITY = ', F5.2, ' FEET/SECOND
   $(',F6.2, ' MILES/DAY )',/)
570 FORMAT(T37, 'MINIMUM HORIZONTAL VELOCITY = ', F5.2, ' FEET/SECOND
   $(',F6.2, ' MILES/DAY )',/)
572 FORMAT(T39, 'MAXIMUM VERTICAL VELOCITY =', E10.3, ' FEET/SECOND ('
   $ , E10.3, ' MILES/DAY )',/)
573 FORMAT(T39, 'MINIMUM VERTICAL VELOCITY =', E10.3, ' FEET/SECOND ('
   $ , E10.3, ' MILES/DAY )',/)
575 FORMAT(T35, 'MAXIMUM HORIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
   $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
580 FORMAT(T35, 'MINIMUM HORIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
   $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
585 FORMAT(T37, 'MAXIMUM VERTICAL DISPERSION = ', F9.3, ' FEET SQUARED
   $ / SECOND ( ', E10.3, ' MILES SQUARED / DAY )',/)
590 FORMAT(T37, 'MINIMUM VERTICAL DISPERSION = ', F9.3, ' FEET SQUARED
   $ / SECOND ( ', E10.3, ' MILES SQUARED / DAY )',/)
600 FORMAT(T51, 'MAXIMUM WIDTH = ', F5.0, ' FEET',/)
605 FORMAT(T51, 'MINIMUM WIDTH = ', F5.0, ' FEET',/)
610 FORMAT(T46, 'MAXIMUM DECAY RATE = ', F5.3, ' PER DAY ( ', E10.3,
   $ ' PER SECOND )', /)
615 FORMAT(T46, 'MINIMUM DECAY RATE = ', F5.3, ' PER DAY ( ', E10.3,
   $ ' PER SECOND )', /)
620 FORMAT(T42, 'OTHER DEMANDS, MAXIMUM = ', F5.2,/)
630 FORMAT(T26, 'NUMBER OF INITIAL CONCENTRATION VALUES = ',I2,/)
```

```
635 FORMAT(T16, 'INITIAL LOCATION OF UPPERMOST PEAK CONCENTRATION = ',
   $  F6.2, ' MILES (SECTION NUMBER ', I2,')',/)
637 FORMAT(T16, 'INITIAL LOCATION OF LOWERMOST PEAK CONCENTRATION = ',
   $  F6.2, ' MILES (SECTION NUMBER ', I2,')')
640 FORMAT(T56, 'STABILITY CRITERIA',/)
645 FORMAT(T35, 'MAXIMUM ALLOWABLE X INCREMENT = ', F7.0, ' FEET (',
   $ F8.5, 'MILES )',/)
650 FORMAT(T35, 'MAXIMUM ALLOWABLE Z INCREMENT = ', F7.0, ' FEET (',
   $ F8.5, 'MILES )',/)
655 FORMAT(T32, 'MAXIMUM ALLOWABLE TIME INCREMENT = ', F7.0, ' SECONDS
   $ (', E10.3, ' DAYS )',/)
660 FORMAT(T38, 'ACTUAL DELTAT/(DELTAX**2.) = ', E10.3,/)
665 FORMAT(T38, 'ACTUAL DELTAT/(DELTAZ**2.) = ', E10.3,/)
700 FORMAT(T25, 'LOCATIONS OF INITIAL CONCENTRATIONS',//)
705 FORMAT(    'A WASTE SOURCE IS LOCATED AT STATION ', I2, ' (MILE ',
   $ F6.3 , ' ).  THE CONCENTRATION IS', F10.2, ' PPM.')
710 FORMAT( 1X,'AN INITIAL CONCENTRATION IS FOUND AT STATION ',I2,
   1  ' (MILE ', F6.3, ' ).',/, ' THE CONCENTRATIONS AT ', F7.1,
   2  ' FOOT INTERVALS WITH DEPTH ARE')
715 FORMAT(T15, 'C(', I2','I2',1) = ', F9.2, ' PPM')
    RETURN
    END
```

```
      C-----------------------------------------------------------------------
766         SUBROUTINE PRINT2
      C
      C              ****************************
      C              ***** SUBRCUTINE PRINT2 *****
      C              ****************************
      C
      C          THIS SUBROUTINE PRINTS OUT THE TIME AND THE CONCENTRATION
      C     PROFILE ACCORDING TO A FORMAT PREVIOUSLY READ INTO THE PROGRAM.
      C     THIS SUBROUTINE CAN BE CHANGED ACCORDING TO THE NEEDS OF THE USER.
      C     ***WARNING--DO NOT CHANGE ANY VARIABLE OCCURING IN A COMMON BLOCK.
      C
      C-----------------------------------------------------------------------
767         INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
768         REAL KD,KDMAX,KDPIN,KDAYS,LOWER
769         COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
770         COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
771         DIMENSION NZREV(20)
772         DIMENSION R(51,21)
773         DAYS = TIME/8640C.
774         HOURS=DAYS*24.
775         WRITE(6,FMT1) HOURS,DAYS
```

```
      DO 100 IZ = 1, NZ
      NZREV(IZ)=IFIX(DELTAZ*(IZ-2))
  100 CONTINUE
      WRITE(6,FMT2) (NZREV(I),I=2,NZM1)
C
C-----------------------------------------------------------------------
C        ALLOW FOR A BACKGROUND CONCENTRATION OF 10.0
C-----------------------------------------------------------------------
      DO 150 X=1,NX
      DO 150 Z=1,NZ
      R(X,Z)=C(X,Z,T)+10.0
  150 CONTINUE
      XFIRST=XSTART-(NX-1)*DELTAX/5280.
      DO 200 X=1,NX,XPRINT
      M=NX+1-X
      XMILE = XFIRST + ((X-1)*DELTAX*XPRINT)/5280.
      WRITE(6,FMT3) XMILE, (R(M,IZ),IZ=1,NZ)
  200 CONTINUE
      RETURN
      END
```

```
C-----------------------------------------------------------------------
      SUBROUTINE ERROR(*)
C
C              ***************************
C             **** SUBROUTINE ERROR ******
C              ***************************
C
C          THIS SUBROUTINE CORRECTS SCME OF THE MOST COMMON ERRORS
C     IN THE INPUT DATA AND PRINTS OUT APPROPRIATE ERROR MESSAGES.
C
C-----------------------------------------------------------------------
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
      IF(OPTION.LT.8) GO TO 200
      PRINT 181
  181 FORMAT(/////,' ****PROGRAM TERMINATED BECAUSE OPTION IS GREATER TH
     $AN 7****',/,1H1)
      RETURN1
  200 IF(LTYPE.LT.3) GO TO 300
      PRINT 201
```

```
      201 FORMAT(/////, ' ****PROGRAM TERMINATED BECAUSE LTYPE IS GREATER TH
         $AN OR EQUAL TO 3.',/,'    IF LTYPE EQUALS 3, THE USER MUST SUPPLY A
         $ SUBROUTINE FOR THE LOADING CONDITIONS****',/,1H1)
          RETURN1
      300 CONTINUE
          IF(ITER.NE.0) GO TO 820
C
C-----------------------------------------------------------------------
C          TERMINATE PROGRAM WITH PRINTED MESSAGE IF INPUT DATA
C          VIOLATES STABILITY CRITERIA
C-----------------------------------------------------------------------
          PRINT 815
      815 FORMAT(////,1X,10('*'),'PROGRAM TERMINATED BECAUSE STABILITY CONDI
         $TIONS WERE VIOLATED****', /,1H1)
          RETURN1
      820 CONTINUE
          RETURN
          END
```

Input Data for MASSTRANS-I and MASSTRANS-II

Each line represents a new card unless single spaced.

1.

33, 11

0.25, 5.0

0.006944444, 2

0.0, 0

21, 1

17., 0.166667, 1, 1, 1

2, 2

33*0.0, 33*0.366667, 33*0.420, 33*0.440, 33*0.440, 33*0.433333,

33*0.389333, 33*0.27333, 33*0.233333, 33*0.233333, 33*0.0

363*450.

497., 510., 524., 537., 550., 562., 575., 562., 550., 537.,
525., 568., 611., 654., 697

756., 814., 872., 931., 863., 796., 728., 661., 657., 654.,
650., 647., 633., 619., 605.,

591, 586., 582.

33*2.50

363*0.0

1

3.2, 3.2, 2.0, 1.7, 2.0, 3.0, 3.0, 1.8, 0.2, 0.0, 0.0

2

4.0, 4.0, 3.5, 2.0, 3.0, 4.0, 4.0, 2.1, 0.3, 0.0, 0.0

3

6.2, 6.2, 5.0, 4.0, 4.0, 5.0, 5.0, 2.3, 0.4, 0.0, 0.0

4

9.5, 9.5, 7.0, 6.0, 6.0, 7.0, 6.0, 2.5, 0.5, 0.0, 0.0

5

90.0, 90.0, 9.0, 9.0, 9.0, 10.0, 7.0, 2.7, 0.6, 0.0, 0.0

6

115.0, 115.0, 70.0, 40.0, 30.0, 25.0, 9.0, 3.0, 0.8, 0.0, 0.0

7

165., 165., 210., 290., 120., 55., 12., 3.5, 1.0, 0.0, 0.0

8

290., 290., 490., 440., 165., 70., 20., 4., 0.8, 0.0, 0.0

9

740., 740., 890., 320., 140., 55., 14., 3.3, 0.6, 0.0, 0.0

10

1990., 1990., 540., 215., 90., 30., 10., 2.6, 0.4, 0.0, 0.0

11

990., 990., 290., 120., 50., 15., 7., 2., 0.3, 0.0, 0.0

12

290., 290., 190., 75., 20., 8., 4., 1.6, 0.2, 0.0, 0.0

13

165., 165., 75., 25., 9., 5., 2., 1.2, 0.1, 0.0, 0.0

14

90., 90., 10., 9., 5., 2., 1.5, 0.8, 0.0, 0.0, 0.0

```
15
20., 20., 8., 7., 2., 1.7, 1.0, 0.4, 0.0, 0.0, 0.0
16
9., 9., 7., 5., 1.8, 1.5, 0.7, 4*0.0
17
7., 7., 6., 4., 1.6, 1.2., 0.4, 4*0.0
18
5., 5., 5., 3., 1.4, 1., 0.2, 4*0.0
19
4., 4., 4., 2., 1.3, 0.7, 5*0.0
20
3., 3., 3., 1.8, 1.2, 0.5, 5*0.0
21
2.5, 2.5, 2.0, 1.6, 1.1, 0.2, 5*0.0
363*0.0
33*0.00333333, 33*0.00333333, 33*0.00382, 33*0.004, 33*0.004,
33*0.00394,
33*0.00354, 33*0.002485, 33*0.00212, 33*0.00212, 33*0.00212
(///,12X,100('*'), /, 12X, '*',/, 12X,'*',2X,
'CONCENTRATIONS AT TIME = ',
F7.2, ' HOURS ( ', F6.4, ' DAYS)', / ,12X, '*',/, 12X,
100('*') )
12X, '*', /, 12X, '*', 42X, 'DEPTH', /, 12X, '*', /, 12X,
'*',3X, 'IMAGE', 4X,
9(I2, ' FEET '), 'IMAGE',/, 1X, 111('*'),/, 12X, '*')
(1X, 'MILE', F5.2, 1X, '*', 2X, 11(F6.1, 4X))
             // END OF DATA FOR MASSTRANS-I //
```

33*0.0, 33*0.366667, 33*0.420, 33*0.440, 33*0.440, 33*0.433333,

33*0.389333, 33*0.273333, 33*0.233333, 33*0.233333, 33*0.0

363*450.

363*0.0

33*0.00333333, 33*0.00333333, 33*0.00382, 33*0.004, 33*0.004, 33*0.00394,

33*0.00354, 33*0.002485, 33*0.00212, 33*0.00212, 33*0.00212

// THIS PAGE OF DATA USED FOR MASSTRANS-II ONLY //

COMPUTER PROGRAM FOR

MASSTRANS-II

Object Code =	67,944	bytes
Array Area =	62,924	bytes
Total =	130,868	bytes

```
C         ***********************************************************
C         *                                                         *
C         *                     MASSTRANS-II                        *
C         *                                                         *
C         *             ESTUARY MASS TRANSPORT MODEL                *
C         *                                                         *
C         *  TWO DIMENSIONAL NUMERICAL ANALYSIS - IMPLICIT METHOD   *
C         *                                                         *
C         ******------------------------------------------------*******
C              *                                              *
C              *  LOGIC AND PROGRAMMING BY                    *
C              *                                              *
C              *         P. JCNATHAN YOUNG                    *
C              *    C/O  ENVIRONMENTAL ENGINEERING DIVISION   *
C              *         CIVIL ENGINEERING DEPARTMENT         *
C              *         TEXAS A&M UNIVERSITY                 *
C              *         CULLEGE STATION, TEXAS 77840         *
C              *                                              *
C              ************************************************
C
C
C          THIS COMPUTER PROGRAM WAS DESIGNED PRIMARILY TO ANALYZE THE
C     MASS TRANSPORT OF DISSCLVED MATERIALS IN A PARTIALLY STRATIFIED
C     ESTUARY.  THESE MATERIALS MAY BE CONSERVATIVE OR NON-CONSERVATIVE.
C     A CRANK-NICOLSON IMPLICIT FINITE DIFFERENCE APPROXIMATION TO THE
C     ESTUARY MASS TRANSPORT EQUATION FORMS THE BASIS OF THE
C     COMPUTATIONS.
C
```

```
C          THIS PROGRAM WAS DEVELOPED TO ANALYZE ESTUARIES WHOSE
C     CHARACTERISTICS DO NOT VARY SIGNIFICANTLY WITH WIDTH.  THE WIDTH
C     OF THE ESTUARY MAY BE VARIED THROUGHOUT BUT THE CONCENTRA-
C     TIONS OF DISSOLVED MATERIALS AT EACH CROSS SECTION ARE CONSIDERED
C     TO BE UNCHANGING IN THE LATERAL DIRECTION.  ALL PHYSICAL AND
C     HYDRODYNAMIC CHARACTERISTICS MAY VARY WITH TIME IN THE LONGITU-
C     DINAL AND VERTICAL DIRECTIONS.  THE PROGRAM CAN BE APPLIED WITH
C     EQUAL EASE TO THE TWO HORIZONTAL DIRECTIONS, ALLOWING DEPTH TO
C     VARY RATHER THAN WIDTH.
C
C          CONCENTRATION PROFILES CAN BE CALCULATED BY THIS PROGRAM FOR
C     CONTINUOUS OR INSTANTANEOUS RELEASES.  INPUT DATA MAY INCLUDE GRID
C     DIMENSIONS, DISTANCE INCREMENTS, TIME INCREMENTS, WIDTHS, LOADING
C     PARAMETERS, VELOCITIES, DISPERSION COEFFICIENTS, DECAY RATES,
C     BENTHAL DEMANDS, AND OTHER SOURCE OR SINK TERMS.
C
C          THE TIME INCREMENTS CAN BE INCREASED OR DECREASED AT ANY TIME
C     DURING THE CALCULATION OF THE CONCENTRATION PROFILE.  THE DIS-
C     TANCES BETWEEN GRID POINTS CAN BE INCREASED AT ANY TIME, AND A
C     ROUTINE WITHIN THE PROGRAM WILL CHOOSE THE APPROPRIATE VALUES FROM
C     THE PREVIOUSLY CALCULATED PROFILE AND WILL PLACE THESE VALUES IN
C     THE DESIRED LOCATIONS IN THE NEW GRID SYSTEM.  THE NUMBER OF GRID
C     POINTS MAY BE INCREASED OR DECREASED AT ANY TIME.  LIKEWISE, AT
C     ANY TIME DURING THE CALCULATION, A NEW SET OF DATA FOR PHYSICAL
C     AND HYDRODYNAMIC CONDITIONS MAY BE READ INTO THE PROGRAM.
```

```
C            A USER OF THIS PRCGRAM MUST BE FAMILIAR WITH THE LIMITATICNS
C       ON ACCURACY AND STABILITY INHERENT IN THE TYPE OF NUMERICAL PROCE-
C       DURE USED IN THESE CALCULATIONS.  A SUBROUTINE WITHIN THE PROGRAM
C       PRINTS OUT THE PROPER INCREMENTS WHICH WILL INSURE STABILITY.
C       ANOTHER SUBROUTINE EXTRAPOLATES THE CONCENTRATIONS AT THE
C       BOUNDARIES--SEVERAL METHODS CAN BE USED FOR THESE EXTRAPOLATIONS
C       DEPENDING ON THE TYPE CF PROFILE BEING ANALYZED AND THE CHOICE OF
C       THE USER.  A SUBROUTINE IS ALSO INCLUDED WHICH PRINTS OUT ERROR
C       MESSAGES AND TERMINATES THE PROGRAM IF CERTAIN INCONSISTENCIES
C       OCCUR IN THE INPUT DATA.
C
C            THIS PROGRAM WAS DEVELOPED PRIMARILY TO ANALYZE PARTIALLY
C       STRATIFIED ESTUARIES WHICH HAVE BEEN DREDGED OUT TO A FAIRLY
C       CONSTANT DEPTH AT THE CENTERLINE OF THE CHANNEL--THESE ESTUARIES
C       ARE COMMON IN THE GULF COAST REGION.  APPLICATION OF THIS PROGRAM
C       TO PARTIALLY STRATIFIED ESTUARIES WITH VARIABLE DEPTHS WOULD
C       REQUIRE MODERATE REVISIONS TO THE PROGRAM AND WOULD MAKE THE
C       PROGRAM ESTUARY-DEPENDENT.
C
C            THIS COMPUTER PROGRAM CAN ALSC BE APPLIED TO ESTUARIES WHICH
C       ARE WELL-MIXED IN THE VERTICAL DIRECTION.  THIS OPTION ALLOWS FOR
C       VARYING WIDTH OR VARYING DEPTH AND USES MOST OF THE ROUTINES
C       AVAILABLE TO THE TWO-DIMENSIONAL ANALYSIS.
C
C            QUESTIONS REGARDING THIS PROGRAM MAY BE REFERRED TO
C       JONATHAN YOUNG AT HYDROSCIENCE, INC, 363 OLD HOOK ROAD,
C       WESTWOOD, NEW JERSEY  07675   PHONE 201 / 666-2600
C
C-----------------------------------------------------------------------
```

```
C-----------------------------------------------------------------------
C
C         ***INPUT VARIABLES***
C
C
C        SET           = DATA SET NUMBER
C                        --CAN BE EQUAL TO 1 IF NO NEW GRID DIMENSIONS
C                          ARE TO BE READ IN--MUST BE GREATER THAN 1 IF
C                          GRID DIMENSIONS ARE TO BE CHANGED
C        **************NOTE--CPTIONAL GRID ADJUSTMENT PARAMETERS ARE
C                          INSERTED HERE.  THESE ARE EXPLAINED BELOW
C        NX            = NUMBER OF GRID POINTS IN THE X DIRECTION
C        NZ            = NUMBER OF GRID POINTS IN THE Z DIRECTION
C        DELMX         = X INCREMENT ( MILES )
C        DELTAZ        = Z INCREMENT ( FEET )
C        DELT1         = FIRST TIME INCREMENT ( DAYS )
C        ITER1         = NUMBER OF ITERATIONS USING DELT1
C        DELT2         = SECOND TIME INCREMENT ( DAYS )
C        ITER2         = NUMBER OF ITERATIONS USING DELT2
C        NC            = NUMBER OF GRID POINTS IN THE X DIRECTION WHERE
C                        CONCENTRATIONS ARE ADDED WITH THE DATA SET
C        NS            = NUMBER OF POINT SOURCES
C        XSTART        = MILEAGE AT UPPERMOST GRID POINT
C        TSTART        = TIME TO BE ADDED TO THE TOTAL TIME AT THE FIRST
C                        ITERATION OF THE DATA SET BEING PROCESSED( DAYS )
C        XPRINT        = PRINTING INTERVAL FOR X DIRECTION ( INTEGER )
C        TPRINT        = PRINTING INTERVAL FOR TIME ( INTEGER )
C        NPPAGE        = NUMBER OF GRIDS PRINTED PER PAGE OF OUTPUT
```

```
C       LTYPE        = TYPE CF LOADING ( INSTANTANEOUS OR CONTINUOUS )
C                        1 MEANS CONSTANT CONCENTRATION AT OUTFALL
C                          (THIS CAN BE USED FOR STEADY-STATE PROFILES)
C                        2 MEANS AN INSTANTANEOUS RELEASE
C                        3 MEANS CONTINUOUS LOADING
C       OPTION       = OPTION FOR BOUNCARY CONDITIONS
C                        1 MEANS NO TRANSFER ACROSS Z BOUNDARIES AND
C                          EXPONENTIAL EXTRAPOLATION IN X DIRECTION
C                        2 MEANS NO TRANSFER ACROSS Z BOUNDARIES AND
C                          CONSTANT SLOPE EXTRAPOLATION IN X DIRECTION
C                        3 MEANS A SYMMETRICAL DISTRIBUTION WITH
C                          EXPONENTIAL EXTRAPOLATION
C                        4 MEANS A CONSTANT SLOPE EXTRAPOLATION IN
C                          BOTH DIRECTIONS
C                        5 MEANS A SPECIAL CASE DEFINED BY THE USER
C       VX(X,Z)      = HORIZONTAL VELOCITY ( FT/SEC )
C       EX(X,Z)      = HORIZONTAL DISPERSION ( FT**2/SEC )
C       W(X)         = WIDTH ( FEET ) FOR THE TWO DIMENSIONAL CASE OR
C                      CROSS SECTIONAL AREA (FT**2) FOR THE ONE
C                      DIMENSIONAL CASE
C       KDAYS(X)     = DECAY RATE ( PER DAY )
C       DEMAND(X,Z)  = OTHER SOURCES AND SINKS (CONCENTRATION UNITS)
C       INSECT(I)    = SECTIONS INTO WHICH ARE ADDED LOADS
C                      OR CONCENTRATIONS
C       CONCIN(I,J)  = INITIAL CONCENTRATIONS ASSOCIATED WITH INSECT(I)
C
C       ***************NOTE--INITIAL CONCENTRATIONS MAY ALSO BE INPUT
C                        THROUGH THE BLOCK DATA SUBPROGRAM
```

```
C         **************NOTE--VALUES FOR VZ AND EZ ARE TO BE INCLUDED
C                            WITH THE DATA ONLY IF NZ IS GREATER THAN 1
C         VZ(X,Z)      = VERTICAL VELOCITIES ( FT/SEC )
C         EZ(X,Z)      = VERTICAL DISPERSION ( FT**2/SEC )
C
C         VXNEXT(X,Z) = HORIZONTAL VELOCITY,NEXT TIME STEP (FT/SEC )
C         EXNEXT(X,Z) = HORIZONTAL DISPERSION,NEXT TIME STEP (FT**2/SEC )
C         VZNEXT(X,Z) = VERTICAL VELOCITY,NEXT TIME STEP (FT/SEC )
C         EZNEXT(X,Z) = VERTICAL DISPERSION,NEXT TIME STEP (FT**2/SEC )
C         FMT1(I)      = FORMAT FOR HEADING FOR COMPUTED PROFILES
C         FMT2(I)      = FORMAT FOR DEPTH NOTATION FOR PROFILES
C         FMT3(I)      = FORMAT FOR OUTPUT OF COMPUTED PROFILES
C
C
C-----------------------------------------------------------------------
C
C           *** GRID ADJUSTMENT PARAMETERS ***
C
C
C         NXSKIP       = NUMBER OF X INTERVALS SKIPPED IN OLD GRID
C         NZSKIP       = NUMBER OF Z INTERVALS SKIPPED IN OLD GRID
C         NXTAKE       = LOWEST GRID NUMBER IN X DIRECTION IN THE OLD GRID
C                        FROM WHICH A CONCENTRATION IS TAKEN
C         NZTAKE       = LOWEST GRID NUMBER IN Z DIRECTION IN THE OLD GRID
C                        FROM WHICH A CONCENTRATION IS TAKEN
C         NXPUT        = LOWEST GRID NUMBER IN THE X DIRECTION IN THE NEW
C                        GRID INTO WHICH A CONCENTRATION IS PLACED
C         NZPUT        = LOWEST GRID NUMBER IN THE Z DIRECTION IN THE NEW
C                        GRID INTO WHICH A CONCENTRATION IS PLACED
C-----------------------------------------------------------------------
```

```
C-----------------------------------------------------------------------
C
C              ************************
C              ***** MAIN PROGRAM *****
C              ************************
C
C           THE MAIN PROGRAM CALLS FOR THE DATA TO BE READ IN,THEN CALLS
C      FOR THE STABILITY CONDITIONS TO BE EVALUATED,AND NEXT CALLS FOR
C      THE PRINTING OF A SUMMARY OF INPUT DATA.  THE NUMERICAL ANALYSIS
C      SUBROUTINE IS THEN CALLED FOR EITHER THE ONE-DIMENSIONAL OR TWO-
C      DIMENSIONAL CASE.  IF THE TIME INCREMENT IS TO BE CHANGED DURING
C      THIS PART OF THE PROGRAM,THE APPROPRIATE PARAMETERS ARE ADJUSTED.
C      STABILITY IS AGAIN CHECKED AND THE NUMERICAL ANALYSIS IS
C      CONTINUED.  NEXT,A NEW DATA SET MAY BE READ TO CONTINUE THE
C      ANALYSIS.  ANY NUMBER OF DATA SETS MAY BE PROCESSED.  WHEN NO MORE
C      DATA SETS ARE AVAILABLE,THE PROGRAM IS TERMINATED.
C
C           THE INPUT AND OUTPUT OF DATA CAN BE CONTROLLED IN THREE
C      WAYS--THE INPUT DATA ITSELF, THE BLOCK DATA SUBROUTINE, AND THE
C      SUBROUTINE PRINT2.  NO OTHER PART OF THE PROGRAM SHOULD REQUIRE
C      CHANGING.
C
C           THE FORTRAN USED IN THIS PROGRAM USES SEVERAL OF THE OPTIONS
C      AVAILABLE IN WATFOR AS DESCRIBED IN THE FOLLOWING TEXT**FORTRAN IV
C      WITH WATFOR AND WATFIV**BY CRESS,DIRKSEN,&GRAHAM,PRENTICE-HALL,
C      INC.,1970. THE PROGRAM CAN BE RUN ON A COMPUTER WITH A WATFOR OR
C      WATFIV COMPILER.  THE SYNTAX HAS BEEN KEPT COMPATIBLE WITH
C      FORTRAN G EXCEPT FOR THE UNFORMATTED READ AND PRINT STATEMENTS.
C
C-----------------------------------------------------------------------
```

