Report Number 81-4
August 1981

A Two-Dimensional Collocation Finite-Element Model for Transient Mixing in Natural Rivers

Yahia S. Halabi & Hung Tao Shen

Department of Civil and Environmental Engineering
Clarkson College, Potsdam, New York 13676

A Two-Dimensional Collocation Finite-Element Model for Transient Mixing in Natural Rivers

by

Yahia S. Halabi

and

Hung Tao Shen

Department of Civil and Environmental Engineering
Clarkson College of Technology
Potsdam, New York 13676

Report No. 81-4

Prepared with the support of
The New York Sea Grant Institute
Office of Sea Grant
National Oceanic and Atmospheric Administration
U.S. Department of Commerce

August, 1981

ABSTRACT

In this study, a numerical model is developed for two-dimensional, transient mixing for steady uniform flow in natural river channels. Through the use of an orthogonal curvilinear coordinate system, the river channel is mapped into a rectangular strip by introducing the cumulative discharge as the new transverse coordinate. Concentration distribution in the channel is determined by a collocation finite element scheme. A computer program is developed and verified with analytical solutions and a steady state field measurement. The result is also compared with a finite difference method which uses a combined implicit/explicit scheme. The collocation finite element method is more efficient and stable. The "overshoot" near the peak of the dispersing front presented in the finite difference solution did not appear in the present solution.

ACKNOWLEDGMENTS

This report represents a part of a study aimed at the development of two-dimensional mathematical models for simulating transport and morphological processes in natural rivers. The work presented in this report was supported by the New York Sea Grant Institute, Office of Sea Grant, National Oceanic and Atmospheric Administration (NOAA), U.S. Department of Commerce, under Contract No. 334-S129-G.

The authors would like to thank Dr. William Briggs and Dr. Theodore Papatheodorou, both of the Department of Mathematics and Computer Science, Clarkson College of Technology, for their helpful discussions.

The computer work was performed on an IBM 4341 Computer at the Clarkson College's Educational Resources Center. Partial support provided by the Graduate School of Clarkson College is also acknowledged.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Mathematical modeling for mixing processes in rivers has developed rapidly during the last decade. Field surveys and laboratory models are not usually feasible for studying the transport of dispersants in a water body due to logistical and economic considerations. Because of this, mathematical models have long been considered as an efficient and practical alternate for predicting the distribution of dissolved or suspended materials, such as water temperature, chemical and biological pollutants, and suspended sediments (43,19). Numerous numerical models have been developed based on finite difference or finite element methods. Loziuk, et. al. (23) developed a two-dimensional model for predicting water temperature distributions in rivers and cooling lakes, using the Galerkin finite element method. Banks (1) developed a finite difference model for predicting BOD and DO distributions in rivers and lakes using a mixing-cell concept. Leimkuhler (22) developed a two-dimensional vertically integrated Galerkin finite-element model to predict sediment dispersion in Massachusetts Bay taking into consideration the variation of depth in the flow field. Any two-dimensional dispersion model requires a hydrodynamic model to provide a correct description of the velocity field. For analysis in tidal estuaries, most of the hydrodynamic models are developed based on the two-dimensional vertically integrated shallow water wave equations in Cartesian coordinates (22,44). For the case of steady state flows in natural rivers, two-dimensional potential flow solutions are often used (1,23). Potential flow solutions are not able to provide an adequate description of velocity field and the effect of depth variations.

Fischer (11) introduced a stream-tube model in a study of transverse mixing to account for transverse variations in depth and flow. Chang (6) used an orthogonal curvilinear coordinate system as a means of accounting for meandering effects in a natural river. Yotsukura and Cobb (46) formalized the stream-tube formulation by using the cumulative discharge to replace the transverse distance coordinate and obtained analytical solutions for transverse diffusion in straight uniform streams. This approach was extended by Sayre and Yeh (33) to meandering channels