```
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
      COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
     1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
     2               G(51),D(51),V(51)
   10 CALL IDATA(&1000)
      CALL STABLI
      CALL PRINT1
      CALL PRINT2
      CALL ERRORI(&1000)
   25 IF(NZ.EQ.1) GO TO 310
      CALL TWODIM
      GO TO 315
  310 CALL ONEDIM
  315 IF(ITER2.EQ.0) GO TO 10
      ITER=ITER2
      ITER2=0
      DELTAT=DELT2*86400.
      GO TO 25
 1000 CONTINUE
      STOP
      END
```

```
    C-----------------------------------------------------------------------
994       BLOCK DATA
    C
    C             *********************
    C             ***** BLCCK DATA *****
    C             *********************
    C
    C              THIS ROUTINE INITIALIZES SELECTED VARIABLES IN THE
    C         COMMON BLOCKS.  THIS SUBPROGRAM MAY BE USED TO ESTABLISH
    C         INITIAL CONCENTRATION PROFILES IF THE USER DESIRES.
    C
    C-----------------------------------------------------------------------
995       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
996       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
997       COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
         1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
         2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
998       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
         1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
         2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
         3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
         4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
         5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
999       COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
         1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
         2               G(51),D(51),V(51)
```

```
C
C-----------------------------------------------------------------------
C        ESTABLISH INITIAL CONCENTRATION IF DESIRED
C-----------------------------------------------------------------------
      DATA C/2142*0.0/
      DATA VX,VZ,EX,EZ,CONCIN,DEMAND/6426*0.0/
      DATA W,KDAYS,KD/153*0.0/
      DATA EXMAX,EZMAX,VXMAX,VZMAX,WMAX,KDMAX,DEMAX,TIME/8*0.0/
      DATA EXMIN,EZMIN,VXMIN,VZMIN,WMIN,KDMIN,DEMIN/7*1000000./
      DATA INSECT/51*0/
      DATA IPAGE/1/
      DATA T/1/,TNEXT/2/
      DATA VXNEXT,VZNEXT,EXNEXT,EZNEXT,CSTAR/5355*0.0/
      DATA A,B,G,D,V/255*0.0/
      END
```

```
     C-----------------------------------------------------------------------
1011       SUBROUTINE IDATA(*)
     C
     C             **************************
     C             ***** SUBROUTINE IDATA ****
     C             **************************
     C
     C          THIS SUBROUTINE READS IN THE APPROPRIATE CATA SET AND ADJUSTS
     C     THE GRID SIZE IF NECESSARY.  THE LOCATIONS OF THE UPPERMOST AND
     C     LOWERMOST PEAKS ARE DETERMINED, AND THE MINIMUM AND MAXIMUM VALUES
     C     FOR THE INPUT PARAMETERS ARE CALCULATED.  ALL OF THE INPUT DATA IS
     C     PRINTED OUT UNFORMATTED.  THE FORMAT IS READ IN FOR THE PRINTING
     C     OUT OF THE CALCULATED CONCENTRATIONS.
     C
     C-----------------------------------------------------------------------
1012       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
1013       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1014       COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
          1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
          2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1015       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
          1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
          2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
          3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
          4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
          5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1016       COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
          1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
          2               G(51),D(51),V(51)
```

```
      READ(5,12,END=13) SET
   12 FORMAT(F10.0)
      GO TO 14
   13 CONTINUE
      RETURN1
   14 CONTINUE
      IF(SET.EQ.1.)  GO TO 400
C
C-----------------------------------------------------------------------
C             CHOOSE CONCENTRATIONS FROM THE OLD GRID TO PUT INTO THE
C             APPROPRIATE PLACES IN THE NEW GRID .  THE GRID CAN ONLY BE
C             CHANGED IF LTYPE EQUALS 2.  OTHERWISE, THE DATA CARD IS READ
C             BUT  IS NOT IMPLEMENTED.
C-----------------------------------------------------------------------
      READ ,NXSKIP,NZSKIP,NXTAKE,NZTAKE,NXPUT,NZPUT
      IF(LTYPE.NE.2) GO TO 400
      NXSTOP=(NX-1)/NXSKIP+1
      NZSTOP=(NZ-1)/NZSKIP+1
      IF(NZ.EQ.1) NZSTOP=1
      DO 350 I=1,NXSTOP
      X=NXPUT-1+I
      II=NXTAKE+NXSKIP*(I-1)
      DO 350 J=1,NZSTOP
      Z=NZPUT-1+J
      JJ=NZTAKE+NZSKIP*(J-1)
      C(X,Z,TNEXT)=C(II,JJ,T)
  350 CONTINUE
  400 CONTINUE
C
C
```

```
C-----------------------------------------------------------------------
C           READ AND PRINT INPUT DATA FCR HCRIZONTAL DIRECTION
C-----------------------------------------------------------------------
      READ,NX,NZ
      READ,DELMX,DELTAZ
      READ,DELT1,ITER1
      READ,DELT2,ITER2
      READ,NC,NS
      READ,XSTART,TSTART,XPRINT,TPRINT,NPPAGE
      READ,LTYPE,OPTION
      PRINT 1001
 1001 FORMAT( 1H1,10X,'INPUT DATA',///,
     1   1X,'NX,NZ,    DELMX,DELTAZ,    DELT1,ITER1')
      PRINT,NX,NZ,DELMX,DELTAZ,DELT1,ITER1
      PRINT 1002
 1002 FORMAT(//,1X,'DELT2,ITER2,    NC,NS,    XSTART,TSTART')
      PRINT, DELT2,ITER2,NC,NS,XSTART,TSTART
      PRINT 1003
 1003 FORMAT(//,1X,'XPRINT,TPRINT,NPPAGE,    LTYPE,OPTION')
      PRINT, XPRINT,TPRINT,NPPAGE,LTYPE,CPTION
      IF(SET.EQ.1.) GO TO 500
      DO 375 Z=1,NZ
      DO 375 X=1,NX
      C(X,Z,T)=C(X,Z,TNEXT)
      C(X,Z,TNEXT)=0.0
  375 CONTINUE
  500 CONTINUE
      NXM1=NX-1
      NXM2=NX-2
      NZM1=NZ-1
```

```
      NZM2=NZ-2
      DELTAT=DELT1*86400.
      DELTAX=DELMX*5280.
      XSF=XSTART*5280.
      ITER=ITER1
      TIME=TIME+TSTART*86400.
      COUNT=(TSTART/DELTAT+0.01)
      COUNT=COUNT-(COUNT/TPRINT)*TPRINT
      READ,((VX(X,Z),X=1,NX),Z=1,NZ)
      PRINT 2
    2 FORMAT(//, 1X, '((VX(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((VX(X,Z),X=1,NX),Z=1,NZ)
      READ,((EX(X,Z),X=1,NX),Z=1,NZ)
      PRINT 3
    3 FORMAT(//, 1X, '((EX(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((EX(X,Z),X=1,NX),Z=1,NZ)
      READ,(W(X),X=1,NX)
      PRINT 4
    4 FORMAT(//, 1X, '(W(X),X=1,NX)')
      PRINT, (W(X),X=1,NX)
      READ,(KDAYS(X),X=1,NX)
      PRINT 5
    5 FORMAT(//, 1X, '(KDAYS(X),X=1,NX)')
      PRINT, (KDAYS(X),X=1,NX)
      READ,((DEMAND(X,Z),X=1,NX),Z=1,NZ)
      PRINT 6
    6 FORMAT(//, 1X, '((DEMAND(X,Z),X=1,NX),Z=1,NZ)')
      PRINT, ((DEMAND(X,Z),X=1,NX),Z=1,NZ)
      IF(NC.EQ.0) GO TO 24
      DO 10 I=1,NC
```

```
      READ,INSECT(I)
      PRINT 7
    7 FORMAT(//, 1X, 'INSECT(I)')
      PRINT,INSECT(I)
      READ, (CONCIN(I,J), J=1,NZ)
      PRINT 8
    8 FORMAT(//, 1X, '(CONCIN(I,J),J=1,NZ)')
      PRINT,(CONCIN(I,J),J=1,NZ)
   10 CONTINUE
      DO 15 I=1,NC
      DO 15 Z=1,NZ
      C(INSECT(I),Z,T)=CONCIN(I,Z) +C(INSECT(I),Z,T)
   15 CONTINUE
C
C-----------------------------------------------------------------
C          DETERMINE LOCATIONS OF UPPERMCST AND LOWERMOST
C          PEAK CONCENTRATIONS
C-----------------------------------------------------------------
      AMAX1=0.0
      PEAKU=0.0
      AMAX2=0.0
      PEAKL=0.0
      UPPER=XSF
      LOWER=XSF
      MSU=1
      MSL=NX
      DO 17 I=1,NC
      DO 16 J=1,NZ
      IF(CONCIN(I,J).LT.PEAKU) GC TO 16
      PEAKU=CONCIN(I,J)
   16 CONTINUE
```

```
      IF(PEAKU.LE.AMAX1) GO TO 20
      AMAX1=PEAKU
      MSU=INSECT(I)
      UPPER=(MSU-1)*DELTAX +XSF
   17 CONTINUE
   20 CONTINUE
      DO 22 II=1,NC
      I=NC+1-II
      DO 21 J=1,NZ
      IF(CONCIN(I,J).LT.PEAKL) GO TO 21
      PEAKL=CONCIN(I,J)
   21 CONTINUE
      IF(PEAKL.LE.AMAX2) GO TO 24
      AMAX2=PEAKL
      MSL=INSECT(I)
      LOWER=(MSL-1)*DELTAX +XSF
   22 CONTINUE
   24 CONTINUE
C
C-----------------------------------------------------------------------
C          DETERMINE MAXIMUM AND MINIMUM VALUES FOR INPUT DATA
C-----------------------------------------------------------------------
      DO 25 X=1,NX
      KD(X)=KDAYS(X)/86400.
      IF(W(X).GT.WMAX)WMAX=W(X)
      IF(W(X).LT.WMIN)WMIN=W(X)
      IF(KD(X).GT.KDMAX)KDMAX=KD(X)
      IF(KD(X).LT.KDMIN)KDMIN=KD(X)
      DO 25 Z=1,NZ
```

```
      ABSVX=ABS(VX(X,Z))
      IF(ABSVX.GT.VXMAX)VXMAX=ABSVX
      IF(ABSVX.LT.VXMIN)VXMIN=ABSVX
      IF(EX(X,Z).GT.EXMAX)EXMAX=EX(X,Z)
      IF(EX(X,Z).LT.EXMIN)EXMIN=EX(X,Z)
      IF(DEMAND(X,Z).GT.DEMAX)DEMAX=DEMAND(X,Z)
      IF(DEMAND(X,Z).LT.DEMIN)DEMIN=DEMAND(X,Z)
   25 CONTINUE
      IF(NZ.EQ.1)GO TO 200
C
C-----------------------------------------------------------------------
C           READ AND PRINT DATA FOR VERTICAL DIRECTION
C-----------------------------------------------------------------------
      READ,((VZ(X,Z),X=1,NX),Z=1,NZ)
      PRINT 30
   30 FORMAT(//, 1X, '((VZ((X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((VZ(X,Z),X=1,NX),Z=1,NZ)
      READ,((EZ(X,Z),X=1,NX),Z=1,NZ)
      PRINT 32
   32 FORMAT(//, 1X, '((EZ(X,Z),X=1,NX),Z=1,NZ)')
      PRINT, ((EZ(X,Z),X=1,NX),Z=1,NZ)
C
C-----------------------------------------------------------------------
C           DETERMINE MAXIMUM AND MINIMUM VALUES FOR INPUT DATA
C-----------------------------------------------------------------------
      DO 120 X=1,NX
      DO 120 Z=1,NZ
      ABSVZ=ABS(VZ(X,Z))
      IF(ABSVZ.GT.VZMAX)VZMAX=ABSVZ
      IF(ABSVZ.LT.VZMIN)VZMIN=ABSVZ
```

```
      IF(EZ(X,Z).GT.EZMAX)EZMAX=EZ(X,Z)
      IF(EZ(X,Z).LT.EZMIN)EZMIN=EZ(X,Z)
  120 CONTINUE
  200 CONTINUE
C
C-----------------------------------------------------------------------
C            READ AND PRINT FORMATS FOR OUTPUT
C-----------------------------------------------------------------------
      READ 205, (FMT1(I), I=1,40)
      PRINT 202
  202 FORMAT(//,1X, 'FORMAT FOR OUTPUT')
      PRINT 205, (FMT1(I),I=1,40)
  205 FORMAT (20A4,/,20A4)
      READ 205,(FMT2(I),I=1,40)
      PRINT 205, (FMT2(I), I=1,40)
      READ 207,(FMT3(I), I=1,20)
      PRINT 207, (FMT3(I), I=1,20)
  207 FORMAT(20A4)
C
C-----------------------------------------------------------------------
C            READ AND PRINT VELOCITIES AND DISPERSION COEFFICIENTS FOR
C            NEXT TIME STEP
C-----------------------------------------------------------------------
      READ,((VXNEXT(X,Z),X=1,NX),Z=1,NZ)
      PRINT 210
  210 FORMAT(//,1X,'((VXNEXT(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((VXNEXT(X,Z),X=1,NX),Z=1,NZ)
      READ,((EXNEXT(X,Z),X=1,NX),Z=1,NZ)
      PRINT 215
  215 FORMAT(//,1X,'((EXNEXT(X,Z),X=1,NX),Z=1,NZ)')
```

```
      PRINT,((EXNEXT(X,Z),X=1,NX),Z=1,NZ)
      IF(NZ.EQ.1) GO TO 250
      READ,((VZNEXT(X,Z),X=1,NX),Z=1,NZ)
      PRINT 220
  220 FORMAT(//,1X,'((VZNEXT(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((VZNEXT(X,Z),X=1,NX),Z=1,NZ)
      READ,((EZNEXT(X,Z),X=1,NX),Z=1,NZ)
      PRINT 225
  225 FORMAT(//,1X,'((EZNEXT(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((EZNEXT(X,Z),X=1,NX),Z=1,NZ)
  250 CONTINUE
      PRINT 259
  259 FORMAT(1H1)
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
1203        SUBROUTINE STABLI
      C
      C             *****************************
      C             ***** SUBROUTINE STABLI *****
      C             *****************************
      C
      C          THE RELATIONSHIP BETWEEN DISTANCE INCREMENTS,TIME INCREMENTS,
      C     DISPERSION COEFFICIENTS, AND VELOCITIES IS NEEDED TO DETERMINE
      C     THE STABILITY OF THE IMPLICIT FINITE-DIFFERENCE PROCEDURE.  THIS
      C     SUBROUTINE CALCULATES THE MAXIMUM ALLOWABLE INCREMENTS FOR TIME
      C     AND DISTANCE WHICH GENERALLY GUARANTEE STABILITY.
      C-----------------------------------------------------------------------
1204        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
1205        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1206        COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CCNCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1207        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1208        COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
           1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
           2               G(51),D(51),V(51)
1209        STABLX = DELTAX
1210        STABLZ = DELTAZ
1211        STABLT = DELTAT
```

```
C
C-----------------------------------------------------------------------
C           CALCULATE STABILITY CRITERIA FOR ONE DIMENSIONAL CASE
C-----------------------------------------------------------------------
      IF(NZ.NE.1) GO TO 500
      IF(EXMIN.GT.0.0.AND.VXMAX.GT.0.0) STABLX=2.*EXMIN/VXMAX
      TERM=2.*EXMAX+DELTAX**2.*KDMAX
      IF(TERM.NE.0.0) STABLT=(DELTAX**2.)/TERM*2.
      R1=DELTAT/(DELTAX**2.)
      R2=0.
      GO TO 800
  500 CONTINUE
C
C-----------------------------------------------------------------------
C           CALCULATE STABILITY CRITERIA FCR TWO DIMENSIONAL CASE
C-----------------------------------------------------------------------
      IF(EXMIN.GT.0.0.AND.VXMAX.GT.0.0) STABLX=2.*EXMIN/VXMAX
      IF(EZMIN.GT.0.0.AND.VZMAX.GT.0.0) STABLZ=2.*EZMIN/VZMAX
      TERM=2.*(EXMAX*DELTAZ**2.+EZMAX*DELTAX**2.)+KDMAX*(DELTAX**2)*
     $  (DELTAZ**2)
      IF(TERM.NE.0.0) STABLT=(DELTAZ**2.)*(DELTAX**2.)/TERM  *2.
      R1=DELTAT/(DELTAX**2.)
      R2=DELTAT/(DELTAZ**2.)
  800 CONTINUE
      RETURN
      END
```

```
     C-----------------------------------------------------------------------
1229       SUBROUTINE ONEDIM
     C
     C           ****************************
     C           ***** SUBRCUTINE ONEDIM *****
     C           ****************************
     C
     C          THIS SUBPROGRAM CALLS FCR THE APPROPRIATE ROUTINES TO SOLVE
     C     FOR THE CONCENTRATION PROFILE IN AN ESTUARY WHERE ONE-DIMENSIONAL
     C     BEHAVIOR CAN BE ASSUMED.  IN ADDITION, WIDTH OF THE ESTUARY CAN
     C     VARY THROUGHOUT.
     C
     C          THE SOLUTION MATRIX IS SET UP ACCORDING TO A CRANK-NICOLSON
     C     IMPLICIT SCHEME IN SUBROUTINE ARRAY3 AND THIS TRIDIAGONAL MATRIX
     C     IS SOLVED BY SUBROUTINE TRIDAG.
     C
     C-----------------------------------------------------------------------
1230       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
1231       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1232       COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
          1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
          2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1233       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
          1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
          2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
          3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
          4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
          5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1234       COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
          1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
          2               G(51),D(51),V(51)
```

```
      DO 300 IT=1,ITER
      TIME=TIME+DELTAT
      CALL ARRAY3
      CALL IBOUND
      Z=1
      DO 250 X=1,NX
      IF(C(X,Z,TNEXT).LT.0.0) C(X,Z,TNEXT)=0.0
      C(X,Z,T)=C(X,Z,TNEXT)
      C(X,Z,TNEXT)=0.0
  250 CONTINUE
      COUNT=COUNT+1
      IF(COUNT.GE.TPRINT) GO TO 292
      GO TO 300
C
C-----------------------------------------------------------------------
C              PRINT CONCENTRATICNS AT APPROPRIATE INTERVALS
C-----------------------------------------------------------------------
  292 CONTINUE
      COUNT=0
      IPAGE=IPAGE+1
      IF(IPAGE.GE.NPPAGE) GO TO 295
      GO TO 297
  295 PRINT 296
  296 FORMAT(1H1)
      IPAGE = 0
  297 CALL PRINT2
  300 CONTINUE
      RETURN
      END
```

```
C-----------------------------------------------------------------------
      SUBROUTINE TWODIM
C
C             *****************************
C             ***** SUBROUTINE TWODIM *****
C             *****************************
C
C          THIS SUBPROGRAM CALLS FOR THE APPROPRIATE ROUTINES TO SOLVE
C     FOR THE CONCENTRATION PROFILE IN AN ESTUARY WHERE TWO DIMENSIONAL
C     BEHAVIOR CAN BE ASSUMED.  IN ADDITION, WIDTH OF THE ESTUARY CAN
C     BE VARIED THROUGHOUT.
C
C          THE SOLUTION MATRIX IS SET UP ACCORDING TO A CRANK-NICOLSON
C     IMPLICIT SCHEME WHICH IS SOLVED BY AN IMPLICIT,ALTERNATING
C     DIRECTION METHOD.  SUBROUTINE ARRAY1 SETS UP THE MATRIX FOR THE
C     FIRST HALF OF THE SOLUTION AND SUBROUTINE ARRAY2 SETS UP THE
C     MATRIX FOR THE SECOND HALF.  SUBROUTINE TRIDAG SOLVES BOTH OF
C     THESE MATRICES.
C
C-----------------------------------------------------------------------
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
```

```
      COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
     1                VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
     2                G(51),D(51),V(51)
      DO 300 IT=1,ITER
      TIME=TIME+DELTAT
      CALL ARRAY1
      CALL ARRAY2
      CALL IBOUND
      DO 250 Z=1,NZ
      DO 250 X=1,NX
      IF(C(X,Z,TNEXT).LT.0.0) C(X,Z,TNEXT)=0.0
      C(X,Z,T)=C(X,Z,TNEXT)
      C(X,Z,TNEXT)=0.0
  250 CONTINUE
      COUNT=COUNT+1
      IF(COUNT.GE.TPRINT) GO TO 292
      GO TO 300
C
C---------------------------------------------------------------------
C           PRINT CONCENTRATICNS AT APPROPRIATE INTERVALS
C---------------------------------------------------------------------
  292 CONTINUE
      COUNT=0
      IPAGE=IPAGE+1
      IF(IPAGE.GE.NPPAGE) GO TO 295
      GO TO 297
  295 PRINT 296
  296 FORMAT(1H1)
```

```
      IPAGE = 0
  297 CALL PRINT2
  300 CONTINUE
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
1292        SUBROUTINE ARRAY1
      C
      C              ******************************
      C              ***** SUBRCUTINE ARRAY1 ****
      C              ******************************
      C
      C
      C-----------------------------------------------------------------------
1293        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
1294        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1295        COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CCNCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1296        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTICN
1297        COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
           1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
           2               G(51),D(51),V(51)
      C
1298        ZUP=UPPER
1299        ZLOW=LOWER
1300        ZSF=XSF
1301        FOURX=4.*DELTAX
1302        FORXSQ=4.*DELTAX**2
1303        FOURZ=4.*DELTAZ
1304        FORZSQ=4.*DELTAZ**2
```

```
             C
             C-----------------------------------------------------------------------
             C             SET UP THE ARRAY ACCORDING TO A CRANK-NICOLSON
             C             IMPLICIT SCHEME
             C-----------------------------------------------------------------------
1305                DO 1005 Z=2,NZM1
1306                DO 100 X=2,NXM1
1307                A(X)=(DELTAT/W(X))*(-W(X)*VXNEXT(X,Z)/FOURX-(W(X)*EXNEXT(X,Z)
                  $       +W(X-1)*EXNEXT(X-1,Z))/FORXSQ)
1308                B(X)=1.+(DELTAT/W(X))*((W(X+1)*EXNEXT(X+1,Z)+2.*W(X)*EXNEXT(X,Z)
                  $          +W(X-1)*EXNEXT(X-1,Z))/FORXSQ)
1309                G(X)=(DELTAT/W(X))*(W(X)*VXNEXT(X,Z)/FOURX-(W(X+1)*EXNEXT(X+1,Z)
                  $       +W(X)*EXNEXT(X,Z))/FORXSQ)
1310                D(X)=C(X,Z,T)+(DELTAT/W(X))*(-(W(X)*VX(X,Z)/FOURX)*(C(X+1,Z,T)
                  1       -C(X-1,Z,T))+((W(X+1)*EX(X+1,Z)+W(X)*EX(X,Z))/FORXSQ)
                  2       *(C(X+1,Z,T)-C(X,Z,T))-((W(X   )*EX(X,Z)+W(X-1)*EX(X-1,Z))
                  3       /FORXSQ)*(C(X,Z,T)-C(X-1,Z,T)))
1311                D(X)=D(X)+(DELTAT/W(X))*(-2.*(W(X)*VZ(X,Z)/FOURZ)*(C(X,Z+1,T)
                  1       -C(X,Z-1,T))+2.*((W(X)*EZ(X,Z+1)+W(X)*EZ(X,Z))/FORZSQ)
                  2       *(C(X,Z+1,T)-C(X   ,Z,T))-2.*((W(X)*EZ(X,Z)+W(X)*EZ(X,Z-1))
                  3       /FORZSQ)*(C(X,Z,T)-C(X,Z-1,T)))
1312           100 CONTINUE
1313                NFLAG=1
             C
             C-----------------------------------------------------------------------
             C             EVALUATE BOUNDARY VALUES BY EXPONENTIAL EXTRAPOLATION OR BY
             C             CONSTANT SLOPE EXTRAPOLATION CEPENDING ON VALUE OF OPTION
             C-----------------------------------------------------------------------
```

```
      GO TO (101,2C2,101,202), OPTION
  101 CONTINUE
      VX1=VXNEXT(1,Z)
      XX=-UPPER+DELTAX+XSF+VX1*TSTART*86400.
      EX1=EXNEXT(1,Z)
      C2=4.*EX1*TIME
      EXPON=( 2.*XX*DELTAX-(DELTAX*DELTAX)-2.*DELTAX*VX1*TIME)/C2
      P=EXP(EXPON)
      B(2)=B(2)+A(2)*P
      CSTAR(1,Z)=CSTAR(2,Z)*P
  150 CONTINUE
      VXNX=VXNEXT(NX,Z)
      XX=(NX-2)*DELTAX-LOWER+XSF+VXNX*TSTART*86400.
      EXNX=EXNEXT(NX,Z)
      C2=4.*EXNX*TIME
      EXPON=(-2.*XX*DELTAX-(CELTAX*DELTAX)+2.*DELTAX*VXNX*TIME)/C2
      P=EXP(EXPON)
      B(NXM1)=B(NXM1)+G(NXM1)*P
      CSTAR(NX,Z)=CSTAR(NXM1,Z)*P
      GO TO 1000
  202 CONTINUE
      B(2)=B(2)+2.*A(2)
      G(2)=G(2)-A(2)
      A(NX-1)=A(NX-1)-G(NX-1)
      B(NX-1)=B(NX-1)+2.*A(NX-1)
      CSTAR(1,Z)=2.*CSTAR(2,Z)-CSTAR(3,Z)
      IF(CSTAR(1,Z).LT.0.) CSTAR(1,Z)=0.
      CSTAR(NX,Z)=2.*CSTAR(NXM1,Z)-CSTAR(NXM2,Z)
      IF(CSTAR(NX,Z).LT.0.) CSTAR(NX,Z)=0.
 1000 CONTINUE
```

```
      IF(NFLAG.GT.1) GO TO 1005
      NFLAG=2
C
C-----------------------------------------------------------------------
C            SOLVE TRIDIAGONAL MATRIX
C-----------------------------------------------------------------------
      CALL TRIDAG(2,NXM1)
C
C-----------------------------------------------------------------------
C            STORE HALF-STEP SCLUTION IN CSTAR
C-----------------------------------------------------------------------
      DO 1004 X=2,NXM1
      CSTAR(X,Z)=V(X)
 1004 CONTINUE
      GO TO (101,202,101,202),CPTION
 1005 CONTINUE
      RETURN
      END
```

```
     C-----------------------------------------------------------------------
1354       SUBROUTINE ARRAY2
     C
     C             *****************************
     C             ***** SUBROUTINE ARRAY2 ****
     C             *****************************
     C
     C
     C-----------------------------------------------------------------------
1355       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
1356       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1357       COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
          1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
          2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1358       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
          1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
          2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
          3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
          4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
          5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1359       COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
          1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
          2               G(51),D(51),V(51)
1360       ZUP=UPPER
1361       ZLOW=LOWER
1362       ZSF=XSF
1363       FOURX=4.*DELTAX
1364       FORXSQ=4.*DELTAX**2
1365       FOURZ=4.*DELTAZ
1366       FORZSQ=4.*DELTAZ**2
```

```
      C
      C-----------------------------------------------------------------
      C          SET UP THE ARRAY ACCORDING TO A CRANK-NICOLSON
      C          IMPLICIT SCHEME
      C-----------------------------------------------------------------
1367        DO 1005 X=2,NXM1
1368        DO 100 Z=2,NZM1
1369        A(Z)=(DELTAT/W(X))*(-W(X)*VZNEXT(X,Z)/FOURZ-(W(X)*EZNEXT(X,Z)
           $     +W(X)*EZNEXT(X,Z-1))/FORZSQ)
1370        B(Z)=1.+(DELTAT/W(X))*(W(X)*EZNEXT(X,Z+1)+2.*W(X)*EZNEXT(X,Z)
           $     +W(X)*EZNEXT(X,Z-1))/FORZSQ
1371        G(Z)=(DELTAT/W(X))*(W(X)*VZNEXT(X,Z)/FOURZ-(W(X)*EZNEXT(X,Z+1)
           $     +W(X)*EZNEXT(X,Z))/FORZSQ)
1372        D(Z)=C(X,Z,T)+(DELTAT/W(X))*(-(W(X)*VXNEXT(X,Z)/FOURX)*(CSTAR(
           1     X+1,Z)-CSTAR(X-1,Z))-(W(X)*VX(X,Z)/FOURX)*(C(X+1,Z,T)
           2     -C(X-1,Z,T))-(W(X)*VZ(X,Z)/FOURZ)*(C(X,Z+1,T)-C(X,Z-1,T))
           3     +((W(X+1)*EXNEXT(X+1,Z)+W(X)*EXNEXT(X,Z))/FORXSQ)*(CSTAR(
           4     X+1,Z)-CSTAR(X,Z))+((W(X+1)*EX(X+1,Z)+W(X)*EX(X,Z))/FORXSQ)
           5     *(C(X+1,Z,T)-C(X,Z,T)))
1373        D(Z)=D(Z)+(DELTAT/W(X))*(-((W(X)*EXNEXT(X,Z)+W(X-1)*EXNEXT(X-1,Z))
           1     /FORXSQ)*(CSTAR(X,Z)-CSTAR(X-1,Z))-((W(X)*EX(X,Z)+W(X-1)
           2     *EX(X-1,Z))/FORXSQ)*(C(X,Z,T)-C(X-1,Z,T))+((W(X)*EZ(X,Z+1)
           3     +W(X)*EZ(X,Z))/FORZSQ)*(C(X,Z+1,T)-C(X,Z,T))-((W(X)*EZ(X,Z)
           4     +W(X)*EZ(X,Z-1))/FORZSQ)*(C(X,Z,T)-C(X,Z-1,T)))
1374    100 CONTINUE
      C
      C-----------------------------------------------------------------
      C          EVALUATE BOUNDARY VALUES BY EXPONENTIAL EXTRAPOLATION OR BY
      C          CONSTANT SLOPE EXTRAPOLATICN DEPENDING ON VALUE OF OPTION
```

```
C-----------------------------------------------------------------------
      GO TO (303,303,101,202), CPTION
  101 CONTINUE
      VZ1=VZNEXT(X,1)
      ZZ=-ZUP+DELTAZ+ZSF+VZ1*TSTART*86400.
      EZ1=EZNEXT(X,1)
      C2=4.*EZ1*TIME
      EXPON=( 2.*ZZ*DELTAZ-(DELTAZ*DELTAZ)-2.*DELTAZ*VZ1*TIME)/C2
      B(2)=B(2)+A(2)*EXP(EXPCN)
  150 CONTINUE
      VZNZ=VZNEXT(X,NZ)
      ZZ=(NZ-2)*DELTAZ-ZLOW+ZSF+VZNZ*TSTART*86400.
      EZNZ=EZNEXT(X,NZ)
      C2=4.*EZNZ*TIME
      EXPON=(-2.*ZZ*DELTAZ-(DELTAZ*DELTAZ)+2.*DELTAZ*VZNZ*TIME)/C2
      B(NZ-1)=B(NZ-1)+G(NZ-1)*EXP(EXPON)
      GO TO 1000
  202 CONTINUE
      B(2)=B(2)+2.*A(2)
      G(2)=G(2)-A(2)
      A(NZ-1)=A(NZ-1)-G(NZ-1)
      B(NZ-1)=B(NZ-1)+2.*A(NZ-1)
      GO TO 1000
  303 B(2)=B(2)+A(2)
      B(NZM1)=B(NZM1)+G(NZM1)
 1000 CONTINUE
```

```
C
C-----------------------------------------------------------------
C         SOLVE TRIDIAGONAL MATRIX
C-----------------------------------------------------------------
      CALL TRIDAG(2,NZM1)
C
C-----------------------------------------------------------------
C         SUBTRACT DECAY AND OTHER DEMANDS AND
C         PLACE FINAL SOLUTION IN C(X,Z,T+1)
C-----------------------------------------------------------------
      DO 1004 Z=2,NZM1
      C(X,Z,T+1)=V(Z)-DELTAT*(KD(X)/2.*(V(Z)+C(X,Z,T)))-DEMAND(X,Z)
 1004 CONTINUE
 1005 CONTINUE
      RETURN
      END
```

```
     C--------------------------------------------------------------------------
1407       SUBROUTINE ARRAY3
     C
     C             *****************************
     C             ***** SUBRCUTINE ARRAY3 ****
     C             *****************************
     C
     C
     C--------------------------------------------------------------------------
1408       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
1409       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1410       COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
          1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
          2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1411       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
          1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
          2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
          3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
          4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
          5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTICN
1412       COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
          1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
          2               G(51),D(51),V(51)
1413       FOURX=4.*DELTAX
1414       FORXSQ=4.*CELTAX**2
1415       Z=1
     C
     C--------------------------------------------------------------------------
     C           SET UP THE ARRAY ACCORDING TO A CRANK-NICOLSON
     C           IMPLICIT SCHEME
     C--------------------------------------------------------------------------
1416       DO 100 X=2,NXM1
1417       A(X)=(DELTAT/W(X))*(-W(X)*VXNEXT(X,Z)/FOURX-(W(X)*EXNEXT(X,Z)
          $      +W(X-1)*EXNEXT(X-1,Z))/FORXSQ)
```

```
      B(X)=1.+(DELTAT/W(X))*((W(X+1)*EXNEXT(X+1,Z)+2.*W(X)*EXNEXT(X,Z)
     $       +W(X-1)*EXNEXT(X-1,Z))/FORXSQ+KD(X)/2.*W(X))
      G(X)=(DELTAT/W(X))*(W(X)*VXNEXT(X,Z)/FOURX-(W(X+1)*EXNEXT(X+1,Z)
     $     +W(X)*EXNEXT(X,Z))/FORXSQ)
      D(X)=C(X,Z,T)+(DELTAT/W(X))*(-(W(X)*VX(X,Z)/FOURX)*(C(X+1,Z,T)
     1     -C(X-1,Z,T))+((W(X+1)*EX(X+1,Z)+W(X)*EX(X,Z))/FORXSQ)
     2     *(C(X+1,Z,T)-C(X,Z,T))-((W(X  )*EX(X,Z)+W(X-1)*EX(X-1,Z))
     3     /FORXSQ)*(C(X,Z,T)-C(X-1,Z,T))-(KD(X)/2.)*C(X,Z,T)*W(X))
  100 CONTINUE
C
C-----------------------------------------------------------------------
C          EVALUATE BOUNDARY VALUES BY EXPONENTIAL EXTRAPOLATION OR BY
C          CONSTANT SLOPE EXTRAPOLATION DEPENDING ON VALUE OF OPTION
C-----------------------------------------------------------------------
      GO TO (101,202,101,202), OPTION
  101 CONTINUE
      VX1=VXNEXT(1,Z)
      XX=-UPPER+DELTAX+XSF+VX1*TSTART*86400.
      EX1=EXNEXT(1,Z)
      C2=4.*EX1*TIME
      EXPON=( 2.*XX*DELTAX-(DELTAX*DELTAX)-2.*DELTAX*VX1*TIME)/C2
      B(2)=B(2)+A(2)*EXP(EXPCN)
  150 CONTINUE
      VXNX=VXNEXT(NX,Z)
```

```
      XX=(NX-2)*DELTAX-LOWER+XSF+VXNX*TSTART*86400.
      EXNX=EXNEXT(NX,Z)
      C2=4.*EXNX*TIME
      EXPON=(-2.*XX*DELTAX-(DELTAX*DELTAX)+2.*DELTAX*VXNX*TIME)/C2
      B(NX-1)=B(NX-1)+G(NX-1)*EXP(EXPON)
      GO TO 1000
  202 CONTINUE
      B(2)=B(2)+2.*A(2)
      G(2)=G(2)-A(2)
      A(NX-1)=A(NX-1)-G(NX-1)
      B(NX-1)=B(NX-1)+2.*A(NX-1)
 1000 CONTINUE
C
C-----------------------------------------------------------------------
C             SOLVE TRIDIAGONAL MATRIX
C-----------------------------------------------------------------------
      CALL TRIDAG(2,NXM1)
C
C-----------------------------------------------------------------------
C             SUBTRACT OTHER DEMANDS AND PLACE FINAL SOLUTION IN C(X,Z,T+1)
C-----------------------------------------------------------------------
      DO 1004 X=2,NXM1
      C(X,Z,T+1)=V(X)-DEMAND(X,Z)
 1004 CONTINUE
      RETURN
      END
```

```
     C------------------------------------------------------------------------
1450       SUBROUTINE TRIDAG(IF,L)
     C
     C              ****************************
     C              ***** SUBROUTINE TRIDAG ****
     C              ****************************
     C
     C         THIS SUBROUTINE SOLVES A SYSTEM OF LINEAR SIMULTANEOUS
     C         EQUATIONS HAVING A TRIDIAGONAL COEFFICIENT MATRIX.
     C         THE EQUATIONS ARE NUMBERED FROM IF THROUGH L, AND THEIR
     C         SUB-DIAGONAL,DIAGONAL,AND SUPER-DIAGONAL COEFFICIENTS
     C         ARE STORED IN THE ARRAYS A,B,AND G.  THE COMPUTED
     C         SOLUTION VECTOR V(IF) TO V(L) IS STORED IN THE ARRAY V.
     C
     C         THIS SUBPROGRAM IS A MODIFIED VERSION OF A PROGRAM FOUND IN THE
     C         FOLLOWING TEXT**APPLIED NUMERICAL METHODS**BY CARNAHAN,LUTHER,
     C         AND WILKES, JOHN WILEY&SONS,INC.,1969,P446.
     C
     C------------------------------------------------------------------------
1451       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
1452       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1453       COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
          1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
          2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1454       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
          1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
          2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
          3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
          4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
          5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
```

```
1455        COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
           1              VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
           2              G(51),D(51),V(51)
1456        DIMENSION BETA(51),GAMMA(51)
      C
      C-----------------------------------------------------------------
      C        COMPUTE THE INTERMEDIATE ARRAYS BETA AND GAMMA
      C-----------------------------------------------------------------
1457        BETA(IF)=B(IF)
1458        GAMMA(IF)=D(IF)/BETA(IF)
1459        IFP1=IF+1
1460        DO 1 I=IFP1,L
1461        BETA(I)=B(I)-A(I)*G(I-1)/BETA(I-1)
1462      1 GAMMA(I)=(D(I)-A(I)*GAMMA(I-1))/BETA(I)
      C
      C-----------------------------------------------------------------
      C        COMPUTE FINAL SOLUTICN VECTOR V
      C-----------------------------------------------------------------
1463        V(L)=GAMMA(L)
1464        LAST=L-IF
1465        DO 2 K=1,LAST
1466        I=L-K
1467      2 V(I)=GAMMA(I)-G(I)*V(I+1)/BETA(I)
1468        RETURN
1469        END
```

```
        C----------------------------------------------------------------------
1470          SUBROUTINE IBOUND
        C
        C             ****************************
        C             ***** SUBRCUTINE IBOUND ****
        C             ****************************
        C
        C           THIS FORM OF SUBRCUTINE BCUND EXTRAPOLATES THE CONCENTRATIONS
        C     AT THE BOUNDARIES BY USING AN EXPONENTIAL EQUATION APPROPRIATE TO
        C     CONCENTRATION PROFILES IN AN ESTUARY OR BY A CONSTANT SLOPE
        C     EXTRAPOLATION.  THE TYPE OF EXTRAPOLATION AT THE BOUNDARIES
        C     DEPENDS ON THE VALUE FOR THE VARIABLE OPTION.
        C
        C----------------------------------------------------------------------
1471          INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
1472          REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1473          COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
             1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
             2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1474          COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
             1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
             2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
             3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
             4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
             5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1475          COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
             1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
             2               G(51),D(51),V(51)
```

```
      IF(NZ.EQ.1) NZM1=1
      ZUP=UPPER
      ZLOW=LOWER
      ZSF=XSF
      GO TO (1001,1002,1003,1004,1005,1002,1004), OPTION
C
C-----------------------------------------------------------------------
C         EXPONENTIAL EXTRAPOLATION
C-----------------------------------------------------------------------
 1003 CONTINUE
      IF(NZ.EQ.1) GO TO 1001
      DO 140 X=2,NXM1
      IF(C(X,2,T+1).LE.0.0) GO TO 140
      VZ1=VZNEXT(X,1)
      ZZ=-ZUP+DELTAZ+ZSF+VZ1*TSTART*86400.
      EZ1=EZNEXT(X,1)
      C2=4.*EZ1*TIME
      EXPON=( 2.*ZZ*DELTAZ-(DELTAZ*DELTAZ)-2.*DELTAZ*VZ1*TIME)/C2
      C(X,1,T+1)=C(X,2,T+1)*EXP(EXPON)
  140 CONTINUE
      DO 190 X=2,NXM1
      IF(C(X,NZM1,T+1).LE.0.0) GO TO 190
      VZNZ=VZNEXT(X,NZ)
      ZZ=(NZ-2)*DELTAZ-ZLOW+ZSF+VZNZ*TSTART*86400.
      EZNZ=EZNEXT(X,NZ)
      C2=4.*EZNZ*TIME
      EXPON=(-2.*ZZ*DELTAZ-(DELTAZ*DELTAZ)+2.*DELTAZ*VZNZ*TIME)/C2
      C(X,NZ,T+1)=C(X,NZM1,T+1)*EXP(EXPON)
  190 CONTINUE
 1001 CONTINUE
      IF(LTYPE.EQ.1.AND.MSU.EQ.1) GO TC 350
```

```
      DO 340 Z=1,NZ
      IF(C(2,Z,T+1).LE.0.0) GO TO 340
      VX1=VXNEXT(1,Z)
      XX=-UPPER+DELTAX+XSF+VX1*TSTART*86400.
      EX1=EXNEXT(1,Z)
      C2=4.*EX1*TIME
      EXPON=( 2.*XX*DELTAX-(DELTAX*DELTAX)-2.*DELTAX*VX1*TIME)/C2
      C(1,Z,T+1)=C(2,Z,T+1)*EXP(EXPON)
  340 CONTINUE
  350 CONTINUE
      IF(LTYPE.EQ.1.AND.MSL.EQ.NZ) GO TO 400
      DO 390 Z=1,NZ
      IF(C(NX-1,Z,T+1).LE.0.0) GO TO 390
      VXNX=VXNEXT(NX,Z)
      XX=(NX-2)*DELTAX-LOWER+XSF+VXNX*TSTART*86400.
      EXNX=EXNEXT(NX,Z)
      C2=4.*EXNX*TIME
      EXPON=(-2.*XX*DELTAX-(CELTAX*DELTAX)+2.*DELTAX*VXNX*TIME)/C2
      C(NX,Z,T+1)=C(NX-1,Z,T+1)*EXP(EXPON)
  390 CONTINUE
  400 CONTINUE
      GO TO 2000
C
C-----------------------------------------------------------------------
C         CONSTANT SLOPE EXTRAPOLATION
C-----------------------------------------------------------------------
 1004 CONTINUE
      IF(NZ.EQ.1) GO TO 1002
```

```
      DO 540 X=2,NXM1
      C(X,1,T+1)=2.*C(X,2,T+1)-C(X,3,T+1)
      IF(C(X,1,T+1).LT.0.0) C(X,1,T+1)=0.0
      C(X,NZ,T+1)=2.*C(X,NZM1,T+1)-C(X,NZM2,T+1)
      IF(C(X,NZ,T+1).LT.0.0) C(X,NZ,T+1)=0.0
  540 CONTINUE
 1002 CONTINUE
      DO 590 Z=1,NZ
      C(1,Z,T+1)=2.*C(2,Z,T+1)-C(3,Z,T+1)
      IF(C(1,Z,T+1).LT.0.0) C(1,Z,T+1)=0.0
      C(NX,Z,T+1)=2.*C(NXM1,Z,T+1)-C(NXM2,Z,T+1)
      IF(C(NX,Z,T+1).LT.0.0) C(NX,Z,T+1)=0.0
  590 CONTINUE
      GO TO 2000
C
C-----------------------------------------------------------------------
C            THIS OPTION ALLOWS THE PROGRAM USER TO SUBSTITUTE HIS OWN
C            EXTRAPOLATION ROUTINE.  THE USER MUST BE CAREFUL TO CHANGE
C            THE APPROPRIATE ARRAY VALUES.
C-----------------------------------------------------------------------
 1005 CONTINUE
      OPTION=2
      GO TO 1002
 2000 CONTINUE
      IF(NZ.EQ.1) GO TO 2005
      IF(OPTION.EQ.3.OR.OPTION.EQ.4.OR.OPTION.EQ.5)  GO TO 2005
      DO 700 X=1,NX
      C(X,1,T+1)=C(X,2,T+1)
      C(X,NZ,T+1)=C(X,NZM1,T+1)
  700 CONTINUE
 2005 CONTINUE
C
```

```
C-----------------------------------------------------------------------
C          IF LTYPE EQUALS 1, SET CONCENTRATIONS AT SOURCE POINTS.
C          IF LTYPE EQUALS 3, THE USER CAN INCLUDE A SPECIAL SUBROUTINE
C          OR SET OF CALCULATIONS IN THIS PART OF THE PROGRAM.
C-----------------------------------------------------------------------
      IF(LTYPE.EQ.1.AND.NS.LE.2) GC TO 275
      IF(LTYPE.EQ.1.AND.NS.GT.2) GO TO 281
      GO TO 290
  275 CONTINUE
      DO 280 Z=1,NZ
      C(MSU,Z,T+1)=C(MSU,Z,T)
      C(MSL,Z,T+1)=C(MSL,Z,T)
  280 CONTINUE
      GO TO 290
  281 CONTINUE
      DO 285 I=1,NC
      DO 285 Z=1,NZ
      C(INSECT(I),Z,T+1)=CONCIN(I,Z)
  285 CONTINUE
  290 CONTINUE
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
1569        SUBROUTINE PRINT1
      C
      C              *****************************
      C              ***** SUBROUTINE PRINT1 *****
      C              *****************************
      C
      C          THIS SUBROUTINE CALCULATES THE CONVERSION VALUES FOR MANY OF
      C     THE INPUT PARAMETERS AND PRINTS OUT A SUMMARY OF THE INPUT DATA
      C
      C-----------------------------------------------------------------------
1570        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
1571        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1572        COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1573        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1574        COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
           1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
           2               G(51),D(51),V(51)
      C
      C-----------------------------------------------------------------------
      C         CALCULATE CONVERSION UNITS
      C-----------------------------------------------------------------------
```

```
      UNITS1 = 86400./5280.
      UNITS2 = 86400./(5280.*5280.)
      DELDAY = DELTAT/86400.
      SECS1=DELT1*86400.
      SECS2=DELT2*86400.
      DELMX=DELTAX/5280.
      DELMZ=DELTAZ/5280.
      TOTALX=(NX-1)*DELTAX
      TOTMX=TOTALX/5280.
      TOTALZ=(NZ-1)*DELTAZ
      TOTMZ=TOTALZ/5280.
      VX1=VXMAX*UNITS1
      VX2=VXMIN*UNITS1
      VZ1=VZMAX*UNITS1
      IF(VZMIN.EQ.1000000.) VZMIN=0.
      VZ2=VZMIN*UNITS1
      EX1=EXMAX*UNITS2
      EX2=EXMIN*UNITS2
      EZ1=EZMAX*UNITS2
      IF(EZMIN.EQ.1000000.) EZMIN=0.
      EZ2=EZMIN*UNITS2
      XKD1=KDMAX*86400.
      XKD2=KDMIN*86400.
      SXM=STABLX/5280.
      SZM=STABLZ/5280.
      STD=STABLT/86400.
C
C---------------------------------------------------------------------
C          PRINT DATA SUMMARY
C---------------------------------------------------------------------
```

```
      PRINT 500
      PRINT 505
      PRINT 507
      PRINT 510
      PRINT 507
      IF(NZ.EQ.1) PRINT 515
      IF(NZ.NE.1) PRINT 520
      PRINT 507
      IF(LTYPE.EQ.1) GO TO 7
      IF(LTYPE.EQ.2) GC TC 5
      PRINT 530
      GO TO 10
    5 PRINT 525
      GO TO 10
    7 PRINT 523
   10 CONTINUE
      PRINT 507
      PRINT 535
      PRINT 507
      PRINT 505
      PRINT 540, DELTAX, DELMX
      PRINT 545, DELTAZ,DELMZ
      PRINT 550, SECS1, DELT1
      PRINT 551, ITER1
      PRINT 552, SECS2, DELT2
      PRINT 553, ITER2
      PRINT 555,NX
      PRINT 560, NZ
      PRINT 565, VXMAX, VX1
      PRINT 570, VXMIN, VX2
```

```
      PRINT 572,VZMAX,VZ1
      PRINT 573,VZMIN,VZ2
      PRINT 575,EXMAX,EX1
      PRINT 580,EXMIN,EX2
      PRINT 585, EZMAX,EZ1
      PRINT 590, EZMIN,EZ2
      PRINT 600,WMAX
      PRINT 605,WMIN
      PRINT 610,XKD1,KDMAX
      PRINT 615,XKD2,KDMIN
      PRINT 620,DEMAX
      PRINT 630, NC
C
C-----------------------------------------------------------------------
C        XUPEAK AND XLPEAK ARE CALCULATED ASSUMING DISTANCES ARE
C     DECREASING IN THE DOWNSTREAM DIRECTION
C-----------------------------------------------------------------------
      XUPEAK = XSTART - (MSU-1)*DELMX
      XLPEAK = XSTART - (MSL-1)*DELMX
      PRINT 635, XUPEAK,MSU
      PRINT 637, XLPEAK,MSL
      PRINT 500
      PRINT 505
      PRINT 502
      PRINT 640
      PRINT 645,STABLX,SXM
      PRINT 650,STABLZ,SZM
      PRINT 655,STABLT,STD
      PRINT 660,R1
      PRINT 665, R2
```

```
      PRINT 505
      IF(NC.EQ.0) GO TO 100
      PRINT 500
      PRINT 700
      IF(NZ.EQ.1) GO TO 50
      DO 25 I=1,NC
C
C-----------------------------------------------------------------------
C           WHERE IS CALCULATED ASSUMING DISTANCES ARE DECREASING IN
C     THE DOWNSTREAM DIRECTION
C-----------------------------------------------------------------------
      WHERE=XSTART-(INSECT(I)-1)*DELMX
      PRINT 710, INSECT(I),WHERE,DELTAZ
      DO 25 Z=1,NZ
      PRINT 715,INSECT(I),Z,CONCIN(I,Z)
   25 CONTINUE
      GO TO 100
   50 DO 75 I=1,NC
      WHERE=XSTART-(INSECT(I)-1)*DELMX
      PRINT 705, INSECT(I), WHERE, CONCIN(I,1)
      PRINT 502
   75 CONTINUE
  100 CONTINUE
      PRINT 500
  500 FORMAT(1H1)
  502 FORMAT(/)
  505 FORMAT(T45, 38('*'))
  507 FORMAT(T45,'*',T82,'*')
  510 FORMAT(T45,'*',T56,'ESTUARY SIMULATION', T82, '*')
  515 FORMAT(T45, '*', T48, 'ONE DIMENSIONAL IMPLICIT METHOD',T82,'*')
```

```
520 FORMAT(T45, '*', T48, 'TWO DIMENSIONAL IMPLICIT METHOD',T82,'*')
523 FORMAT(T45, '*', T53, 'CONSTANT CONCENTRATION',T82,'*')
525 FORMAT(T45, '*', T54, 'INSTANTANEOUS RELEASE', T82,'*')
530 FORMAT(T45, '*', T56, 'CONTINUOUS LOADING', T82, '*')
535 FORMAT(T45, '*', T51, 'PROGRAMMER -- JONATHAN YOUNG', T82,'*')
540 FORMAT(   /, T53, 'X INCREMENT =', F7.0, ' FEET (', F8.5, ' MILES
   $)',/)
545 FORMAT(T53, 'Z INCREMENT =', F7.0, ' FEET (', E10.3, ' MILES )'/)
550 FORMAT(T42, 'INITIAL TIME INCREMENT =',F7.0, ' SECONDS (', F6.4,
   $  ' DAYS )'/)
551 FORMAT(T44,'NUMBER OF ITERATIONS = ',I5,/)
552 FORMAT( T42, 'REVISED TIME INCREMENT = ', F7.0, ' SECONDS(',F6.4,
   $  ' DAYS )'/)
553 FORMAT(T44,'NUMBER OF ITERATIONS = ',I5,/)
555 FORMAT(T37, 'NUMBER OF HORIZONTAL POINTS =', I4,/)
560 FORMAT(T39, 'NUMBER OF VERTICAL POINTS =',I4,/)
565 FORMAT(T37, 'MAXIMUM HORIZONTAL VELOCITY = ', F5.2, ' FEET/SECOND
   $(',F6.2, ' MILES/DAY )',/)
570 FORMAT(T37, 'MINIMUM HORIZONTAL VELOCITY = ', F5.2, ' FEET/SECCND
   $(',F6.2, ' MILES/DAY )',/)
572 FORMAT(T39, 'MAXIMUM VERTICAL VELOCITY =', E10.3, ' FEET/SECOND ('
   $ , E10.3, ' MILES/DAY )',/)
573 FORMAT(T39, 'MINIMUM VERTICAL VELOCITY =', E10.3, ' FEET/SECOND ('
   $ , E10.3, ' MILES/DAY )',/)
575 FORMAT(T35, 'MAXIMUM HCRIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
   $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
580 FORMAT(T35, 'MINIMUM HORIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
   $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
585 FORMAT(T37, 'MAXIMUM VERTICAL DISPERSION = ', F9.3, ' FEET SQUARED
   $ / SECOND ( ', E10.3, ' MILES SQUARED / DAY )',/)
```

```
1701    590 FORMAT(T37, 'MINIMUM VERTICAL DISPERSION = ', F9.3, ' FEET SQUARED
           $ / SECOND ( ', E10.3, ' MILES SQUARED / DAY )',/)
1702    600 FORMAT(T51, 'MAXIMUM WIDTH = ', F5.0, ' FEET',/)
1703    605 FORMAT(T51, 'MINIMUM WIDTH = ', F5.0, ' FEET',/)
1704    610 FORMAT(T46, 'MAXIMUM DECAY RATE = ', F5.3, ' PER DAY ( ', E10.3,
           $ ' PER SECOND )', /)
1705    615 FORMAT(T46, 'MINIMUM DECAY RATE = ', F5.3, ' PER DAY ( ', E10.3,
           $ ' PER SECOND )', /)
1706    620 FORMAT(T42, 'OTHER DEMANDS, MAXIMUM = ', F5.2,/)
1707    630 FORMAT(T26, 'NUMBER OF INITIAL CONCENTRATION VALUES = ',I2,/)
1708    635 FORMAT(T16, 'INITIAL LCCATION CF UPPERMOST PEAK CONCENTRATION = ',
           $  F6.2, ' MILES (SECTICN NUMBER ', I2,')',/)
1709    637 FORMAT(T16, 'INITIAL LCCATION CF LCWERMOST PEAK CONCENTRATION = ',
           $  F6.2, ' MILES (SECTION NUMBER ', I2,')')
1710    64C FORMAT(T56, 'STABILITY CRITERIA',/)
1711    645 FORMAT(T35, 'MAXIMUM ALLOWABLE X INCREMENT = ', F7.0, ' FEET (',
           $ F8.5, 'MILES )',/)
1712    650 FORMAT(T35, 'MAXIMUM ALLCWABLE Z INCREMENT = ', F7.0, ' FEET (',
           $ F8.5, 'MILES )',/)
1713    655 FORMAT(T32, 'MAXIMUM ALLOWABLE TIME INCREMENT = ', F7.0, ' SECCNDS
           $ (', F10.3, ' DAYS )',/)
1714    660 FORMAT(T38, 'ACTUAL DELTAT/(DELTAX**2.) = ', E10.3,/)
1715    665 FORMAT(T38, 'ACTUAL DELTAT/(DELTAZ**2.) = ', E10.3,/)
1716    700 FORMAT(T25, 'LOCATIONS OF INITIAL CONCENTRATIONS',//)
1717    705 FORMAT(     'A WASTE SOURCE IS LOCATED AT STATION ', I2, ' (MILE ',
           $ F6.3 , ' ).  THE CONCENTRATION IS', F10.2, ' PPM.')
1718    710 FORMAT( 1X,'AN INITIAL CONCENTRATION IS FOUND AT STATION ',I2,
           1  ' (MILE ', F6.3, ' ).',/, ' THE CONCENTRATIONS AT ', F7.1,
           2  ' FOOT INTERVALS WITH DEPTH ARE')
1719    715 FORMAT(T15, 'C(', I2','I2',1) = ', F9.2, ' PPM')
1720        RETURN
1721        END
```

```
      C-----------------------------------------------------------------------
1722        SUBROUTINE PRINT2
      C
      C             *****************************
      C             ***** SUBROUTINE PRINT2 *****
      C             *****************************
      C
      C          THIS SUBROUTINE PRINTS OUT THE TIME AND THE CONCENTRATION
      C     PROFILE ACCORDING TO A FORMAT PREVIOUSLY READ INTO THE PROGRAM.
      C     THIS SUBROUTINE CAN BE CHANGED ACCORDING TO THE NEEDS OF THE USER.
      C     ***WARNING--DO NOT CHANGE ANY VARIABLE OCCURING IN A COMMON BLOCK.
      C
      C-----------------------------------------------------------------------
1723        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
1724        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1725        COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1726        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1727        COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
           1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
           2               G(51),D(51),V(51)
1728        DIMENSION NZREV(20)
1729        DIMENSION R(51,21)
```

```
      DAYS = TIME/8640C.
      HOURS=DAYS*24.
      WRITE(6,FMT1) HCURS,DAYS
      DO 100 IZ = 1, NZ
      NZREV(IZ)=IFIX(DELTAZ*(IZ-2))
  100 CONTINUE
      WRITE(6,FMT2) (NZREV(I),I=2,NZM1)
C
C-----------------------------------------------------------------------
C         ALLOW FOR A BACKGROUND CONCENTRATION OF 10.0
C-----------------------------------------------------------------------
      DO 150 X=1,NX
      DO 150 Z=1,NZ
      R(X,Z)=C(X,Z,T)+10.0
  150 CONTINUE
      XFIRST=XSTART-(NX-1)*DELTAX/5280.
      DO 200 X=1,NX,XPRINT
      M=NX+1-X
      XMILE = XFIRST + ((X-1)*DELTAX*XPRINT)/5280.
      WRITE(6,FMT3) XMILE, (R(M,IZ),IZ=1,NZ)
  200 CONTINUE
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
1749        SUBROUTINE ERRORI(*)
      C
      C             **************************
      C            **** SUBROUTINE ERRORI *****
      C             **************************
      C
      C          THIS SUBROUTINE CORRECTS SOME OF THE MOST COMMON ERRORS
      C     IN THE INPUT DATA AND PRINTS OUT APPROPRIATE ERROR MESSAGES.
      C
      C-----------------------------------------------------------------------
1750        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
1751        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1752        COMMON/ARRAYS/ C(51,21,2),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1753        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1754        COMMON/IMPLCT/EXNEXT(51,21),EZNEXT(51,21),CSTAR(51,21),
           1               VXNEXT(51,21),VZNEXT(51,21),A(51),B(51),
           2               G(51),D(51),V(51)
1755        IF(OPTION.GT.7) GO TO 180
1756        IF(OPTION.EQ.7) GO TO 140
1757        IF(OPTION.EQ.6) GO TO 120
1758        GO TO 200
```

```
 120 CONTINUE
     OPTION=2
     PRINT 130
 130 FORMAT(/////, ' ****OPTION IS SET EQUAL TO 2. OPTION WAS EQUAL TO
    $6 ****',/,1H1)
     IPAGE = 0
     GO TO 200
 140 CONTINUE
     OPTION=4
     PRINT 150
 150 FORMAT(/////, ' **** OPTION IS SET EQUAL TO 4. OPTION WAS EQUAL TO
    $ 7 ****',/,1H1)
     IPAGE=0
     GO TO 200
 180 CONTINUE
     PRINT 181
 181 FORMAT(/////,' **** PROGRAM TERMINATED BECAUSE CPTION WAS GREATER
    $THAN 7 ****',/,1H1)
     RETURN1
 200 IF(LTYPE.LT.3) GC TC 300
     PRINT 201
 201 FORMAT(/////, ' **** PROGRAM TERMINATED BECAUSE LTYPE IS GREATER T
    $HAN OR EQUAL TC 3.',/,'   IF LTYPE EQUALS 3, THE USER MUST SUPPLY
    $A SUBROUTINE FOR THE LCADING CONDITIONS.',/,1H1)
     RETURN1
 300 CONTINUE
     RETURN
     END
```

Input Data for MASSTRANS-II

See Input Data for MASSTRANS-I.

COMPUTER PROGRAM FOR

OXTRANS-I

Object Code	=	66,312 bytes
Array Area	=	58,372 bytes
Total	=	124,684 bytes

```
C     ************************************************************
C     *                                                          *
C     *                        OXTRANS-I                         *
C     *                                                          *
C     *     ESTUARY MASS TRANSPORT AND OXYGEN TRANSPORT MODEL    *
C     *                                                          *
C     *  TWO DIMENSIONAL NUMERICAL ANALYSIS - EXPLICIT METHOD    *
C     *                                                          *
C     ******---------------------------------------------*******
C          *                                             *
C          *  LOGIC AND PROGRAMMING BY                   *
C          *                                             *
C          *         P. JONATHAN YOUNG                   *
C          *    C/O  ENVIRONMENTAL ENGINEERING DIVISION  *
C          *         CIVIL ENGINEERING DEPARTMENT        *
C          *         TEXAS A&M UNIVERSITY                *
C          *         COLLEGE STATION, TEXAS 77840        *
C          *                                             *
C          ***********************************************
C
C
C
C         THIS PROGRAM ANALYZES THE RELATIONSHIP BETWEEN A PRIMARY
C     POLLUTANT AND A DISSOLVED GAS IN A PARTIALLY STRATIFIED ESTUARY.
C     THE PRINCIPLE APPLICATION OF OXTRANS-I IS TO THE INTERACTION OF
C     BIOCHEMICAL OXYGEN DEMAND (BOD) AND DISSOLVED OXYGEN (DO).
C
```

```
C           OXTRANS-I PERFORMS TWO BASIC CALCULATIONS DURING EACH TIME
C      STEP.  THE FIRST CALCULATION IS TO DETERMINE THE PROFILE FOR THE
C      PRIMARY POLLUTANT, SUCH AS BOD, BY AN EXPLICIT FINITE DIFFERENCE
C      APPROXIMATION TO THE MASS TRANSPORT EQUATION.  THE SECOND CALCULA-
C      TION USES INFORMATICN FROM THE FIRST CALCULATION TO DETERMINE
C      THE VALUES FOR THE DISSOLVED GAS.  THIS PROGRAM HAS OPTIONS FOR
C      ANAEROBIC CONDITIONS AND FOR MECHANICAL REAERATION.  DURING A
C      CALCULATION FOR BOD AND DO, THE PROGRAM AUTOMATICALLY SWITCHES TO
C      ANAEROBIC DECAY RATES WHEREVER THE CONCENTRATION OF OXYGEN IS
C      ZERO.  REAERATION TERMS CAN BE APPLIED TO ANY POINT ON THE GRID
C      AND, THUS, THE EFFECTS OF MECHANICAL AERATION AT ANY POINT OR
C      POINTS CAN BE INVESTIGATED.  AT THE USERS OPTION, THE PROFILES
C      FOR THE PRIMARY POLLUTANTS ALONE CAN BE CALCULATED, AND THE
C      CALCULATIONS FOR THE SECONDARY PARAMETERS CAN BE BY-PASSED.
C
C           THIS PROGRAM WAS DEVELOPED TO ANALYZE ESTUARIES WHOSE
C      CHARACTERISTICS DO NOT VARY SIGNIFICANTLY WITH WIDTH.  THE WIDTH
C      OF THE ESTUARY MAY BE VARIED THROUGHOUT BUT THE CONCENTRA-
C      TIONS OF DISSOLVED MATERIALS AT EACH CROSS SECTION ARE CONSIDERED
C      TO BE UNCHANGING IN THE LATERAL DIRECTION.  ALL PHYSICAL AND
C      HYDRODYNAMIC CHARACTERISTICS MAY VARY WITH TIME IN THE LONGITU-
C      DINAL AND VERTICAL DIRECTIONS.  THE PROGRAM CAN BE APPLIED WITH
C      EQUAL EASE TO THE TWO HORIZONTAL DIRECTIONS, ALLOWING DEPTH TO
C      VARY RATHER THAN WIDTH.
C
C           CONCENTRATION PROFILES CAN BE CALCULATED BY THIS PROGRAM FOR
C      CONTINUOUS OR INSTANTANEOUS RELEASES.  INPUT DATA MAY INCLUDE GRID
C      DIMENSIONS, DISTANCE INCREMENTS, TIME INCREMENTS, WIDTHS, LOADING
C      PARAMETERS, VELOCITIES, DISPERSION COEFFICIENTS, DECAY RATES,
C      BENTHAL DEMANDS, REAERATION COEFFICIENTS, AEROBIC DECAY RATES,
C      ANAEROBIC DECAY RATES, AND OTHER OXYGEN DEMANDS OR SOURCES.
```

```
C
C          THE TIME INCREMENTS CAN BE INCREASED OR DECREASED AT ANY TIME
C     DURING THE CALCULATION OF THE CONCENTRATION PROFILE.  THE DIS-
C     TANCES BETWEEN GRID POINTS CAN BE INCREASED AT ANY TIME, AND A
C     ROUTINE WITHIN THE PROGRAM WILL CHOOSE THE APPROPRIATE VALUES FROM
C     THE PREVIOUSLY CALCULATED PROFILE AND WILL PLACE THESE VALUES IN
C     THE DESIRED LOCATIONS IN THE NEW GRID SYSTEM.  THE NUMBER OF GRID
C     POINTS MAY BE INCREASED OR DECREASED AT ANY TIME.  LIKEWISE, AT
C     ANY TIME DURING THE CALCULATION, A NEW SET OF DATA FOR PHYSICAL
C     AND HYDRODYNAMIC CONDITIONS MAY BE READ INTO THE PROGRAM.
C
C          A USER OF THIS PROGRAM MUST BE FAMILIAR WITH THE LIMITATICNS
C     ON ACCURACY AND STABILITY INHERENT IN THE TYPE OF NUMERICAL PROCE-
C     DURE USED IN THESE CALCULATIONS.  A SUBROUTINE WITHIN THE PROGRAM
C     PRINTS OUT THE PROPER INCREMENTS TO INSURE STABILITY AND TERMI-
C     NATES THE PROGRAM IF THIS CRITERIA IS VIOLATED BY THE INPUT PARA-
C     METERS.  ANOTHER SUBROUTINE EXTRAPOLATES CONCENTRATIONS AT THE
C     BOUNDARIES--SEVERAL METHODS CAN BE USED FOR THESE EXTRAPOLATIONS
C     DEPENDING ON THE TYPE CF PROFILE BEING ANALYZED AND THE CHOICE OF
C     THE USER.  A SUBROUTINE IS ALSO INCLUDED WHICH PRINTS OUT ERROR
C     MESSAGES AND TERMINATES THE PROGRAM IF CERTAIN INCONSISTENCIES
C     OCCUR IN THE INPUT DATA.
C
```

```
C          THIS PROGRAM WAS DEVELOPED PRIMARILY TO ANALYZE PARTIALLY
C     STRATIFIED ESTUARIES WHICH HAVE BEEN DREDGED OUT TO A FAIRLY
C     CONSTANT DEPTH AT THE CENTERLINE OF THE CHANNEL--THESE ESTUARIES
C     ARE COMMON IN THE GULF COAST REGION.  APPLICATION OF THIS PROGRAM
C     TO PARTIALLY STRATIFIED ESTUARIES WITH VARIABLE DEPTHS WOULD
C     REQUIRE MODERATE REVISIONS TC THE PROGRAM AND WOULD MAKE THE
C     PROGRAM ESTUARY-DEPENDENT.
C
C          THIS COMPUTER PROGRAM CAN ALSC BE APPLIED TO ESTUARIES WHICH
C     ARE WELL-MIXED IN THE VERTICAL DIRECTION.  THIS OPTION ALLOWS FOR
C     VARYING WIDTH OR VARYING DEPTH AND USES MOST OF THE ROUTINES
C     AVAILABLE TO THE TWO-DIMENSIONAL ANALYSIS.
C
C          QUESTIONS REGARDING THIS PROGRAM MAY BE REFERRED TO
C     JONATHAN YOUNG AT HYDROSCIENCE, INC, 363 OLD HOOK ROAD,
C     WESTWOOD, NEW JERSEY  C7675   PHCNE 201 / 666-2600
C
C-----------------------------------------------------------------------
```

```
C--------------------------------------------------------------------
C         ***INPUT VARIABLES***
C
C        SET          = DATA SET NUMBER
C                       --CAN BE EQUAL TO 1 IF NO NEW GRID DIMENSIONS
C                         ARE TO BE READ IN--MUST BE GREATER THAN 1 IF
C                         GRID DIMENSIONS ARE TO BE CHANGED
C        SECOND       = .TRUE. FOR CALCULATION OF SECONDARY PARAMETERS
C                     = .FALSE. FOR CALCULATING PRIMARY POLLUTANTS ALONE
C        **************NOTE--CPTIONAL GRID ADJUSTMENT PARAMETERS ARE
C                         INSERTED HERE.  THESE ARE EXPLAINED BELOW
C        NX           = NUMBER OF GRID POINTS IN THE X DIRECTION
C        NZ           = NUMBER OF GRID POINTS IN THE Z DIRECTION
C        DELMX        = X INCREMENT ( MILES )
C        DELTAZ       = Z INCREMENT ( FEET )
C        DELT1        = FIRST TIME INCREMENT ( DAYS )
C        ITER1        = NUMBER OF ITERATIONS USING DELT1
C        DELT2        = SECOND TIME INCREMENT ( DAYS )
C        ITER2        = NUMBER OF ITERATIONS USING DELT2
C        NC           = NUMBER OF GRID POINTS IN THE X DIRECTION WHERE
C                       CONCENTRATIONS ARE ADDED WITH THE DATA SET
C        NS           = NUMBER OF POINT SOURCES
C        XSTART       = MILEAGE AT UPPERMOST GRID POINT
C        TSTART       = TIME TO BE ADDED TO THE TOTAL TIME AT THE FIRST
C                       ITERATION OF THE DATA SET BEING PROCESSED( DAYS )
C        XPRINT       = PRINTING INTERVAL FOR X DIRECTION ( INTEGER )
C        TPRINT       = PRINTING INTERVAL FOR TIME ( INTEGER )
C        NPPAGE       = NUMBER OF GRIDS PRINTED PER PAGE OF OUTPUT
```

```
C         LTYPE         = TYPE CF LOADING ( INSTANTANEOUS OR CONTINUOUS )
C                           1 MEANS CONSTANT CONCENTRATION AT OUTFALL
C                             (THIS CAN BE USED FOR STEADY-STATE PROFILES)
C                           2 MEANS AN INSTANTANEOUS RELEASE
C                           3 MEANS CONTINUOUS LOADING
C         OPTION        = OPTION FOR BOUNDARY CONDITIONS
C                           1 MEANS NC TRANSFER ACROSS Z BOUNDARIES AND
C                             EXPONENTIAL EXTRAPOLATION IN X DIRECTION
C                           2 MEANS NO TRANSFER ACROSS Z BOUNDARIES AND
C                             CCNSTANT SLOPE EXTRAPOLATION IN X DIRECTION
C                           3 MEANS A SYMMETRICAL DISTRIBUTION WITH
C                             EXPONENTIAL EXTRAPOLATION
C                           4 MEANS A CONSTANT SLOPE EXTRAPOLATION IN
C                             BOTH DIRECTIONS
C                           5 MEANS A SPECIAL CASE DEFINED BY THE USER
C                           6 MEANS NO TRANSFER ACROSS Z BOUNDARIES AND
C                             EXTRAPOLATICN IN X DIRECTION BY INVERTED
C                             DIFFERENCES (ACFI)
C                           7 MEANS EXTRAPOLATION IN BOTH DIRECTIONS BY
C                             INVERTED DIFFERENCES (ACFI)
C         VX(X,Z)       = HORIZONTAL VELOCITY ( FT/SEC )
C         EX(X,Z)       = HORIZONTAL DISPERSION ( FT**2/SEC )
C         W(X)          = WIDTH ( FEET ) FOR THE TWO DIMENSIONAL CASE OR
C                         CROSS SECTIONAL AREA (FT**2) FOR THE ONE
C                         DIMENSIONAL CASE
C         KDAYS(X)      = DECAY RATE ( PER DAY )
C         DEMAND(X,Z)   = OTHER SOURCES AND SINKS (CONCENTRATION UNITS)
C         INSECT(I)     = SECTICNS INTO WHICH ARE ADDED LOADS
C                         OR CONCENTRATIONS
C         CONCIN(I,J)   = INITIAL CONCENTRATIONS ASSOCIATED WITH INSECT(I)
```

```
C         **************NOTE--VALUES FOR VZ AND EZ ARE TO BE INCLUDED
C                           WITH THE DATA ONLY IF NZ IS GREATER THAN 1
C         VZ(X,Z)      = VERTICAL VELOCITIES ( FT/SEC )
C         EZ(X,Z)      = VERTICAL DISPERSION ( FT**2/SEC )
C
C         FMT1(I)      = FORMAT FOR HEADING FOR PRIMARY POLLUTANT PROFILES
C         FMT2(I)      = FORMAT FOR DEPTH NOTATION FOR PRIMARY POLLUTANTS
C         FMT3(I)      = FORMAT FOR PRIMARY POLLUTANT PROFILES
C
C         **************NOTE--THE FOLLOWING VALUES ARE TO BE INCLUDED
C                           IN THE INPUT DATA ONLY IF SECOND = .TRUE.
C         FMT4(I)      = FORMAT FOR HEADING FOR SECONDARY PARAMETERS
C         FMT5(I)      = FORMAT FOR DEPTH NOTATION FOR SECCNDARY PROFILES
C         FMT6(I)      = FORMAT FOR SECONDARY PARAMETER PROFILES
C         RDAYS(X,Z)   = REAERATION COEFFICIENTS ( /DAY )
C         AKDAYS(X)    = ANAEROBIC DECAY RATE ( /DAY )
C         OXSAT(X)     = OXYGEN SATURATION VALUES (CONCENTRATION UNITS )
C         OXOUT(X,Z)   = OXYGEN SINKS (CONCENTRATION UNITS )
C         ***************NOTE--INITIAL CONCENTRATIONS MAY ALSO BE INPUT
C                        THROUGH THE BLOCK DATA SUBPROGRAM
C
C-----------------------------------------------------------------------
C
```

```
C
C---------------------------------------------------------------------
C
C          *** GRID ADJUSTMENT PARAMETERS ***
C
C
C        NXSKIP       = NUMBER OF X INTERVALS SKIPPED IN CLD GRID
C        NZSKIP       = NUMBER OF Z INTERVALS SKIPPED IN OLD GRID
C        NXTAKE       = LOWEST GRID NUMBER IN X DIRECTION IN THE OLD GRID
C                       FRCM WHICH A CONCENTRATION IS TAKEN
C        NZTAKE       = LOWEST GRID NUMBER IN Z DIRECTION IN THE OLD GRID
C                       FROM WHICH A CONCENTRATION IS TAKEN
C        NXPUT        = LOWEST GRID NUMBER IN THE X DIRECTION IN THE NEW
C                       GRID INTO WHICH A CONCENTRATION IS PLACED
C        NZPUT        = LOWEST GRID NUMBER IN THE Z DIRECTION IN THE NEW
C                       GRID INTO WHICH A CONCENTRATION IS PLACED
C---------------------------------------------------------------------
```

```
C-----------------------------------------------------------------------
C
C          ************************
C          ***** MAIN PROGRAM *****
C          ************************
C
C          THE MAIN PROGRAM CALLS FOR THE DATA TO BE READ IN,THEN CALLS
C     FOR THE STABILITY CONDITIONS TO BE EVALUATED,AND NEXT CALLS FOR
C     THE PRINTING OF A SUMMARY OF INPUT DATA.  THE NUMERICAL ANALYSIS
C     SUBROUTINE IS THEN CALLED FOR EITHER THE ONE-DIMENSIONAL OR TWO-
C     DIMENSIONAL CASE.  IF THE TIME INCREMENT IS TO BE CHANGED DURING
C     THIS PART OF THE PROGRAM,THE APPROPRIATE PARAMETERS ARE ADJUSTED.
C     STABILITY IS AGAIN CHECKED AND THE NUMERICAL ANALYSIS IS
C     CONTINUED.  NEXT,A NEW DATA SET MAY BE READ TO CONTINUE THE
C     ANALYSIS.  ANY NUMBER OF DATA SETS MAY BE PROCESSED.  WHEN NO MORE
C     DATA SETS ARE AVAILABLE,THE PROGRAM IS TERMINATED.
C
C          THE INPUT AND OUTPUT OF DATA CAN BE CONTROLLED IN THREE
C     WAYS--THE INPUT DATA ITSELF, THE BLOCK DATA SUBROUTINE, AND THE
C     SUBROUTINE PRINT2.  NO OTHER PART OF THE PROGRAM SHOULD REQUIRE
C     CHANGING.
C
C          THE FORTRAN USED IN THIS PROGRAM USES SEVERAL OF THE OPTIONS
C     AVAILABLE IN WATFOR AS DESCRIBED IN THE FOLLOWING TEXT**FORTRAN IV
C     WITH WATFOR AND WATFIV**BY CRESS,DIRKSEN,&GRAHAM,PRENTICE-HALL,
C     INC.,1970. THE PROGRAM CAN BE RUN ON A COMPUTER WITH A WATFOR OR
C     WATFIV COMPILER.  THE SYNTAX HAS BEEN KEPT COMPATIBLE WITH
C     FORTRAN G EXCEPT FOR THE UNFORMATTED READ AND PRINT STATEMENTS.
C
C-----------------------------------------------------------------------
```

```
1977          LOGICAL SECOND
1978          INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
1979          REAL KD,KDMAX,KDMIN,KDAYS,LOWER
1980          COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
             1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
             2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
1981          COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
             1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
             2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
             3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
             4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
             5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
1982          COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
             1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
             2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
1983       10 CALL DATA(&1000)
1984          CALL STABLE
1985          CALL PRINT1
1986          CALL PRINT2
1987          T=3
1988          PRINT 15
1989       15 FORMAT(1H1)
1990          CALL PRINT2
1991          CALL ERROR(&1000)
1992       25 IF(NZ.EQ.1) GC TO 310
1993          CALL TWODEX
1994          GO TO 315
1995      310 CALL ONEDEX
1996      315 IF(ITER2.EQ.0) GC TO 10
```

```
      ITER=ITER2
      ITER2=0
      TPRINT=TPRINT*(DELT1/DELT2+.00001)
      DELTAT=DELT2*86400.
      CALL STABLE
      IF(ITER.EQ.0) CALL ERROR(&1000)
      GO TO 25
 1000 CONTINUE
      STOP
      END
```

```
      C------------------------------------------------------------------
2007        BLOCK DATA
      C
      C              **********************
      C              ***** BLOCK DATA *****
      C              **********************
      C
      C          THIS ROUTINE INITIALIZES SELECTED VARIABLES IN THE
      C     COMMON BLOCKS.  THIS SUBPROGRAM MAY BE USED TO ESTABLISH
      C     INITIAL CONCENTRATION PROFILES IF THE USER DESIRES.
      C
      C------------------------------------------------------------------
2008        LOGICAL SECOND
2009        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
2010        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
2011        COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1                W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2                DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
2012        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1                KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2                TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3                VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4                DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5                MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTICN
2013        COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
           1                AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
           2                SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
```

```
C
C-------------------------------------------------------------------------
C        ESTABLISH INITIAL CONCENTRATION IF DESIRED
C-------------------------------------------------------------------------
      DATA C/ 9*13.,42*0.,9*13.,42*0.,9*13.,42*0.,9*13.,42*0.,
     1        9*13.,42*0.,9*13.,42*0.,9*13.,42*0., 1785*0.0,
     2        9*0., 42*8., 9*0., 42*8., 9*0.,42*8., 9*0., 42*8.,
     3        9*0., 42*8., 9*0., 42*8., 9*0.,42*8., 1785*8. /
      DATA VX,VZ,EX,EZ,CONCIN,DEMAND/6426*0.0/
      DATA W,KDAYS,KD/153*0.0/
      DATA EXMAX,EZMAX,VXMAX,VZMAX,WMAX,KDMAX,DEMAX,TIME/8*0.0/
      DATA EXMIN,EZMIN,VXMIN,VZMIN,WMIN,KDMIN,DEMIN/7*1000000./
      DATA INSECT/51*0/
      DATA IPAGE/1/
      DATA T/1/,TNEXT/2/
      DATA AKMAX,RMAX,OUTMAX/3*0.0/
      DATA AKMIN,RMIN,OUTMIN/3*1000000./
      END
```

```
      C-----------------------------------------------------------------------
2025        SUBROUTINE DATA(*)
      C
      C             ***************************
      C             ***** SUBRCUTINE DATA *****
      C             ***************************
      C
      C          THIS SUBROUTINE READS IN THE APPROPRIATE CATA SET AND ADJUSTS
      C     THE GRID SIZE IF NECESSARY.  THE LOCATIONS OF THE UPPERMOST AND
      C     LOWERMOST PEAKS ARE DETERMINED, ANC THE MINIMUM AND MAXIMUM VALUES
      C     FOR THE INPUT PARAMETERS ARE CALCULATED.  ALL OF THE INPUT DATA IS
      C     PRINTED OUT UNFORMATTED.  THE FORMAT IS READ IN FOR THE PRINTING
      C     OUT OF THE CALCULATED CONCENTRATIONS.
      C
      C-----------------------------------------------------------------------
2026        LOGICAL SECOND
2027        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
2028        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
2029        COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
2030        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
2031        COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
           1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
           2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
```

```
      READ(5,12,END=13) SET
   12 FORMAT(F10.0)
      GO TO 14
   13 CONTINUE
      RETURN1
   14 CONTINUE
      READ,SECOND
      IF(SET.EQ.1.)  GO TO 400
C
C-----------------------------------------------------------------------
C           CHOOSE CONCENTRATIONS FROM THE OLD GRID TO PUT INTO THE
C           APPROPRIATE PLACES IN THE NEW GRID .  THE GRID CAN ONLY BE
C           CHANGED IF LTYPE EQUALS 2.  OTHERWISE, THE DATA CARD IS READ
C           BUT  IS NOT IMPLEMENTED.
C-----------------------------------------------------------------------
      READ ,NXSKIP,NZSKIP,NXTAKE,NZTAKE,NXPUT,NZPUT
      IF(LTYPE.NE.2) GO TO 400
      NXSTOP=(NX-1)/NXSKIP+1
      NZSTOP=(NZ-1)/NZSKIP+1
      IF(NZ.EQ.1) NZSTOP=1
      DO 350 I=1,NXSTOP
      X=NXPUT-1+I
      II=NXTAKE+NXSKIP*(I-1)
      DO 350 J=1,NZSTOP
      Z=NZPUT-1+J
      JJ=NZTAKE+NZSKIP*(J-1)
      C(X,Z,TNEXT)=C(II,JJ,T)
  350 CONTINUE
  400 CONTINUE
```

```
C
C------------------------------------------------------------------------
C           READ AND PRINT INPUT DATA FOR HCRIZONTAL DIRECTION
C------------------------------------------------------------------------
      READ,NX,NZ
      READ,DELMX,DELTAZ
      READ,DELT1,ITER1
      READ,DELT2,ITER2
      READ,NC,NS
      READ,XSTART,TSTART,XPRINT,TPRINT,NPPAGE
      READ,LTYPE,OPTION
      PRINT 1001
 1001 FORMAT( 1H1,10X,'INPUT DATA',///,
     1   1X,'NX,NZ,    DELMX,DELTAZ,   DELT1,ITER1')
      PRINT,NX,NZ,DELMX,DELTAZ,DELT1,ITER1
      PRINT 1010
 1010 FORMAT(//,1X,'SECCND')
      PRINT,SECOND
      PRINT 1002
 1002 FORMAT(//,1X,'DELT2,ITER2,    NC,NS,    XSTART,TSTART')
      PRINT, DELT2,ITER2,NC,NS,XSTART,TSTART
      PRINT 1003
 1003 FORMAT(//,1X,'XPRINT,TPRINT,NPPAGE,    LTYPE,OPTION')
      PRINT, XPRINT,TPRINT,NPPAGE,LTYPE,CPTION
      IF(SET.EQ.1.) GC TO 500
      DO 375 Z=1,NZ
      DO 375 X=1,NX
      C(X,Z,T)=C(X,Z,TNEXT)
      C(X,Z,TNEXT)=0.0
  375 CONTINUE
  500 CONTINUE
```

```
      NXM1=NX-1
      NXM2=NX-2
      NZM1=NZ-1
      NZM2=NZ-2
      DELTAT=DELT1*86400.
      DELTAX=DELMX*5280.
      XSF=XSTART*5280.
      ITER=ITER1
      TIME=TIME+TSTART*86400.
      COUNT=(TSTART/DELTAT+0.01)
      COUNT=COUNT-(COUNT/TPRINT)*TPRINT
      IF(SET.EQ.2) RETURN
      READ,((VX(X,Z),X=1,NX),Z=1,NZ)
      PRINT 2
    2 FORMAT(//, 1X, '((VX(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((VX(X,Z),X=1,NX),Z=1,NZ)
      READ,((EX(X,Z),X=1,NX),Z=1,NZ)
      PRINT 3
    3 FORMAT(//, 1X, '((EX(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((EX(X,Z),X=1,NX),Z=1,NZ)
      READ,(W(X),X=1,NX)
      PRINT 4
    4 FORMAT(//, 1X, '(W(X),X=1,NX)')
      PRINT, (W(X),X=1,NX)
      READ,(KDAYS(X),X=1,NX)
      PRINT 5
    5 FORMAT(//, 1X, '(KDAYS(X),X=1,NX)')
      PRINT, (KDAYS(X),X=1,NX)
```

```
      READ,((DEMAND(X,Z),X=1,NX),Z=1,NZ)
      PRINT 6
    6 FORMAT(//, 1X, '((DEMAND(X,Z),X=1,NX),Z=1,NZ)')
      PRINT, ((DEMAND(X,Z),X=1,NX),Z=1,NZ)
      IF(NC.EQ.0) GO TO 24
      DO 10 I=1,NC
      READ,INSECT(I)
      PRINT 7
    7 FORMAT(//, 1X, 'INSECT(I)')
      PRINT,INSECT(I)
      READ, (CONCIN(I,J), J=1,NZ)
      PRINT 8
    8 FORMAT(//, 1X, '(CONCIN(I,J),J=1,NZ)')
      PRINT,(CONCIN(I,J),J=1,NZ)
   1C CONTINUE
      DO 15 I=1,NC
      DO 15 Z=1,NZ
      C(INSECT(I),Z,T)=CONCIN(I,Z)
   15 CONTINUE
C
C-----------------------------------------------------------------
C           DETERMINE LOCATIONS OF UPPERMOST AND LOWERMOST
C           PEAK CONCENTRATIONS
C-----------------------------------------------------------------
      AMAX1=0.0
      PEAKU=0.0
      AMAX2=0.0
      PEAKL=0.0
      UPPER=XSF
      LOWER=XSF
```

```
      MSU=1
      MSL=NX
      DO 17 I=1,NC
      DO 16 J=1,NZ
      IF(CONCIN(I,J).LT.PEAKU) GO TO 16
      PEAKU=CONCIN(I,J)
   16 CONTINUE
      IF(PEAKU.LE.AMAX1) GO TO 20
      AMAX1=PEAKU
      MSU=INSECT(I)
      UPPER=(MSU-1)*DELTAX +XSF
   17 CONTINUE
   20 CONTINUE
      DO 22 II=1,NC
      I=NC+1-II
      DO 21 J=1,NZ
      IF(CONCIN(I,J).LT.PEAKL) GO TO 21
      PEAKL=CONCIN(I,J)
   21 CONTINUE
      IF(PEAKL.LE.AMAX2) GO TO 24
      AMAX2=PEAKL
      MSL=INSECT(I)
      LOWER=(MSL-1)*DELTAX +XSF
   22 CONTINUE
   24 CONTINUE
C
C-----------------------------------------------------------------------
C           DETERMINE MAXIMUM AND MINIMUM VALUES FOR INPUT DATA
C-----------------------------------------------------------------------
      DO 25 X=1,NX
```

```
      KD(X)=KDAYS(X)/86400.
      IF(W(X).GT.WMAX)WMAX=W(X)
      IF(W(X).LT.WMIN)WMIN=W(X)
      IF(KD(X).GT.KDMAX)KDMAX=KD(X)
      IF(KD(X).LT.KDMIN)KDMIN=KD(X)
      DO 25 Z=1,NZ
      ABSVX=ABS(VX(X,Z))
      IF(ABSVX.GT.VXMAX)VXMAX=ABSVX
      IF(ABSVX.LT.VXMIN)VXMIN=ABSVX
      IF(EX(X,Z).GT.EXMAX)EXMAX=EX(X,Z)
      IF(EX(X,Z).LT.EXMIN)EXMIN=EX(X,Z)
      IF(DEMAND(X,Z).GT.DEMAX)DEMAX=DEMAND(X,Z)
      IF(DEMAND(X,Z).LT.DEMIN)DEMIN=DEMAND(X,Z)
   25 CONTINUE
      IF(NZ.EQ.1)GO TO 200
C
C-----------------------------------------------------------------------
C          READ AND PRINT DATA FOR VERTICAL DIRECTION
C-----------------------------------------------------------------------
      READ,((VZ(X,Z),X=1,NX),Z=1,NZ)
      PRINT 30
   30 FORMAT(//, 1X, '((VZ((X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((VZ(X,Z),X=1,NX),Z=1,NZ)
      READ,((EZ(X,Z),X=1,NX),Z=1,NZ)
      PRINT 32
   32 FORMAT(//, 1X, '((EZ(X,Z),X=1,NX),Z=1,NZ)')
      PRINT, ((EZ(X,Z),X=1,NX),Z=1,NZ)
C
C-----------------------------------------------------------------------
C          DETERMINE MAXIMUM AND MINIMUM VALUES FOR INPUT DATA
C-----------------------------------------------------------------------
```

```
      DO 120 X=1,NX
      DO 120 Z=1,NZ
      ABSVZ=ABS(VZ(X,Z))
      IF(ABSVZ.GT.VZMAX)VZMAX=ABSVZ
      IF(ABSVZ.LT.VZMIN)VZMIN=ABSVZ
      IF(EZ(X,Z).GT.EZMAX)EZMAX=EZ(X,Z)
      IF(EZ(X,Z).LT.EZMIN)EZMIN=EZ(X,Z)
  120 CONTINUE
  200 CONTINUE
C
C-----------------------------------------------------------------------
C            READ AND PRINT FORMATS FOR OUTPUT
C-----------------------------------------------------------------------
      READ 205, (FMT1(I), I=1,40)
      PRINT 202
  202 FORMAT(//,1X, 'FORMAT FOR OUTPUT')
      PRINT 205, (FMT1(I),I=1,40)
  205 FORMAT (20A4,/,20A4)
      READ 205,(FMT2(I),I=1,40)
      PRINT 205, (FMT2(I), I=1,40)
      READ 207,(FMT3(I), I=1,20)
      PRINT 207, (FMT3(I), I=1,20)
  207 FORMAT(20A4)
      IF(.NOT.SECOND) GO TO 250
      READ  205, (FMT4(I),I=1,40)
      PRINT 205, (FMT4(I),I=1,40)
      READ  205, (FMT5(I),I=1,40)
      PRINT 205, (FMT5(I),I=1,40)
      READ  207, (FMT6(I),I=1,20)
```

```
      PRINT 207, (FMT6(I),I=1,20)
      READ , ((RDAYS(X,Z),X=1,NX),Z=1,NZ)
      PRINT 210
  210 FORMAT(//,1X,'((RDAYS(X,Z),X=1,NX),Z=1,NZ)')
      PRINT, ((RCAYS(X,Z),X=1,NX),Z=1,NZ)
      READ , (AKDAYS(X),X=1,NX)
      PRINT 215
  215 FORMAT(//,1X,'(AKDAYS(X),X=1,NX)')
      PRINT, (AKDAYS(X),X=1,NX)
      READ ,(OXSAT(X),X=1,NX)
      PRINT 220
  220 FORMAT(//,1X,'(OXSAT(X),X=1,NX)')
      PRINT,(OXSAT(X),X=1,NX)
      READ ,((OXOUT(X,Z),X=1,NX),Z=1,NZ)
      PRINT 222
  222 FORMAT(//,1X,'((OXOUT(X,Z),X=1,NX),Z=1,NZ)')
      PRINT,((OXOUT(X,Z),X=1,NX),Z=1,NZ)
      DO 225 X=1,NX
      AK(X)=AKDAYS(X)/86400.
      IF(AK(X).GT.AKMAX) AKMAX=AK(X)
      IF(AK(X).LT.AKMIN) AKMIN=AK(X)
      DO 225 Z=1,NZ
      R(X,Z)=RDAYS(X,Z)/86400.
      IF(R(X,Z).GT.RMAX) RMAX=R(X,Z)
      IF(R(X,Z).LT.RMIN) RMIN=R(X,Z)
      IF(OXOUT(X,Z).GT.OUTMAX) OUTMAX=OXOUT(X,Z)
      IF(OXOUT(X,Z).LT.OUTMIN) OUTMIN=OXCUT(X,Z)
  225 CONTINUE
  250 CONTINUE
      PRINT 259
  259 FORMAT(1H1)
      RETURN
      END
```

```
      C-----------------------------------------------------------------
2240        SUBROUTINE STABLE
      C
      C              *****************************
      C              ***** SUBRCUTINE STABLE *****
      C              *****************************
      C
      C          THE RELATIONSHIP BETWEEN DISTANCE INCREMENTS,TIME INCREMENTS,
      C     DISPERSION COEFFICIENTS, AND VELOCITIES IS NEEDED TO DETERMINE
      C     THE STABILITY OF THE EXPLICIT FINITE-DIFFERENCE PROCEDURE.  THIS
      C     SUBROUTINE CALCULATES THE MAXIMUM ALLOWABLE INCREMENTS FOR TIME
      C     AND DISTANCE.  THE PROGRAM IS TERMINATED IF THE INPUT PARAMETERS
      C     VIOLATE THIS CRITERIA.
      C
      C-----------------------------------------------------------------
2241        LOGICAL SECOND
2242        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
2243        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
2244        COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
2245        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
2246        COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
           1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
           2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
```

```
      STABLX = DELTAX
      STABLZ = DELTAZ
      STABLT = DELTAT
C
C-----------------------------------------------------------------------
C          CALCULATE STABILITY CRITERIA FCR ONE DIMENSIONAL CASE
C-----------------------------------------------------------------------
      IF(NZ.NE.1) GO TC 500
      IF(EXMIN.GT.0.0.AND.VXMAX.GT.0.0) STABLX=2.*EXMIN/VXMAX
      IF(STABLX.LT.DELTAX) ITER=0
      TERM=2.*EXMAX+DELTAX**2.*KDMAX
      IF(TERM.NE.0.0) STABLT=(DELTAX**2.)/TERM
      IF(STABLT.LT.DELTAT) ITER=0
      R1=DELTAT/(DELTAX**2.)
      R2=0.
      GO TO 800
  500 CONTINUE
C
C-----------------------------------------------------------------------
C          CALCULATE STABILITY CRITERIA FCR TWO DIMENSIONAL CASE
C-----------------------------------------------------------------------
      IF(EXMIN.GT.0.0.AND.VXMAX.GT.0.0) STABLX=2.*EXMIN/VXMAX
      IF(STABLX.LT.DELTAX) ITER=0
      IF(EZMIN.GT.0.0.AND.VZMAX.GT.0.0) STABLZ=2.*EZMIN/VZMAX
      IF(STABLZ.LT.DELTAZ) ITER=0
      TERM=2.*(EXMAX*DELTAZ**2.+EZMAX*DELTAX**2.)+KDMAX*(DELTAX**2)*
     $  (DELTAZ**2)
      IF(TERM.NE.0.0) STABLT=(DELTAZ**2.)*(DELTAX**2.)/TERM
      IF(STABLT.LT.DELTAT) ITER=0
      R1=DELTAT/(DELTAX**2.)
      R2=DELTAT/(DELTAZ**2.)
  800 CONTINUE
      RETURN
      END
```

```
C-----------------------------------------------------------------------
      SUBROUTINE ONEDEX
C
C           *****************************
C           ***** SUBROUTINE ONEDEX *****
C           *****************************
C
C          THIS SUBROUTINE CALCULATES THE ONE DIMENSIONAL CONCENTRATION
C     PROFILE FOR THE ASSIGNED GRID AND THE ASSIGNED NUMBER OF ITERA-
C     TIONS.  A FINITE-DIFFERENCE,EXPLICIT SCHEME IS USED TO SOLVE THE
C     PARTIAL DIFFERENTIAL EQUATION FOR MASS TRANSPORT.
C
C-----------------------------------------------------------------------
      LOGICAL SECOND
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
      COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
     1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
     2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
      Z=1
      NXM1=NX-1
```

```
C
C------------------------------------------------------------------
C           CALCULATE TERMS WHICH ARE USED REPEATEDLY
C------------------------------------------------------------------
      TWOXSQ=2.*DELTAX**2.
      TWOX=2.*DELTAX
C
C------------------------------------------------------------------
C           CALCULATE NEW CONCENTRATION AT EACH POINT
C------------------------------------------------------------------
      DO 300 IT=1,ITER
      TIME=TIME+DELTAT
      T=1
      TNEXT=2
   10 DO 200 X=2,NXM1
      DECAY=KD(X)
      IF(.NOT.SECOND) GO TO 15
      IF(C(X,Z,3).LE.0.) DECAY=AK(X)
   15 CONTINUE
      TERM1A=W(X)*VX(X,Z)/TWCX
      TERM1B=(W(X)*EX(X,Z)+W(X-1)*EX(X-1,Z))/TWOXSQ
      TERM2B = (W(X+1)*EX(X+1,Z) + W(X)*EX(X,Z)) / TWOXSQ
     $         +(W(X)*EX(X,Z) + W(X-1)*EX(X-1,Z)) / TWOXSQ
      TERM3A=(-W(X)*VX(X,Z))/TWOX
      TERM3B=(W(X+1)*EX(X+1,Z)+W(X)*EX(X,Z))/TWOXSQ
      TERM1=C(X-1,Z,T)*(TERM1A+TERM1B)
      TERM2=-C(X,Z,T)*TERM2B
      TERM3=C(X+1,Z,T)*(TERM3A+TERM3B)
      IF(T.EQ.3) GO TO 50
```

```
      C(X,Z,2)=C(X,Z,1)+(DELTAT/W(X)) *(TERM1+TERM2+TERM3)
     $          -DEMAND(X,Z)-DECAY*DELTAT*C(X,Z,1)
      GO TO 75
   50 CONTINUE
      C(X,Z,4)=          -DECAY*DELTAT*C(X,Z,4)-OXOUT(X,Z)
     $          +R(X,Z)*(OXSAT(X)-C(X,Z,3))*DELTAT
      C(X,Z,4)=C(X,Z,3)+(DELTAT/W(X))*(TERM1+TERM2+TERM3) +C(X,Z,4)
   75 IF(C(X,Z,TNEXT).LT.0.0) C(X,Z,TNEXT)=0.0
  200 CONTINUE
C
C-----------------------------------------------------------------------
C           CALCULATE BOUNDARY VALUES
C-----------------------------------------------------------------------
      CALL BOUND
      Z=1
      DO 250 X=1,NX
      IF(TNEXT.EQ.2) C(X,Z,4)=C(X,Z,1)
      C(X,Z,T)=C(X,Z,TNEXT)
      IF(C(X,Z,3).GT.OXSAT(X)) C(X,Z,3) = OXSAT(X)
      C(X,Z,TNEXT)=0.0
  250 CONTINUE
      IF(T.EQ.1) COUNT=COUNT+1
      IF(COUNT.GE.TPRINT) GO TO 292
      GO TO 298
C
C-----------------------------------------------------------------------
C           PRINT CONCENTRATIONS AT APPROPRIATE INTERVALS
C-----------------------------------------------------------------------
  292 CONTINUE
      IF(T.EQ.3.OR.(.NOT.SECOND)) COUNT=0
```

```
      IPAGE=IPAGE+1
      IF(IPAGE.GE.NPPAGE) GO TO 295
      GO TO 297
  295 PRINT 296
  296 FORMAT(1H1)
      IPAGE = 0
  297 CALL PRINT2
  298 CONTINUE
      IF((.NOT.SECOND).OR.(T.EQ.3)) GO TO 300
      T=3
      TNEXT=4
      GO TO 10
  300 CONTINUE
      RETURN
      END
```

```
C-----------------------------------------------------------------------
      SUBROUTINE TWODEX
C
C               ****************************
C               ***** SUBROUTINE TWODEX *****
C               ****************************
C
C          THIS SUBROUTINE CALCULATES THE TWO-DIMENSIONAL CONCENTRATION
C     PROFILE FOR THE ASSIGNED GRID AND THE ASSIGNED NUMBER OF ITERA-
C     TIONS.  A FINITE-DIFFERENCE,EXPLICIT SCHEME IS USED TO SOLVE THE
C     PARTIAL DIFFERENTIAL EQUATION FOR MASS TRANSPORT.
C
C-----------------------------------------------------------------------
      LOGICAL SECOND
      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
      COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
     1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
     2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
     1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
     2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
     3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
     4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
     5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
      COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
     1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
     2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
      NXM1=NX-1
      NZM1=NZ-1
      NZ1=2
```

```
C
C-------------------------------------------------------------------
C          CALCULATE TERMS WHICH ARE USED REPEATEDLY
C-------------------------------------------------------------------
      TWOXSQ=2.*DELTAX**2.
      TWOX=2.*DELTAX
      TWOZSQ=2.*DELTAZ**2.
      TWOZ=2.*DELTAZ
C
C-------------------------------------------------------------------
C          CALCULATE NEW CONCENTRATION AT EACH POINT
C-------------------------------------------------------------------
      DO 300 IT=1,ITER
      TIME=TIME+DELTAT
      T=1
      TNEXT=2
   10 DO 200 X=2,NXM1
      DO 100 Z=NZ1,NZM1
      DECAY=KD(X)
      IF(.NOT.SECOND) GO TO 15
      IF(C(X,Z,3).LE.0.) DECAY=AK(X)
   15 CONTINUE
      TERM1A=W(X)*VX(X,Z)/TWCX
      TERM1B=(W(X)*EX(X,Z)+W(X-1)*EX(X-1,Z))/TWOXSQ
      TERM2B = (W(X+1)*EX(X+1,Z) + W(X)*EX(X,Z)) / TWOXSQ
     $         +(W(X)*EX(X,Z) + W(X-1)*EX(X-1,Z)) / TWOXSQ
      TERM3A=(-W(X)*VX(X,Z))/TWOX
      TERM3B=(W(X+1)*EX(X+1,Z)+W(X)*EX(X,Z))/TWOXSQ
```

```
      TERM4A=W(X)*VZ(X,Z)/TWOZ
      TERM4B=(W(X)*EZ(X,Z)+W(X)*EZ(X,Z-1))/TWOZSQ
      TERM5A=(-W(X)*VZ(X,Z))/TWOZ
      TERM5B=(W(X)*EZ(X,Z+1)+W(X)*EZ(X,Z))/TWOZSQ
      TERM6B = (W(X) * EZ(X,Z+1) + W(X)*EZ(X,Z))/ TWOZSQ
     $          +(W(X)*EZ(X,Z)+ W(X)*EZ(X,Z-1)) / TWOZSQ
      TERM1=C(X-1,Z,T)*(TERM1A+TERM1B)
      TERM2=-C(X,Z,T)*(TERM2B+TERM6B)
      TERM3=C(X+1,Z,T)*(TERM3A+TERM3B)
      TERM4=C(X,Z-1,T)*(TERM4A+TERM4B)
      TERM5=C(X,Z+1,T)*(TERM5A+TERM5B)
      IF(T.EQ.3) GO TO 50
      C(X,Z,2)=C(X,Z,1)+(DELTAT/W(X))*(TERM1+TERM2+TERM3+TERM4+TERM5)
     $           -DEMAND(X,Z)-DECAY*DELTAT*C(X,Z,1)
      GO TO 75
   50 CONTINUE
      C(X,Z,4)=         -DECAY*DELTAT*C(X,Z,4)-OXOUT(X,Z)
     $           +R(X,Z)*(OXSAT(X)-C(X,Z,3))*DELTAT
      C(X,Z,4)=C(X,Z,3)+(DELTAT/W(X))*(TERM1+TERM2+TERM3+TERM4+TERM5)
     $  +C(X,Z,4)
   75 IF(C(X,Z,TNEXT).LT.0.0) C(X,Z,TNEXT)=0.0
  100 CONTINUE
  200 CONTINUE
C
C-----------------------------------------------------------------------
C           CALCULATE BOUNDARY VALUES
C-----------------------------------------------------------------------
      CALL BOUND
      DO 250 Z=1,NZ
      DO 250 X=1,NX
```

```
      IF(TNEXT.EQ.2) C(X,Z,4)=C(X,Z,1)
      C(X,Z,T)=C(X,Z,TNEXT)
      IF(C(X,Z,3).GT.OXSAT(X)) C(X,Z,3) = CXSAT(X)
      C(X,Z,TNEXT)=0.0
  250 CONTINUE
      IF(T.EQ.1) COUNT=COUNT+1
      IF(COUNT.GE.TPRINT) GO TO 292
      GO TO 298
C
C-----------------------------------------------------------------------
C           PRINT CONCENTRATIONS AT APPROPRIATE INTERVALS
C-----------------------------------------------------------------------
  292 CONTINUE
      IF(T.EQ.3.OR.(.NOT.SECOND)) COUNT=0
      IPAGE=IPAGE+1
      IF(IPAGE.GE.NPPAGE) GO TO 295
      GO TO 297
  295 PRINT 296
  296 FORMAT(1H1)
      IPAGE = 0
  297 CALL PRINT2
  298 CONTINUE
      IF((.NOT.SECOND).OR.(T.EQ.3)) GO TO 300
      T=3
      TNEXT=4
      GO TO 10
  300 CONTINUE
      RETURN
      END
```

```
C-----------------------------------------------------------------------
2412      SUBROUTINE BOUND
C
C                *****************************
C                ***** SUBROUTINE BOUND *****
C                *****************************
C
C            SUBROUTINE BOUND PROVIDES FOR EXTRAPOLATION OF CONCENTRATIONS
C         AT THE BOUNDARIES.  THE TYPE OF EXTRAPOLATION DEPENDS ON THE VALUE
C         OF THE VARIABLE OPTION.
C
C-----------------------------------------------------------------------
2413      LOGICAL SECOND
2414      INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
2415      REAL KD,KDMAX,KDMIN,KDAYS,LOWER
2416      COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
         1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
         2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
2417      COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
         1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
         2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
         3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
         4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
         5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
2418      COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
         1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
         2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
```

```
      IF(NZ.EQ.1) NZM1=1
      ZUP=UPPER
      ZLOW=LOWER
      ZSF=XSF
      GO TO (1001,1002,1003,1004,1005,1006,1006), OPTION
C
C-----------------------------------------------------------------------
C        EXPONENTIAL EXTRAPOLATICN
C-----------------------------------------------------------------------
 1003 CONTINUE
      IF(NZ.EQ.1) GO TO 1001
      DO 140 X=2,NXM1
      IF(C(X,2,T+1).LE.0.0) GO TO 140
      VZ1=VZ(X,1)
      ZZ=-ZUP+DELTAZ+ZSF+VZ1*TSTART*86400.
      EZ1=EZ(X,1)
      C2=4.*EZ1*TIME
      EXPON=( 2.*ZZ*DELTAZ-(DELTAZ*DELTAZ)-2.*DELTAZ*VZ1*TIME)/C2
      C(X,1,T+1)=C(X,2,T+1)*EXP(EXPON)
  140 CONTINUE
      DO 190 X=2,NXM1
      IF(C(X,NZM1,T+1).LE.0.0) GO TO 190
      VZNZ=VZ(X,NZ)
      ZZ=(NZ-2)*DELTAZ-ZLOW+ZSF+VZNZ*TSTART*86400.
      EZNZ=EZ(X,NZ)
      C2=4.*EZNZ*TIME
      EXPON=(-2.*ZZ*DELTAZ-(DELTAZ*DELTAZ)+2.*DELTAZ*VZNZ*TIME)/C2
      C(X,NZ,T+1)=C(X,NZM1,T+1)*EXP(EXPON)
  190 CONTINUE
 1001 CONTINUE
      IF(LTYPE.EQ.1.AND.MSU.EQ.1) GO TO 350
      DO 340 Z=1,NZ
```

```
      IF(C(2,Z,T+1).LE.0.0) GO TO 340
      VX1=VX(1,Z)
      XX=-UPPER+DELTAX+XSF+VX1*TSTART*86400.
      EX1=EX(1,Z)
      C2=4.*EX1*TIME
      EXPON=( 2.*XX*DELTAX-(DELTAX*DELTAX)-2.*DELTAX*VX1*TIME)/C2
      C(1,Z,T+1)=C(2,Z,T+1)*EXP(EXPON)
  340 CONTINUE
  350 CONTINUE
      IF(LTYPE.EQ.1.AND.MSL.EQ.NZ) GO TO 400
      DO 390 Z=1,NZ
      IF(C(NX-1,Z,T+1).LE.0.0) GO TO 390
      VXNX=VX(NX,Z)
      XX=(NX-2)*DELTAX-LOWER+XSF+VXNX*TSTART*86400.
      EXNX=EX(NX,Z)
      C2=4.*EXNX*TIME
      EXPON=(-2.*XX*DELTAX-(DELTAX*DELTAX)+2.*DELTAX*VXNX*TIME)/C2
      C(NX,Z,T+1)=C(NX-1,Z,T+1)*EXP(EXPON)
  390 CONTINUE
  400 CONTINUE
      GO TO 2000
C
C-----------------------------------------------------------------------
C         CONSTANT SLOPE EXTRAPCLATION
C-----------------------------------------------------------------------
 1004 CONTINUE
      IF(NZ.EQ.1) GO TC 1002
      DO 540 X=2,NXM1
      C(X,1,T+1)=2.*C(X,2,T+1)-C(X,3,T+1)
      IF(C(X,1,T+1).LT.0.0) C(X,1,T+1)=0.0
```

```
      C(X,NZ,T+1)=2.*C(X,NZM1,T+1)-C(X,NZM2,T+1)
      IF(C(X,NZ,T+1).LT.0.0) C(X,NZ,T+1)=0.0
  540 CONTINUE
 1002 CONTINUE
      DO 590 Z=1,NZ
      C(1,Z,T+1)=2.*C(2,Z,T+1)-C(3,Z,T+1)
      IF(C(1,Z,T+1).LT.0.0) C(1,Z,T+1)=0.0
      C(NX,Z,T+1)=2.*C(NXM1,Z,T+1)-C(NXM2,Z,T+1)
      IF(C(NX,Z,T+1).LT.0.0) C(NX,Z,T+1)=0.0
  590 CONTINUE
      GO TO 2000
 1005 CONTINUE
C
C-----------------------------------------------------------------------
C         THIS OPTION ALLOWS THE PROGRAM USER TO SUBSTITUTF HIS OWN
C         EXTRAPOLATION ROUTINE
C-----------------------------------------------------------------------
      ZZ=-40.
      EZ1=0.025
      VZ1=0.
      C2=4.*EZ1*TIME
      EXPON=(+2.*ZZ*DELTAZ-(DELTAZ**2.)-2*DELTAZ*VZ1*TIME)/C2
      DO 240 X=1,NX
      C(X,1,T+1)=C(X,2,T+1)*EXP(EXPON)
      C(X,NZ,T+1)=C(X,1,T+1)
  240 CONTINUE
      GO TO 1001
C
C-----------------------------------------------------------------------
C           INVERTED DIFFFRENCES EXTRAPOLATION (ACFI)
```

```
C-----------------------------------------------------------------------
 1006 CALL EXTRAP
      GO TO 2000
 2000 CONTINUE
      IF(NZ.EQ.1) GO TO 2005
      IF(OPTION.EQ.3.OR.OPTION.EQ.4.OR.OPTION.EQ.5)  GO TO 2005
      DO 700 X=1,NX
      C(X,1,T+1)=C(X,2,T+1)
      C(X,NZ,T+1)=C(X,NZM1,T+1)
  700 CONTINUE
 2005 CONTINUE
C
C-----------------------------------------------------------------------
C         IF LTYPE EQUALS 1, SET CONCENTRATIONS AT SOURCE POINTS.
C         IF LTYPE EQUALS 3, THE USER CAN INCLUDE A SPECIAL SUBROUTINE
C         OR SET OF CALCULATIONS IN THIS PART OF THE PROGRAM.
C-----------------------------------------------------------------------
      IF(T.EQ.3) GO TO 290
      IF(LTYPE.EQ.1.AND.NS.LE.2) GO TO 275
      IF(LTYPE.EQ.1.AND.NS.GT.2) GO TO 281
      GO TO 290
  275 CONTINUE
      DO 280 Z=1,NZ
      C(MSU,Z,T+1)=C(MSU,Z,T)
      C(MSL,Z,T+1)=C(MSL,Z,T)
  280 CONTINUE
      GO TO 290
  281 CONTINUE
      DO 285 I=1,NC
      DO 285 Z=1,NZ
      C(INSECT(I),Z,T+1)=CONCIN(I,Z)
  285 CONTINUE
  290 CONTINUE
      RETURN
      END
```

```
     C------------------------------------------------------------------------
2523       SUBROUTINF EXTRAP
     C
     C              ****************************
     C              ***** SUBROUTINE EXTRAP ****
     C              ****************************
     C
     C          THIS SUBROUTINE EXTRAPOLATES THE CONCENTRATION PROFILE BY
     C     USING A CONTINUED FRACTIONS AND INVERTED DIFFERENCES SCHEME. IT IS
     C     A MODIFIED VERSION OF THE IBM SCIENTIFIC SUBPROGRAM ACFI.
     C
     C          IF THE EXTRAPOLATED CONCENTRATION IS GREATER THAN THE
     C     ADJACENT CONCENTRATION OR IF IT IS NEGATIVE, THEN THE BOUNDARY
     C     CONCENTRATION IS EXTRAPOLATED ACCORDING TO THE PROPORTION
     C     BETWEEN THE CONCENTRATIONS AT THE TWO ADJACENT INTERNAL POINTS
     C     (I.E.  C1/C2 = C2/C3 ).
     C
     C------------------------------------------------------------------------
2524       LOGICAL SFCOND
2525       INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
2526       REAL KD,KDMAX,KDMIN,KDAYS,LOWER
2527       COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
          1               W(51),CCNCIN(51,21),INSECT(51),KDAYS(51),KD(51),
          2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
2528       COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
          1               KDMAX,KCMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
          2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
          3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
          4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
          5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTICN
2529       COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
          1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
          2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
```

```
      DIMENSION  ARG(10), VALY(10)
      NZ1=2
      IF(NZ.EQ.1) NZ1=1
      IF(NZ.EQ.1)NZM1=1
      IF(MSU.EQ.1.AND.LTYPE.EQ.1) GO TO 228
C
C-----------------------------------------------------------------------
C          EXTRAPOLATE THE PROFILE IN THE X DIRECTION
C-----------------------------------------------------------------------
      DO 225 Z=NZ1,NZM1
      IF(C(2, Z, T+1) . LE. 0.00) GO TO 220
C
C-----------------------------------------------------------------------
C          CHOOSE THE NUMBER OF POINTS (NDIM) TO BE USED IN THE
C          EXTRAPOLATION.
C-----------------------------------------------------------------------
      NDIM=3
      NSTOP=NDIM+1
C
C-----------------------------------------------------------------------
C          PLACE CONCENTRATIONS IN PROPER ORDER FOR SUBROUTINE ACFI
C-----------------------------------------------------------------------
      DO 210 X=2,NSTOP
      ARG(X - 1) = FLOAT(X)
      VALY(X-1) = C(X, Z, T+1)
      IF(VALY(X-1).LE.0.0) GO TO 205
```

```
      GO TO 210
  205 NDIM = X-1
      GO TO 211
  210 CONTINUE
  211 CONTINUE
      EPS = VALY(1)/1000.
      CALL ACFI   ( 1.0, ARG, VALY, Y, NDIM, EPS, IER, J )
      C(1, Z, T+1)=Y
C
C-----------------------------------------------------------------------
C             IF THE EXTRAPOLATED VALUE IS TOO HIGH OR IS NEGATIVE,THEN USE
C             A PROPORTION EQUATION.
C-----------------------------------------------------------------------
      IF(Y.LE.0.0.OR.Y.GT.C(2,Z,T+1)) C(1,Z,T+1)=C(2,Z,T+1)*C(2,Z,T+1)
     $  /C(3,Z,T+1)
      GO TO 225
  220 C(1, Z, T+1) = 0.0
  225 CONTINUE
  228 CONTINUE
      IF(LTYPE.EQ.1.AND.MSL.EQ.NX) GO TO 251
      DO 250 Z=NZ1,NZM1
      IF ( C(NX-1, Z, T+1) . LE. 0.00) GO TO 245
      NDIM=3
      NSTOP=NDIM
      DO 235 I=1,NSTOP
      IBACK = NX - I
      ARG(I) = FLOAT(I+1)
      VALY(I) = C( IBACK, Z, T+1)
      IF( VALY(I) . LE. 0.00) GO TO 230
      GO TO 235
```

```
  230 NDIM = I
      GO TO 236
  235 CONTINUE
  236 CONTINUE
      EPS = VALY(1)/1000.
      CALL ACFI   ( 1.0, ARG, VALY, Y, NDIM, EPS, IER, J )
      C( NX, Z, T+1) = Y
      IF(Y.LE.0.0.OR.Y.GT.C(NX-1,Z,T+1))C(NX,Z,T+1)=C(NXM1,Z,T+1)*
     $  C(NXM1,Z,T+1)/C(NXM2,Z,T+1)
      GO TO 250
  245 C(NX, Z, T+1) = 0.0
  250 CONTINUE
  251 CONTINUE
      IF(NZ.EQ.1.OR.OPTION.EQ.6) RETURN
C
C-----------------------------------------------------------------------
C           EXTRAPOLATE THE PROFILE IN THE Z DIRECTION
C-----------------------------------------------------------------------
      DO 275 X = 1, NX
      IF(C(X, 2, T+1) . LE. 0.00) GO TO 270
      NDIM=3
      NSTOP=NDIM
      DO 260 Z = 1, NSTOP
       ARG(Z) = FLOAT(Z+1)
      VALY(Z) = C(X, Z+1, T+1)
      IF( VALY(Z) . LE. 0.00) GO TO 255
      GO TO 260
  255 NDIM = Z
      GO TO 261
  260 CONTINUE
```

```
  261 CONTINUE
      EPS = VALY(1)/1000.
      CALL ACFI   ( 1.0, ARG, VALY, Y, NDIM, EPS, IER, J )
      C ( X, 1, T+1) = Y
      IF(Y.LE.0.0.OR.Y.GT.C(X,2,T+1))C(X,1,T+1)=C(X,2,T+1)*C(X,2,T+1)
     $  /C(X,3,T+1)
      IF(C(X,1,T+1).LT.0.0) C(X,1,T+1)=0.0
      GO TO 275
  270 C(X, 1, T+1) = 0.00
  275 CONTINUE
      DO 295 X = 1, NX
      IF (C(X, NZ-1, T+1) . LE. 0.00) GO TO 292
      NDIM=3
      NSTOP=NDIM
      DO 285 I = 1, NSTOP
      ARG(I) = FLOAT(I+1)
      IBACK = NZ - I
      VALY(I) = C(X, IBACK, T+1)
      IF(VALY(I). LE. 0.00) GO TO 280
      GO TO 285
  280 NDIM = I
      GO TO 286
  285 CONTINUE
  286 CONTINUE
      EPS = VALY(1)/ 1000.
      CALL ACFI   ( 1.0, ARG, VALY, Y, NDIM, EPS, IER, J )
      C(X, NZ, T+1) = Y
      IF(Y.LE.0.0.OR.Y.GT.C(X,NZ-1,T+1)) C(X,NZ,T+1)=C(X,NZM1,T+1)*
     $  C(X,NZM1,T+1)/C(X,NZ*2,T+1)
      GO TO 295
  292 C(X, NZ, T+1)=0.0
  295 CONTINUE
      RETURN
      END
```

```
C-----------------------------------------------------------------------
2624      SUBROUTINE ACFI   ( X, ARG, VALY, Y, NDIM, EPS, IER, J )
C
C               ****************************
C               ****** SUBROUTINE ACFI *****
C               ****************************
C
C     .................................................................
C
C        PURPOSE
C           TO INTERPOLATE FUNCTION VALUE Y FOR A GIVEN ARGUMENT VALUE
C           X USING A GIVEN TABLE (ARG,VAL) OF ARGUMENT AND FUNCTION
C           VALUES.
C
C        DESCRIPTION OF PARAMETERS
C           X      - THE ARGUMENT VALUE SPECIFIED BY INPUT.
C           ARG    - THE INPUT VECTOR (DIMENSION NDIM) OF ARGUMENT
C                    VALUES OF THE TABLE (POSSIBLY DESTROYED).
C           VAL    - THE INPUT VECTOR (DIMENSION NDIM) OF FUNCTION
C                    VALUES OF THE TABLE (DESTROYED).
C           Y      - THE RESULTING INTERPOLATED FUNCTION VALUE.
C           NDIM   - AN INPUT VALUE WHICH SPECIFIES THE NUMBER OF
C                    POINTS IN TABLE (ARG,VAL).
C           EPS    - AN INPUT CONSTANT WHICH IS USED AS UPPER BOUND
C                    FOR THE ABSOLUTE ERROR.
C           IER    - A RESULTING ERROR PARAMETER.
C
```

```
C        REMARKS
C           (1) TABLE (ARG,VAL) SHOULD REPRESENT A SINGLE-VALUED
C               FUNCTION AND SHOULD BE STORED IN SUCH A WAY, THAT THE
C               DISTANCES ABS(ARG(I)-X) INCREASE WITH INCREASING
C               SUBSCRIPT I. TO GENERATE THIS ORDER IN TABLE (ARG,VAL),
C               SUBROUTINES ATSG, ATSM OR ATSE COULD BE USED IN A
C               PREVIOUS STAGE.
C           (2) NO ACTION BESIDES ERROR MESSAGE IN CASE NDIM LESS
C               THAN 1.
C           (3) INTERPOLATION IS TERMINATED EITHER IF THE DIFFERENCE
C               BETWEEN TWO SUCCESSIVE INTERPOLATED VALUES IS
C               ABSOLUTELY LESS THAN TOLERANCE EPS, OR IF THE ABSOLUTE
C               VALUE OF THIS DIFFERENCE STOPS DIMINISHING, OR AFTER
C               (NDIM-1) STEPS (THE NUMBER OF POSSIBLE STEPS IS
C               DIMINISHED IF AT ANY STAGE INFINITY ELEMENT APPEARS IN
C               THE DOWNWARD DIAGONAL OF INVERTED-DIFFERENCES-SCHEME
C               AND IF IT IS IMPOSSIBLE TO ELIMINATE THIS INFINITY
C               ELEMENT BY INTERCHANGING OF TABLE POINTS).
C               FURTHER IT IS TERMINATED IF THE PROCEDURE DISCOVERS TWO
C               ARGUMENT VALUES IN VECTOR ARG WHICH ARE IDENTICAL.
C               DEPENDENT ON THESE FOUR CASES, ERROR PARAMETER IER IS
C               CODED IN THE FOLLOWING FORM
C                IER=0 - IT WAS POSSIBLE TO REACH THE REQUIRED
C                        ACCURACY (NO ERROR).
C                IER=1 - IT WAS IMPOSSIBLE TO REACH THE REQUIRED
C                        ACCURACY BECAUSE OF ROUNDING ERRORS.
C                IER=2 - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE
C                        NDIM IS LESS THAN 2, OR THE REQUIRED ACCURACY
C                        COULD NOT BE REACHED BY MEANS OF THE GIVEN
C                        TABLE. NDIM SHOULD BE INCREASED.
C                IER=3 - THE PROCEDURE DISCOVERED TWO ARGUMENT VALUES
C                        IN VECTOR ARG WHICH ARE IDENTICAL.
```

```
C
C          METHOD
C             INTERPOLATION IS DONE BY CONTINUED FRACTIONS AND INVERTED-
C             DIFFERENCES-SCHEME. ON RETURN Y CONTAINS AN INTERPOLATED
C             FUNCTION VALUE AT POINT X, WHICH IS IN THE SENSE OF REMARK
C             (3) OPTIMAL WITH RESPECT TO GIVEN TABLE. FOR REFERENCE, SEE
C             F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,
C             MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.395-406.
C
C        ..................................................................
C
C
C
         DIMENSION ARG(10), VALY(10), VAL(10)
         DO 100 M = 1,NDIM
         VAL(M) = VALY(M)
     100 CONTINUE
         IER=2
         IF(NDIM)20,20,1
       1 Y=VAL(1)
         DELT2=0.
         IF(NDIM-1)20,20,2
C
C        PREPARATIONS FOR INTERPOLATICN LOOP
       2 P2=1.
         P3=Y
         Q2=0.
         Q3=1.
C
```

```
C
C     START INTERPOLATION LOCP
      DO 16 I=2,NDIM
      II=0
      P1=P2
      P2=P3
      Q1=Q2
      Q2=Q3
      Z=Y
      DELT1=DELT2
      JEND=I-1
C
C     COMPUTATION OF INVERTEC DIFFERENCES
    3 AUX=VAL(I)
      DO 10 J=1,JEND
      H=VAL(I)-VAL(J)
      IF(ABS(H)-1.E-6*ABS(VAL(I)))4,4,9
    4 IF(ARG(I)-ARG(J))5,17,5
    5 IF(J-JEND)8,6,6
C
C     INTERCHANGE ROW I WITH ROW I+II
    6 II=II+1
      III=I+II
      IF(III-NDIM)7,7,19
    7 VAL(I)=VAL(III)
      VAL(III)=AUX
      AUX=ARG(I)
      ARG(I)=ARG(III)
      ARG(III)=AUX
      GOTO 3
C
```

```
C       COMPUTATION OF VAL(I) IN CASE VAL(I)=VAL(J) AND J LESS THAN I-1
      8 VAL(I)=1.E75
        GOTO 10
C
C       COMPUTATION OF VAL(I) IN CASE VAL(I) NOT EQUAL TO VAL(J)
      9 VAL(I)=(ARG(I)-ARG(J))/H
     10 CONTINUE
C       INVERTED DIFFERENCES ARE COMPUTED
C
C       COMPUTATION OF NEW Y
        P3=VAL(I)*P2+(X-ARG(I-1))*P1
        Q3=VAL(I)*Q2+(X-ARG(I-1))*Q1
        IF(Q3)11,12,11
     11 Y=P3/Q3
        GOTO 13
     12 Y=1.E75
     13 DELT2=ABS(Z-Y)
        IF(DELT2-EPS)19,19,14
     14 IF(I-8)16,15,15
     15 IF(DELT2-DELT1)16,18,18
     16 CONTINUE
C       END OF INTERPOLATICN LCOP
C
C
        RETURN
C
C       THERE ARE TWO IDENTICAL ARGUMENT VALUES IN VECTOR ARG
     17 IER=3
```

```
      RETURN
C
C     TEST VALUE DELT2 STARTS OSCILLATING
   18 Y=Z
      IER=1
      RETURN
C
C     THERE IS SATISFACTORY ACCURACY WITHIN NDIM-1 STEPS
   19 IER=0
   20 RETURN
      END
```

```
      C-----------------------------------------------------------------------
2686        SUBROUTINE PRINT1
      C
      C             *****************************
      C             ***** SUBROUTINE PRINT1 *****
      C             *****************************
      C
      C           THIS SUBROUTINE CALCULATES THE CONVERSION VALUES FOR MANY OF
      C     THE INPUT PARAMETERS AND PRINTS OUT A SUMMARY OF THE INPUT DATA
      C     AND STABILITY CRITERIA.
      C
      C-----------------------------------------------------------------------
2687        LOGICAL SECOND
2688        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
2689        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
2690        COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
2691        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
2692        COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
           1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
           2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
2693        UNITS1 = 86400./5280.
2694        UNITS2 = 86400./(5280.*5280.)
2695        DELDAY = DELTAT/86400.
```

```
      SECS1=DELT1*86400.
      SECS2=DELT2*86400.
      DELMX=DELTAX/5280.
      DELMZ=DELTAZ/5280.
      TOTALX=(NX-1)*DELTAX
      TOTMX=TOTALX/5280.
      TOTALZ=(NZ-1)*DELTAZ
      TOTMZ=TOTALZ/5280.
      VX1=VXMAX*UNITS1
      VX2=VXMIN*UNITS1
      VZ1=VZMAX*UNITS1
      IF(VZMIN.EQ.1000000.) VZMIN=0.
      VZ2=VZMIN*UNITS1
      EX1=EXMAX*UNITS2
      EX2=EXMIN*UNITS2
      EZ1=EZMAX*UNITS2
      IF(EZMIN.EQ.1000000.) EZMIN=0.
      EZ2=EZMIN*UNITS2
      XKD1=KDMAX*86400.
      XKD2=KDMIN*86400.
      XAK1=AKMAX*86400.
      XAK2=AKMIN*86400.
      XR1=RMAX*86400.
      XR2=RMIN*86400.
      SXM=STABLX/5280.
      SZM=STABLZ/5280.
      STD=STABLT/86400.
      PRINT 500
      PRINT 505
```

```
      PRINT 507
      PRINT 510
      PRINT 507
      IF(NZ.EQ.1) PRINT 515
      IF(NZ.NE.1) PRINT 520
      PRINT 507
      IF(LTYPE.EQ.1) GO TO 7
      IF(LTYPE.EQ.2) GO TO 5
      PRINT 530
      GO TO 10
    5 PRINT 525
      GO  TO 10
    7 PRINT 523
   10 CONTINUE
      PRINT 507
      PRINT 535
      PRINT 507
      PRINT 505
      PRINT 540, DELTAX, DELMX
      PRINT 545, DELTAZ,DELMZ
      PRINT 550, SECS1, DELT1
      PRINT 551, ITER1
      PRINT 552, SECS2, DELT2
      PRINT 553, ITER2
      PRINT 555,NX
      PRINT 560, NZ
      PRINT 565, VXMAX, VX1
      PRINT 570, VXMIN, VX2
      PRINT 572,VZMAX,VZ1
      PRINT 573,VZMIN,VZ2
```

```
      PRINT 575,EXMAX,EX1
      PRINT 580,EXMIN,EX2
      PRINT 585, EZMAX,EZ1
      PRINT 590, EZMIN,EZ2
      PRINT 600,WMAX
      PRINT 605,WMIN
      PRINT 610,XKD1,KDMAX
      PRINT 615,XKD2,KDMIN
      PRINT 620,DEMAX
      PRINT 630, NC
C
C------------------------------------------------------------------
C        XUPEAK AND XLPEAK ARE CALCULATED ASSUMING DISTANCES ARE
C     DECREASING IN THE DOWNSTREAM DIRECTION
C------------------------------------------------------------------
      XUPEAK = XSTART - (MSU-1)*DELMX
      XLPEAK = XSTART - (MSL-1)*DELMX
      PRINT 635, XUPEAK,MSU
      PRINT 637, XLPEAK,MSL
      PRINT 500
      IF(.NOT.SECOND) GO TO 20
      PRINT 800
      PRINT 810, XR1,RMAX
      PRINT 815, XR2, RMIN
      PRINT 820, XAK1,AKMAX
      PRINT 825, XAK2,AKMIN
      PRINT 830, OUTMAX
   20 CONTINUE
      PRINT 505
      PRINT 502
```

```
      PRINT 640
      PRINT 645,STABLX,SXM
      PRINT 650,STABLZ,SZM
      PRINT 655,STABLT,STD
      PRINT 660,R1
      PRINT 665, R2
      PRINT 505
      IF(NC.EQ.0) GO TO 100
      PRINT 500
      PRINT 700
      IF(NZ.EQ.1) GO TO 50
      DO 25 I=1,NC
C
C-----------------------------------------------------------------------
C           WHERE IS CALCULATED ASSUMING DISTANCES ARE DECREASING IN
C     THE DOWNSTREAM DIRECTION
C-----------------------------------------------------------------------
      WHERE=XSTART-(INSECT(I)-1)*DELMX
      PRINT 710, INSECT(I),WHERE,DELTAZ
      DO 25 Z=1,NZ
      PRINT 715,INSECT(I),Z,CONCIN(I,Z)
   25 CONTINUE
      GO TO 100
   50 DO 75 I=1,NC
      WHERE=XSTART-(INSECT(I)-1)*DELMX
      PRINT 705, INSECT(I), WHERE, CONCIN(I,1)
      PRINT 502
   75 CONTINUE
  100 CONTINUE
      PRINT 500
```

```
      500 FORMAT(1H1)
      502 FORMAT(/)
      505 FORMAT(T45, 38('*'))
      507 FORMAT(T45,'*',T82,'*')
      510 FORMAT(T45,'*',T56,'ESTUARY SIMULATION', T82, '*')
      515 FORMAT(T45, '*', T48, 'ONE DIMENSIONAL EXPLICIT METHOD',T82,'*')
      520 FORMAT(T45, '*', T48, 'TWO DIMENSIONAL EXPLICIT METHOD',T82,'*')
      523 FORMAT(T45, '*', T53, 'CONSTANT CONCENTRATION',T82,'*')
      525 FORMAT(T45, '*', T54, 'INSTANTANEOUS RELEASE', T82,'*')
      530 FORMAT(T45, '*', T56, 'CONTINUOUS LOADING', T82, '*')
      535 FORMAT(T45, '*', T51, 'PROGRAMMER -- JONATHAN YOUNG', T82,'*')
      540 FORMAT(   /, T53, 'X INCREMENT =', F7.0, ' FEET (', F8.5, ' MILES
         $)',/)
      545 FORMAT(T53, 'Z INCREMENT =', F7.0, ' FEET (', E10.3, ' MILES )'/)
      550 FORMAT(T42, 'INITIAL TIME INCREMENT =',F7.0, ' SECONDS (', F6.4,
         $  ' DAYS )'/)
      551 FORMAT(T44,'NUMBER OF ITERATIONS = ',I5,/)
      552 FORMAT( T42, 'REVISED TIME INCREMENT = ', F7.0, ' SECONDS(',F6.4,
         $  ' DAYS )'/)
      553 FORMAT(T44,'NUMBER OF ITERATIONS = ',I5,/)
      555 FORMAT(T37, 'NUMBER OF HORIZONTAL POINTS =', I4,/)
      560 FORMAT(T39, 'NUMBER OF VERTICAL POINTS =',I4,/)
      565 FORMAT(T37, 'MAXIMUM HORIZONTAL VELOCITY = ', F5.2, ' FEET/SECOND
         $(',F6.2, ' MILES/DAY )',/)
      570 FORMAT(T37, 'MINIMUM HORIZONTAL VELOCITY = ', F5.2, ' FEET/SECOND
         $(',F6.2, ' MILES/DAY )',/)
      572 FORMAT(T39, 'MAXIMUM VERTICAL VELOCITY =', E10.3, ' FEET/SECOND ('
         $ , E10.3, ' MILES/DAY )',/)
      573 FORMAT(T39, 'MINIMUM VERTICAL VELOCITY =', E10.3, ' FEET/SECOND ('
         $ , E10.3, ' MILES/DAY )',/)
```

```
  575 FORMAT(T35, 'MAXIMUM HORIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
     $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
  580 FORMAT(T35, 'MINIMUM HORIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
     $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
  585 FORMAT(T37, 'MAXIMUM VERTICAL DISPERSION = ', F9.3, ' FEET SQUARED
     $ / SECOND ( ', E10.3, ' MILES SQUARED / DAY )',/)
  590 FORMAT(T37, 'MINIMUM VERTICAL DISPERSION = ', F9.3, ' FEET SQUARED
     $ / SECOND ( ', E10.3, ' MILES SQUARED / DAY )',/)
  600 FORMAT(T51, 'MAXIMUM WIDTH = ', F5.0, ' FEET',/)
  605 FORMAT(T51, 'MINIMUM WIDTH = ', F5.0, ' FEET',/)
  610 FORMAT(T46, 'MAXIMUM DECAY RATE = ', F5.3, ' PER DAY ( ', E10.3,
     $ ' PER SECOND )', /)
  615 FORMAT(T46, 'MINIMUM DECAY RATE = ', F5.3, ' PER DAY ( ', E10.3,
     $ ' PER SECOND )', /)
  620 FORMAT(T42, 'OTHER DEMANDS, MAXIMUM = ', F5.2,/)
  630 FORMAT(T26, 'NUMBER OF INITIAL CONCENTRATION VALUES = ',I2,/)
  635 FORMAT(T16, 'INITIAL LOCATION OF UPPERMOST PEAK CONCENTRATION = ',
     $  F5.2,' MILES (SECTION NUMBER ', I2,' )',/)
  637 FORMAT(T16, 'INITIAL LCCATION CF LOWERMOST PEAK CONCENTRATION = ',
     $  F5.2,' MILES (SECTION NUMBER ', I2,' )',/)
  640 FORMAT(T56, 'STABILITY CRITERIA',/)
  645 FORMAT(T35, 'MAXIMUM ALLOWABLE X INCREMENT = ', F7.0, ' FEET (',
     $ F8.5, 'MILES )',/)
  650 FORMAT(T35, 'MAXIMUM ALLOWABLE Z INCREMENT = ', F7.0, ' FEET (',
     $ F8.5, 'MILES )',/)
  655 FORMAT(T32, 'MAXIMUM ALLOWABLE TIME INCREMENT = ', F7.0, ' SECONDS
     $ (', E10.3, ' DAYS )',/)
  660 FORMAT(T38, 'ACTUAL DELTAT/(DELTAX**2.) = ', E10.3,/)
  665 FORMAT(T38, 'ACTUAL DELTAT/(DELTAZ**2.) = ', E10.3,/)
```

```
  700 FORMAT(T25, 'LOCATIONS OF INITIAL CONCENTRATIONS',//)
  705 FORMAT(     'A WASTE SOURCE IS LOCATED AT STATION ', I2, ' (MILE ',
     $ F6.3 , ' ).  THE CONCENTRATION IS', F10.2, ' PPM.')
  710 FORMAT( 1X,'AN INITIAL CONCENTRATION IS FOUND AT STATION ',I2,
     1  ' (MILE ', F6.3, ' ).',/, ' THE CONCENTRATIONS AT ', F7.1,
     2  ' FOOT INTERVALS WITH DEPTH ARE')
  715 FORMAT(T15, 'C(', I2','I2',1) = ', F9.2, ' PPM')
  800 FORMAT(T45,'*** SECONDARY INPUT PARAMETERS ***',//)
  810 FORMAT(T41,'MAXIMUM REAERATION RATE = ',
     $        F5.3, ' PER DAY ( ',E10.3, ' PER SECOND )',/)
  815 FORMAT(T41,'MINIMUM REAERATION RATE = ',
     $        F5.3, ' PER DAY ( ',E10.3, ' PER SECOND )',/)
  820 FORMAT(T41, 'MAXIMUM ANAEROBIC DECAY = ',
     $        F5.3, ' PER DAY ( ',E10.3, ' PER SECOND )',/)
  825 FORMAT(T41, 'MINIMUM ANAEROBIC DECAY = ',
     $        F5.3, ' PER DAY ( ',E10.3, ' PER SECOND )',/)
  830 FORMAT(T35, 'OTHER OXYGEN DEMANDS, MAXIMUM = ', F6.3,/////)
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
            SUBROUTINE PRINT2
      C
      C              *****************************
      C              ***** SUBRCUTINE PRINT2 *****
      C              *****************************
      C
      C          THIS SUBROUTINE PRINTS OUT THE TIME AND THE CONCENTRATION
      C     PROFILE ACCORDING TO A FORMAT PREVIOUSLY READ INTO THE PROGRAM.
      C     THIS SUBROUTINE CAN BE CHANGED ACCORDING TO THE NEEDS OF THE USER.
      C     ***WARNING--DO NOT CHANGE ANY VARIABLE OCCURING IN A COMMON BLOCK.
      C
      C-----------------------------------------------------------------------
            LOGICAL SECOND
            INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,CCUNT,OPTION
            REAL KD,KDMAX,KDMIN,KDAYS,LOWER
            DIMENSION NZREV(20)
            COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
            COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
            COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
           1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
           2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
```

```
      DAYS = TIME/86400.
      HOURS=DAYS*24.
      XFIRST=XSTART-(NX-1)*DELTAX/5280.
      DO 100 IZ = 1, NZ
      NZREV(IZ)=IFIX(DELTAZ*(IZ-2))
  100 CONTINUE
      IF(T.EQ.3) GO TO 300
      WRITE(6,FMT1) HOURS,DAYS
      WRITE(6,FMT2) (NZREV(I),I=2,NZM1)
      DO 200 X=1,NX,XPRINT
      XMILE = XFIRST + ((X-1)*DELTAX*XPRINT)/5280.
      M=NX+1-X
      WRITE(6,FMT3) XMILE, (C(M,Z,T),Z=1,NZ)
  200 CONTINUE
      GO TO 600
  300 CONTINUE
      WRITE(6,FMT4) HCURS, DAYS
      WRITE(6,FMT5) (NZREV(I), I=2,NZM1)
      DO 500 X=1,NX,XPRINT
      XMILE = XFIRST + ((X-1)*DELTAX*XPRINT)/5280.
      M=NX+1-X
      WRITE(6,FMT6) XMILE, (C(M,Z,T),Z=1,NZ)
  500 CONTINUE
  600 CONTINUE
      RETURN
      END
```

```
      C-----------------------------------------------------------------------
2892        SUBROUTINE ERROR(*)
      C
      C            ***************************
      C           **** SUBROUTINE ERROR ******
      C            ***************************
      C
      C          THIS SUBROUTINE CORRECTS SCME OF THE MOST COMMON ERRORS
      C     IN THE INPUT DATA AND PRINTS OUT APPROPRIATE ERROR MESSAGES.
      C
      C-----------------------------------------------------------------------
2893        LOGICAL SECOND
2894        INTEGER X,Z,T,TNEXT,TPRINT,XPRINT,COUNT,OPTION
2895        REAL KD,KDMAX,KDMIN,KDAYS,LOWER
2896        COMMON/ARRAYS/ C(51,21,4),VX(51,21),VZ(51,21),EX(51,21),EZ(51,21),
           1               W(51),CONCIN(51,21),INSECT(51),KDAYS(51),KD(51),
           2               DEMAND(51,21),FMT1(40),FMT2(40),FMT3(20)
2897        COMMON/NAMES/X,Z,T,NX,NZ,NC,NS,EXMAX,EXMIN,VXMAX,VXMIN,WMAX,WMIN,
           1               KDMAX,KDMIN,ITER,ITER1,ITER2,DELT1,DELT2,TNEXT,
           2               TPRINT,XPRINT,STABLX,STABLZ,STABLT,R1,R2,VZMAX,SET,
           3               VZMIN,EZMAX,EZMIN,DEMAX,DEMIN,NPPAGE,DELTAX,DELTAZ,
           4               DELTAT,XSTART,XSF,TSTART,COUNT,IPAGE,UPPER,LOWER,
           5               MSU,MSL,TIME,NXM1,NXM2,NZM1,NZM2,ZSF,LTYPE,OPTION
2898        COMMON/OXYGEN/OXOUT(51,21),R(51,21),RDAYS(51,21),OXSAT(51),AK(51),
           1               AKDAYS(51),FMT4(40),FMT5(40),FMT6(20),
           2               SECOND,AKMAX,RMAX,OUTMAX,AKMIN,RMIN,OUTMIN
2899        IF(OPTION.LT.8) GO TO 200
2900        PRINT 181
2901    181 FORMAT(/////,' ****PROGRAM TERMINATED BECAUSE OPTION IS GREATER TH
           $AN 7****',/,1H1)
2902        RETURN1
2903    200 IF(LTYPE.LT.3) GO TO 300
```

```
      PRINT 201
  201 FORMAT(/////,1X,'***** WARNING -- LTYPE IS GREATER THAN OR EQUAL T
     1O 3.',/,4X,'IF LTYPE EQUALS 3, THE USER MUST SUPPLY A SUBROUTINE F
     2OR  THE LOADING CONDITIONS.',/,4X,'IF LTYPE IS GREATER THAN 3, THE
     3 PROGRAM IS TERMINATED *****',/,1H1)
      IF(LTYPE.GT.3) RETURN1
      RETURN
  300 CONTINUE
      IF((SECOND).AND.(OPTION.EQ.3.OR.OPTION.EQ.4.OR.OPTION.EQ.7))
     $   GO TO 350
      GO TO 400
  350 PRINT 360
  360 FORMAT(/////,1X,'**** PROGRAM TERMINATED BECAUSE WHEN SECOND EQUAL
     1S 2 THEN OPTION CANNOT EQUAL 3 OR 4 OR 7 ****')
      RETURN1
  400 CONTINUE
      IF(ITER.NE.0) GO TO 820
C
C-----------------------------------------------------------------------
C          TERMINATE PROGRAM WITH PRINTED MESSAGE IF INPUT DATA
C          VIOLATES STABILITY CRITERIA
C-----------------------------------------------------------------------
      PRINT 815
  815 FORMAT(////,1X,10('*'),'PROGRAM TERMINATED BECAUSE STABILITY CONDI
     $TIONS WERE VIOLATED****', /,1H1)
      RETURN1
  820 CONTINUE
      RETURN
      END
```

Input Data for OXTRANS-I

Each line represents a new card unless single spaced.

1.

.TRUE.

33, 7

0.25, 10.,

0.006944444, 2

0.0, 0

1, 1

17., 4.0, 1, 1, 1

1, 6

33*0.0, 33*0.366667, 33*0.440, 33*0.433333, 33*0.273333,
33*0.233333, 33*0.0

231*450.

497., 510., 524., 537., 550., 562., 575., 562., 550., 537.,
525., 568., 611., 654., 697.,

756., 814., 872., 931., 863., 796., 728., 661., 657., 654.,
650., 647., 633., 619., 605.,

591., 586., 582.

33*0.25

231*0.0

2

7*13.

231*0.0

```
66*0.00333333, 33*0.004, 33*0.00394, 33*0.002485, 66*0.00212

(///, 12X, 100('*'), /, 12X, '*',/, 12X, '*', 2X,
'CONCENTRATIONS AT TIME = ',

F7.2, ' HOURS ( ', F6.4, ' DAYS )', /,12X, '*',/, 12X,
100('*') )

(12X, '*', /, 12X, '*', 42X, 'DEPTH', /, 12X, '*', /, 12X,
'*',3X, 'IMAGE', 4X,

5(12, 'FEET'), 'IMAGE', /, 1X, 111('*'),/, 12X, '*')

(1X, 'MILE', F5.2, 1X, '*', 2X, 7(F6.1, 4X) )

(///, 12X, 100('*'), /, 12X, '*',/, 12X, '*', 2X,
'OXYGEN AT TIME = ',

F7.2, 'HOURS ( ', F6.4, ' DAYS)', /,12X, '*',/, 12X,
100('*') )

(12X, '*', /, 12X, '*', 42X, 'DEPTH', /, 12X, '*', /, 12X,
'*', 3X, 'IMAGE', 4X,

5(I2,' FEET '), 'IMAGE', /, 1X, 111('*'),/, 12X, '*')

(1X, 'MILE', F5.2, 1X, '*', 2X, 7(F6.1, 4X))

33*0., 33*0.1, 165*0.0

33*0.08

33*8.

231*0.
```

COMPUTER PROGRAM FOR

STABLE-I

Object Code = 928 bytes

Array Area = 0 bytes

Total = 928 bytes

```
C
C
C
C                STABLE-I
C
C
C           THIS PROGRAM COMPUTES THE STABILITY CRITERIA FOR TIME
C      INCREMENTS AND DISTANCE INCREMENTS FOR A LARGE NUMBER OF
C      COMBINATIONS OF DISPERSION COEFFICIENTS AND VELOCITIES.  THESE
C      CRITERIA APPLY TO ONE DIMENSIONAL FINITE DIFFERENCE MODELS FOR
C      MASS TRANSPORT IN AN ESTUARY.
C
C         LOGIC AND PROGRAMMING--JONATHAN YOUNG, TEXAS A&M UNIVERSITY
C
C
C-----------------------------------------------------------------------
C
C         *** OUTPUT VARIABLES ***
C
C     E       = DISPERSION COEFFICIENT (MILES**2/DAY)
C     EF      = DISPERSION COEFFICIENT (FEET**2/SECOND)
C     U       = VELOCITY (MILES/DAY)
C     VF      = VELOCITY (FEET/SECOND)
C     DELX    = ALLOWABLE DISTANCE INCREMENT (MILES)
C     DELXF   = ALLOWABLE DISTANCE INCREMENT (FEET)
C     DELT    = ALLOWABLE TIME INCREMENT (DAYS)
C     DELTS   = ALLOWABLE TIME INCREMENT (SECONDS)
C
C-----------------------------------------------------------------------
```

```
      PRINT 10
   10 FORMAT(1H1, T5, 'E', T20, 'EF', T35, 'U', T50, 'UF', T65, 'DELX',
     $  T80, 'DELXF', T95, 'DELT', T110, 'DELTS',//)
      DELX=.25
      E=.333E-05
      DO 200 J=1,7
      UF=0.5E-05
      DO 100 I=1,7
      U=UF*86400./5280.
      IF(U.NE.0)  DELX=2.*E/U
      DELT=(DELX**2.)/(2.*E)
      DELXF=5280.*DELX
      DELTS=86400.*DELT
      EF=(5280.*5280.*E)/86400.
      PRINT 75, E,EF,U,UF,DELX,DELXF,DELT,DELTS
   75 FORMAT(1X, 8E15.5)
      UF=UF*2.
  100 CONTINUE
      E=2.*E
  200 CONTINUE
      STOP
      END
```

COMPUTER PROGRAM FOR

STABLE-II

Object Code = 4016 bytes

Array Area = 0 bytes

Total = 4016 bytes

```
C
C
C
C      *---------------------------------------------------*
C      *                                                   *
C      *                    STABLE-II                      *
C      *                                                   *
C      * STABILITY CRITERIA FOR ONE- AND TWO-DIMENSIONAL   *
C      *       EXPLICIT FINITE DIFFERENCE SCHEMES          *
C      *                                                   *
C      *---------------------------------------------------*
C
C
C          THIS COMPUTER PROGRAM COMPUTES THE MAXIMUM ALLOWABLE DISTANCE
C     AND TIME INCREMENTS FOR A SET OF ESTUARY DATA WHICH IS TO BE USED
C     IN A ONE DIMENSIONAL OR TWO DIMENSIONAL EXPLICIT FINITE DIFFERENCE
C     MODEL FOR MASS TRANSPORT.
C
C        LOGIC AND PROGRAMMING--JONATHAN YOUNG, TEXAS A&M UNIVERSITY
C
C-----------------------------------------------------------------------
C        *** INPUT VARIABLES ***
C
C     NSETS  = NUMBER OF DATA SETS TO BE PROCESSED
C     DELMX  = DISTANCE INCREMENT IN THE X DIRECTION THAT THE USER
C              WISHES TO USE IN HIS MODEL (MILES)
C     DELTAZ = DISTANCE INCREMENT IN THE Z DIRECTION THAT THE USER
C              WISHES TO USE IN HIS MODEL (FEET)
C     NZ     = NUMBER OF POINTS IN THE Z DIRECTION  ( IF NZ = 1, THE
C              MODEL IS ONE DIMENSIONAL.  IF NZ IS GREATER THAN 1, THE
C              MODEL IS CONSIDERED TO BE TWO DIMENSIONAL. )
```

```
C      EXMIN  = MINIMUM DISPERSION COEFFICIENT IN THE X DIRECTION
C                  ( FEET**2/SECOND )
C      EXMAX  = MAXIMUM DISPERSION COEFFICIENT IN THE X DIRECTION
C                  ( FEET**2/SECOND )
C      EZMIN  = MINIMUM DISPERSION COEFFICIENT IN THE Z DIRECTION
C                  ( FEET**2/SECOND )
C      EZMAX  = MAXIMUM DISPERSION COEFFICIENT IN THE Z DIRECTION
C                  ( FEET**2/SECOND )
C      VXMAX  = MAXIMUM VELOCITY IN THE X DIRECTION (FEET/SECOND)
C      VZMAX  = MAXIMUM VELOCITY IN THE Z DIRECTION (FEET/SECOND)
C      KDAYS  = MAXIMUM DECAY RATE ( /DAY)
C
C-----------------------------------------------------------------------
C
C
C
```

```
      REAL KDMAX
      REAL KDAYS
      READ , NSETS
      PRINT, NSETS
      DO 1000 II=1,NSETS
      READ ,DELMX,DELTAZ,NZ,EXMIN,EXMAX,EZMIN,EZMAX,VXMAX,VZMAX,KDAYS
      PRINT,DELMX,DELTAZ,NZ,EXMIN,EXMAX,EZMIN,EZMAX,VXMAX,VZMAX,KDAYS
      DELTAX=DELMX*5280.
      XKD1=KDAYS
      KDMAX=KDAYS/86400.
      UNITS1 = 86400./5280.
      UNITS2 = 86400./(5280.*5280.)
      DELMZ=DELTAZ/5280.
      VX1=VXMAX*UNITS1
      VZ1=VZMAX*UNITS1
      EX1=EXMAX*UNITS2
      EX2=EXMIN*UNITS2
      EZ1=EZMAX*UNITS2
      EZ2=EZMIN*UNITS2
      STABLX = DELTAX
      STABLZ = DELTAZ
      IF(NZ.NE.1) GO TO 500
      IF(EXMIN.GT.0.0.AND.VXMAX.GT.0.0) STABLX=2.*EXMIN/VXMAX
      TERM=2.*EXMAX+DELTAX**2.*KDMAX
      IF(TERM.NE.0.0) STABLT=(DELTAX**2.)/TERM
      GO TO 800
  500 CONTINUE
      IF(EXMIN.GT.0.0.AND.VXMAX.GT.0.0) STABLX=2.*EXMIN/VXMAX
      IF(EZMIN.GT.0.0.AND.VZMAX.GT.0.0) STABLZ=2.*EZMIN/VZMAX
```

```
      TERM=2.*(EXMAX*DELTAZ**2.+EZMAX*DELTAX**2.)+KDMAX*(DELTAX**2)*
     $   (DELTAZ**2)
      IF(TERM.NE.0.0) STABLT=(DELTAZ**2.)*(DELTAX**2.)/TERM
  800 CONTINUE
      SXM=STABLX/5280.
      SZM=STABLZ/5280.
      STD=STABLT/86400.
      PRINT 100
      PRINT 540, DELTAX, DELMX
      PRINT 545, DELTAZ,DELMZ
      PRINT 560, NZ
      PRINT 565, VXMAX, VX1
      PRINT 572,VZMAX,VZ1
      PRINT 575,EXMAX,EX1
      PRINT 580,EXMIN,EX2
      PRINT 585, EZMAX,EZ1
      PRINT 590, EZMIN,EZ2
      PRINT 610,XKD1,KDMAX
      PRINT 505
      PRINT 502
      PRINT 640
      PRINT 645,STABLX,SXM
      PRINT 650,STABLZ,SZM
      PRINT 655,STABLT,STD
      PRINT 505
  100 FORMAT(1H1)
  502 FORMAT(/)
  505 FORMAT(T45, 38('*'))
  540 FORMAT(    /, T53, 'X INCREMENT =', F7.0, ' FEET (', F8.5, ' MILES
     $)',/)
```

```
      545 FORMAT(T53, 'Z INCREMENT =', F7.0, ' FEET (', E10.3, ' MILES )'/)
      560 FORMAT(T39, 'NUMBER OF VERTICAL POINTS =',I4,/)
      565 FORMAT(T37, 'MAXIMUM HORIZONTAL VELOCITY = ', F5.2, ' FEET/SECOND
         $(',F6.2, ' MILES/DAY )',/)
      572 FORMAT(T39, 'MAXIMUM VERTICAL VELOCITY =', E10.3, ' FEET/SECOND ('
         $ , E10.3, ' MILES/DAY )',/)
      575 FORMAT(T35, 'MAXIMUM HORIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
         $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
      580 FORMAT(T35, 'MINIMUM HORIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
         $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
      585 FORMAT(T37, 'MAXIMUM VERTICAL DISPERSION = ', F6.3, ' FEET SQUARED
         $ / SECOND ( ', E10.3, ' MILES SQUARED / DAY )',/)
      590 FORMAT(T37, 'MINIMUM VERTICAL DISPERSION = ', F6.0, ' FEET SQUARED
         $ / SECOND ( ', E10.3, ' MILES SQUARED / DAY )',/)
      610 FORMAT(T46, 'MAXIMUM DECAY RATE = ', F5.3, ' PER DAY ( ', E10.3,
         $ ' PER SECOND )', /)
      640 FORMAT(T56, 'STABILITY CRITERIA',/)
      645 FORMAT(T35, 'MAXIMUM ALLOWABLE X INCREMENT = ', F7.0, ' FEET (',
         $ F8.5, 'MILES )',/)
      650 FORMAT(T35, 'MAXIMUM ALLOWABLE Z INCREMENT = ', F7.0, ' FEET (',
         $ F8.5, 'MILES )',/)
      655 FORMAT(T32, 'MAXIMUM ALLOWABLE TIME INCREMENT = ', F7.0, ' SECONDS
         $ (', E10.3, ' DAYS )',/)
     1000 CONTINUE
          STOP
          END
```

Input Data for STABLE-II

Each line represents a new card unless single spaced.

1

0.2, 5., 11, 500., 500., 0.001, 0.005, 0.441, 0., 0.384

COMPUTER PROGRAM FOR

EXACT-I

Object Code = 5664 bytes

Array Area = 400 bytes

Total = 6064 bytes

```
C
C
C     ***************************************************************
C     *******************  EXACT-I  *********************************
C     ***********  ANALYTICAL SOLUTIONS FOR  ************************
C     *****  ONE DIMENSIONAL CONVECTIVE DISPERSION  *****************
C     ***************************************************************
C
C
C
C        THIS COMPUTER PROGRAM COMPUTES THE PROFILES FOR VARIOUS
C     CLOSED-FORM SOLUTIONS TO THE ESTUARY MASS TRANSPORT EQUATION.
C     THESE SOLUTIONS ARE OBTAINED BY ASSUMING CONSTANT COEFFICIENTS
C     AND ESTUARIES OF INFINITE LENGTH.  SOLUTIONS ARE CALCULATED FOR
C     CONSTANT CONCENTRATIONS, INSTANTANEOUS RELEASES, AND CONTINUOUS
C     DISCHARGES.  THIS PROGRAM CAN BE USED TO CHECK THE ACCURACY OF
C     FINITE DIFFERENCE FORMULATIONS.
C
C        LOGIC AND PROGRAMMING--JONATHAN YOUNG, TEXAS A&M UNIVERSITY
C
```

```
C-----------------------------------------------------------------------
C
C          *** INPUT VARIABLES ***
C
C      DELTAX = DISTANCE INCREMENT (MILES OR FEET)
C      DELTAT = TIME INCREMENT (DAYS OR SECONDS)
C      U      = VELOCITY (FEET/SECOND)
C      EX     = DISPERSION COEFFICIENT (MILES**2/DAY OR FEET**2/SECOND)
C      CZERO  = CONSTANT CONCENTRATION AT AN OUTFALL, OR MASS PER UNIT
C               CROSS SECTIONAL AREA, OR MASS RATE PER UNIT AREA
C      ITEND  = NUMBER OF ITERATIONS
C      IXEND  = NUMBER OF X INCREMENTS
C      XBEGIN = INITIAL X VALUE (FEET)
C      LTYPE  = TYPE OF LOADING
C                  1 MEANS CONSTANT CONCENTRATION AT THE OUTFALL
C                  2 MEANS AN INSTANTANEOUS RELEASE
C                  3 MEANS A CONTINUOUS DISCHARGE
C      XKD    = DECAY RATE ( /DAY) -- USED FOR LTYPE = 2 ONLY
C
C-----------------------------------------------------------------------
C
C
```

```
      DIMENSION C(100)
      DATA C/100*0.0/, PI/ 3.141593/
      READ , DELTAX, DELTAT, U, EX, CZERO, ITEND, IXEND, XBEGIN, LTYPE,
     $ XKD
      PRINT, DELTAX, DELTAT, U, EX, CZERC, ITEND, IXEND, XBEGIN, LTYPE,
     $ XKD
      DAYS=DELTAT
      SECS=DELTAT
      IF(DELTAT.LE.1.) SECS=DELTAT*86400.
      IF(DELTAT.GT.1.) DAYS=DELTAT/86400.
      DELF=DELTAX
      DELM=DELTAX
      IF(DELTAX.GT.2.0) DELM=DELTAX/5280.
      IF(DELTAX.LE.2.0) DELF=DELTAX*5280.
      UM=U*86400./5280.
      D=EX
      DM=EX
      IF(EX.LE.25.0) D=EX*5280.*5280./86400.
      IF(EX.GT.25.0) DM=EX*86400./(5280.*5280.)
      XKDS=XKD/86400.
      PRINT 5
    5 FORMAT(1H1,1X,110('*'),//)
      PRINT 7
    7 FORMAT(40X, 42('*'))
      PRINT 25
   25 FORMAT(40X, '******* ONE DIMENSIONAL ANALYSIS ********')
```

```
      GO TO (10,20,30,40,50), LTYPE
   10 PRINT 11, CZERO
   11 FORMAT(40X, '********  CONSTANT CONCENTRATION  ********', //,10X,
     $   ' THE FOLLOWING PARAMETERS ARE APPLIED TO A CONSTANT CONCENTRAT
     $ION OF ', F7.0, ' PPM MAINTAINED AT X = 0.0', /)
      GO TO 60
   20 PRINT 11, CZERO
      GO TO 60
   30 PRINT 31, CZERO
   31 FORMAT(40X, '********  INSTANTANEOUS RELEASE  ********', //, 10X,
     $ ' THE FOLLOWING PARAMETERS ARE APPLIED TO AN INSTANTANEOUS RELEAS
     $E OF ', F10.0, ' UNITS OF MASS PER UNIT AREA',/, 15X, ' AT X=0.0 A
     $ND TIME = 0.0',/)
      GO TO 60
   40 PRINT 31, CZERO
      GO TO 60
   50 PRINT 51, CZERO
   51 FORMAT(40X, '********  CONTINUOUS DISCHARGE  ********', //,  10X,
     $   ' THE FOLLOWING PARAMETERS ARE APPLIED TO A CONTINUOUS DISCHARG
     $E OF ', F9.0, ' UNITS OF MASS PER UNIT AREA PER UNIT TIME AT X = C
     $.0 ',/)
      GO TO 60
   60 CONTINUE
      PRINT 535, LTYPE
  535 FORMAT(//, T59, 'LTYPE =', I3,/)
      PRINT 540, DELF, DELM
  540 FORMAT(      T53, 'X INCREMENT =', F7.0, ' FEET (', F8.5, ' MILES
     $)',/)
      PRINT 550, SECS,DAYS
  550 FORMAT(T50, 'TIME INCREMENT =', F7.0, ' SECONDS (', E10.3, ' DAYS
     $)'/)
```

```
      PRINT 565, U,UM
  565 FORMAT(T37, '        HCRIZONTAL VELOCITY = ', F5.2, ' FEET/SECCND
     $(',F6.2, ' MILES/DAY )',/)
      PRINT 575, D, DM
  575 FORMAT(T35, '        HCRIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
     $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
      PRINT 580, XKD, XKDS
  580 FORMAT( T54, 'DECAY RATE = ', F6.3, ' PER DAY ( ', E10.3,
     $    ' PER SECOND )',/)
      PRINT 65
   65 FORMAT(//,1X,110('*'),//)
      DO 1000 IT=1,ITEND
      T=SECS*IT
      TIME=T/86400.
      GO TO (100, 300, 500) , LTYPE
  100 CONTINUE
      IF(U.NE.0.0) GO TO 200
      DO 120 IX=1,IXEND
      X=(IX-1)*DELF+XBEGIN
      B3=X/(2.*SQRT(D*T))
      C(IX)=CZERO*ERFC(B3)
  120 CONTINUE
      GO TO 800
  200 DO 220 IX=1,IXEND
      X=(IX-1)*DELF+XBEGIN
      A=U*X/D
      B1=(X+U*T)/(2.*SQRT(D*T))
      B2=(X-U*T)/(2.*SQRT(D*T))
      C(IX)=(CZERO/2.)*(EXP(A)*ERFC(B1)+ERFC(B2))
```

```
  220 CONTINUE
      GO TO 800
  300 XM=CZERO
      DECAY = -XKD*TIME
      DO 350 IX=1,IXEND
      X=(IX-1)*DELF+XBEGIN
      C1=XM/SQRT(4.*PI*D*T)
      C2=-(X-U*T)**2/(4.*D*T)
      C(IX)=C1*EXP(C2)
      C(IX)=C(IX)*EXP(DECAY)
  350 CONTINUE
      GO TO 800
  500 XM=CZERO
      DO 520 IX=1,IXEND
      X=(IX-1)*DELF+XBEGIN
      E1=XM*T**0.5/SQRT(PI*D)
      E2=-X**2/(4.*D*T)
      E3=XM*X/(2.*D)
      E4=X/(2.*SQRT(D*T))
      C(IX)=E1*EXP(E2)-E3*ERFC(E4)
  520 CONTINUE
      GO TO 800
  800 CONTINUE
      PRINT 825, T,TIME
  825 FORMAT(///, ' TIME = ', F7.0, ' SECONDS ( ', F8.5, ' DAYS )',/)
```

```
      MSTOP=IXEND/2
      DO 875 M=1,MSTOP
      P1=DELF*(M-1)
      P2=DELM*(M-1)
      P3=DELF*(M+MSTOP-1)
      P4=DELM*(M+MSTOP-1)
      M1=M+MSTOP
      PRINT 850, P1,P2,C(M),P3,P4,C(M1)
  850 FORMAT(2(' C AT ', F7.0, ' FEET (', F8.4, ' MILES ) = ', F10.2,
     $   10X))
  875 CONTINUE
 1000 CONTINUE
      STOP
      END
```

Input Data for EXACT-I

Each line represents a new card unless single spaced.

0.25, 0.1, 5., 10., 100., 2, 20, -5280., 1, 0.

COMPUTER PROGRAM FOR

EXACT-II

Object Code = 4,656 bytes

Array Area = 40,160 bytes

Total = 44,816 bytes

```
C
C
C
C         ********************************************************
C         ***********************  EXACT-II  **********************
C         *****  TWO DIMENSIONAL CONVECTIVE DISPERSION  *********
C         ********************************************************
C
C
C            THIS COMPUTER PROGRAM COMPUTES THE CONCENTRATION PROFILE WITH
C        TIME IN TWO DIMENSIONS FOR AN INSTANTANEOUS RELEASE OF A MASS M
C        PER UNIT DEPTH ( OR WIDTH ).  THE CALCULATIONS ARE BASED UPON A
C        KNOWN, CLOSED-FORM SOLUTION TO THE ESTUARY MASS TRANSPORT EQUATION
C        ASSUMING CONSTANT COEFFICIENTS AND INFINITE BOUNDARIES.  THIS
C        PROGRAM CAN BE USED TO CHECK THE ACCURACY OF FINITE DIFFERENCE
C        FORMULATIONS.
C
C           LOGIC AND PROGRAMMING--JONATHAN YOUNG, TEXAS A&M UNIVERSITY
C
```

```
C-----------------------------------------------------------------------
C
C         *** INPUT VARIABLES ***
C
C      NX     = NUMBER OF POINTS IN THE X DIRECTION
C      NZ     = NUMBER OF POINTS IN THE Z DIRECTION
C      DELTAX = DISTANCE INCREMENT IN THE X DIRECTION (MILES)
C      DELTAZ = DISTANCE INCREMENT IN THE Z DIRECTION (MILES)
C      DELTAT = TIME INCREMENT (DAYS)
C      XBEGIN = INITIAL X VALUE (MILES)
C      ZBEGIN = INITIAL Z VALUE (MILES)
C      ITER   = NUMBER OF ITERATIONS
C      EX     = DISPERSION COEFFICIENT IN THE X DIRECTION
C                 (MILES**2/DAY)
C      EZ     = DISPERSION COEFFICIENT IN THE Z DIRECTION
C                 (MILES**2/DAY)
C      VXF    = VELOCITY IN X DIRECTION (FEET/SECOND)
C      VZF    = VELOCITY IN Z DIRECTION (FEET/SECOND)
C      XM     = MASS PER UNIT DEPTH OR WIDTH
C      XKD    = DECAY RATE ( /DAY)
C      FMT1   = FORMAT FOR HEADING
C
C-----------------------------------------------------------------------
C
```

```
C
      DIMENSION C(100,100)
      DIMENSION FMT1(40)
      INTEGER COUNT
      DATA C/10000*0.0/
      READ , NX,NZ,DELTAX, DELFZ,DELTAT,XBEGIN,ZBEGIN,ITER
      PRINT, NX,NZ,DELTAX, DELFZ,DELTAT,XBEGIN,ZBEGIN,ITER
      READ , EXF,EZF,VXF,VZF,XM,XKD
      PRINT, EXF,EZF,VXF,VZF,XM,XKD
      READ 205, (FMT1(I), I=1,40)
      PRINT 202
  202 FORMAT(//,1X, 'FORMAT FOR OUTPUT')
      PRINT 205, (FMT1(I),I=1,40)
  205 FORMAT (20A4,/,20A4)
      DELTAT=1./24.
      COUNT=2
      PI=3.14159
      SECS=DELTAT*86400.
      DELFX=DELTAX*5280.
      DELTAZ=DELFZ/5280.
      VX=VXF*86400./5280.
      VZ=VZF*86400./5280.
      EX=EXF*86400./(5280.*5280.)
      EZ= EZF*86400./(5280.*5280.)
      XKDS = XKD/86400.
      PRINT 5
    5 FORMAT(1H1,1X,110('*'),//)
      PRINT 7
    7 FORMAT(40X, 42('*'))
```

```
1921         PRINT 25
1922      25 FORMAT(40X, '******* TWC DIMENSICNAL ANALYSIS ********')
1923         PRINT 31, XM
1924      31 FORMAT(40X, '*******   INSTANTANEOUS RELEASE  ********', /,
            $  40X, 42('*'),//,
            $ ' THE FOLLOWING PARAMETERS ARE APPLIED TO AN INSTANTANEOUS RELEAS
            $E OF ',F10.0, ' PPM AT X = 0.0 AND TIME = 0.0', /)
1925         PRINT 540, DELFX, DELTAX
1926     540 FORMAT(      T53, 'X INCREMENT =', F7.0, ' FEET (', F8.5, ' MILES
            $)',/)
1927         PRINT 545, DELFZ, DELTAZ
1928     545 FORMAT(      T53, 'Z INCREMENT =', F7.0, ' FEET (', F8.5, ' MILES
            $)',/)
1929         PRINT 550, SECS, DELTAT
1930     550 FORMAT(T50, 'TIME INCREMENT =', F7.0, ' SECONDS (', E10.3, ' DAYS
            $)'/)
1931         PRINT 565, VXF, VX
1932     565 FORMAT(T37, '        HCRIZONTAL VELOCITY = ', F5.2, ' FEET/SECOND
            $(',F6.2, ' MILES/DAY )',/)
1933         PRINT 570, VZF, VZ
1934     570 FORMAT(T37, '          VERTICAL VELCCITY = ', F5.2, ' FEET/SECOND
            $(',F6.2, ' MILES/DAY )',/)
1935         PRINT 575, EXF, EX
1936     575 FORMAT(T35, '        HCRIZONTAL DISPERSION = ', F6.0, ' FEET SQUAR
            $ED / SECOND ( ', F5.2, ' MILES SQUARED / DAY )',/)
1937         PRINT 580, EZF,EZ
1938     580 FORMAT(T35, '          VERTICAL CISPERSION = ', F9.3, ' FEET SQUAR
            $ED / SECOND ( ', F8.5, ' MILES SQUARED / DAY )',/)
1939         PRINT  582, XKD,XKDS
1940     582 FORMAT(T54, 'DECAY RATE =', F6.3, ' PER DAY ( ' E10.3, ' PER SECCN
            $D )',/)
```

```
      XBF=XBEGIN*5280.
      PRINT 585, XBF, XBEGIN
  585 FORMAT(T49,'INITIAL X VALUE =',    F7.0, ' FEET (', F8.5, ' MILES
     $)',/)
      ZBF=ZBEGIN*5280.
      PRINT 590, ZBF, ZBEGIN
  590 FORMAT(T49,'INITIAL Z VALUE =',    F7.0, ' FEET (', F8.5, ' MILES
     $)',/)
      PRINT 595
  595 FORMAT( 1X, 110('*') )
      DO 300 IT=1,ITER
      T=IT*DELTAT
      DO 200 I=1,NX
      X=(I-1)*DELTAX+XBEGIN
      DO 100 J=1,NZ
      Z=(J-1)*DELTAZ+ZBEGIN
      TERM1=-((X-VX*T)**2 )/(4.*T*EX)
      IF(TERM1.LT.-50.) TERM1=-50.
      TERM2=SQRT(2.*T*EX)
      TERM3=-((Z-VZ*T)**2 )/(4.*T*EZ)
      IF(TERM3.LT.-50.) TERM3=-50.
      TERM4=SQRT(2.*T*EZ)
      C(I,J)=(XM/(2.*PI))*EXP(TERM1)*EXP(TERM3)/(TERM2*TERM4)
      XKDT=-XKD*T
      C(I,J)=C(I,J)*EXP(XKDT)
  100 CONTINUE
  200 CONTINUE
      PRINT 275
  275 FORMAT(1H1)
```

```
      TIME=T*86400.
      HOURS=T*24.
      WRITE(6,FMT1) HOURS,T
      PRINT225, ((C(I,J),J=1,NZ),I=1,NX)
  225 FORMAT(1(1X,11(F6.0)))
  300 CONTINUE
      STOP
      END
```

Input Data for EXACT-II

Each line represents a new card unless single spaced.

```
28, 11, 0.2, 5., 0.04083333, -2.6, 0.0, 1

500., 0.010, 0., 0., 5.0, 0.

(///, 12X, 100('*'), /, 12X, '*',/, 12X, '*', 2X,
'CONCENTRATIONS AT TIME = ',

F7.2, ' HOURS (', F6.4, ' DAYS )', /, 12X, '*',/, 12X,
100('*') )
```

COMPUTER PROGRAM FOR

PROFILE-I

Object Code = 1864 bytes

Array Area = 0 bytes

Total = 1864 bytes

```
C                     PROFILE - I
C
C
C             THIS PROGRAM COMPUTES THE STEADY-STATE CONCENTRATION PROFILE
C        FOR A NONCONSERVATIVE SUBSTANCE , WHERE THE CONCENTRATION IS KNOWN
C        AT THE SOURCE POINT
C
C
C           *** INPUT VARIABLES ***
C
C        E     = DISPERSION COEFFICIENT (MILES**2/DAY)
C        U     = VELOCITY (MILES/DAY)
C        XKD   = DECAY COEFFICIENT ( /DAY)
C        CZERO = CONCENTRATION AT OUTFALL
C-----------------------------------------------------------------------
C
         READ(5,10)E,U,XKD,CZERO
      10 FORMAT(4F10.0)
         WRITE(6,15)E
      15 FORMAT(4X,24HDIFFUSION COEFFICIENT = ,F5.1,21H SQUARE MILES PER DA
        2Y,/)
         WRITE(6,20)U
      20 FORMAT(4X,11HVELOCITY = ,F5.1,14H MILES PER DAY,/)
         WRITE(6,23) CZERO
      23 FORMAT(4X,'CONCENTRATION AT OUTFALL = ', F6.1, ' PPM',/)
         WRITE(6,25)XKD
      25 FORMAT(4X,20HDECAY COEFFICIENT = ,F5.2,8H PER DAY,////)
         WRITE(6,30)
      30 FORMAT(5X,1HX,13X,1HC,//)
```

```
      ADOWN=U/(2.*E)*(1.0-SQRT(1.0+4.*XKD*E/(U**2.)))
      AUP=  U/(2.*E)*(1.0+SQRT(1.0+4.*XKD*E/(U**2.)))
      X=-11.0
      DO 100 I=1,9
      X=X+1.0
      C=CZERO*EXP(AUP*X)
      WRITE(6,50)X,C
   50 FORMAT(F10.2,4X,F9.4,/)
  100 CONTINUE
      DO 150 J=1,8
      X=X+0.25
      C=CZERO*EXP(AUP*X)
      WRITE(6,50)X,C
  150 CONTINUE
      DO 200 K=1,20
      X=X+0.25
      C=CZERO*EXP(ADOWN*X)
      WRITE(6,50)X,C
  200 CONTINUE
      DO 250 L=1,45
      X=X+1.0
      C=CZERO*EXP(ADOWN*X)
      WRITE(6,50)X,C
  250 CONTINUE
      STOP
      END
```

Input Data for PROFILE-I

Each line represents a new card unless single spaced.

10.0	1.0	0.25	10.0

COMPUTER PROGRAM FOR

PROFILE-II

Object Code	=	2,864 bytes
Array Area	=	12,060 bytes
Total	=	14,924 bytes

```
C
C                  PROFILE-II
C
C
C          THIS COMPUTER PROGRAM CALCULATES THE STEADY-STATE PROFILES
C     FOR BIOCHEMICAL OXYGEN DEMAND (BOD) AND DISSOLVED OXYGEN (DO)
C     FOR A CONTINUOUS RELEASE OF A SUBSTANCE INTO AN IDEALIZED, ONE-
C     DIMENSIONAL ESTUARY.
C
C        LOGIC AND PROGRAMMING--JONATHAN YOUNG, TEXAS A&M UNIVERSITY
C
C
C-----------------------------------------------------------------------
C
C              *** INPUT VARIABLES ***
C
C     W       = LOADING RATE (POUNDS OF BOD PER DAY)
C     Q       = FLOWRATE (CFS)
C     U       = VELOCITY (FEET/SECOND)
C     E       = DISPERSION COEFFICIENT (FEET**2/SECOND)
C     K       = DECAY RATE ( /DAY, BASE E)
C     K2      = REAERATICN COEFFICIENT ( /DAY)
C     CSAT    = OXYGEN SATURATION VALUE (PPM)
C     XUP     = DISTANCE ABOVE OUTFALL (MILES)
C     XDOWN   = DISTANCE BELOW OUTFALL (MILES)
C     DELMX   = DISTANCE INCREMENT (MILES)
C     DI      = INITIAL DEFICIT AT OUTFALL (PPM)
C
C-----------------------------------------------------------------------
C
```

```
      REAL L,K,K2,M1,M2,LZ
      DIMENSION L(1005),D(1005), C(1005)
      READ , W,Q,U,E,K,K2,CSAT,XUP,XDOWN,DELMX , DI
      PRINT, W,Q,U,E,K,K2,CSAT,XUP,XDOWN,DELMX , DI
      Q=Q*62.4
      W=W/86400. * 1.E6
      XXK=K
      XXK2=K2
      K=K/86400.
      K2=K2/86400.
      DELTAX=DELMX*5280.
      IF(XXK.EQ.XXK2)  XUP=0.0
      NDIM=(-XUP+XDOWN)/DELMX + 2
      TERM1=1. + (4.*K*E)/(U*U)
      M1=SQRT(TERM1)
      TERM2=1.+(4.*K2*E)/(U*U)
      M2=SQRT(TERM2)
      LZ=W/(Q*M1)
      U2E=U/(2.*E)
      XJ=U2E*(1.-M1)
      X=(XUP-DELMX)*5280.
      PRINT 50
   50 FORMAT(1H1,//,1X,'BOD AND OXYGEN PROFILES',//,
     $    T5, 'X', T15, 'L', T25, 'D', T32, 'OX',/)
      SIGN=1.
      TEST=ABS(XXK-XXK2)
      IF(TEST-.001) 250,25,25
   25 CONTINUE
      DZ=(K*W)/((K2-K)*Q)
      DO 200 I=1,NDIM
```

```
      X=X+DELTAX
      IF(X.GE.0.0)  SIGN=-1.
      EXPON1=U2E *X*(1.+SIGN*M1)
      EXPON2=U2E *X*(1.+SIGN*M2)
      L(I)=LZ*EXP(EXPON1)
      D(I)=DZ*((1./M1)*EXP(EXPON1)-(1./M2)*EXP(EXPON2))
      D(I)=D(I) + DI*EXP(EXPON2)
      C(I)=CSAT-D(I)
      XM=X/5280.
      PRINT 100, XM,L(I),D(I),C(I)
  100 FORMAT(1X,F7.2,3X,F8.3,3X,F5.2,3X,F5.2)
  200 CONTINUE
      GO TO 400
  250 CONTINUE
      X=-DELTAX
      DO 300 I=1,NDIM
      X=X+DELTAX
      IF(X.GE.0.0) SIGN=-1.
      EXPON=U2E*X*(1.+SIGN*M1)
      L(I)=LZ*EXP(EXPON )
      DZ=K*LZ*X/(U-2.*E*XJ)
      XJX=XJ*X
      D(I)=DZ*EXP(EXPON )+DI*EXP(XJX)
      C(I)=CSAT-D(I)
      XM=X/5280.
      PRINT 100, XM,L(I),D(I),C(I)
  300 CONTINUE
  400 CONTINUE
      STOP
      END
```

Input Data for PROFILE-II

Each line represents a new card unless single spaced.

50000., 1000., .2, 400., .23, .10, 8., -10., 20., 0.5, 0.0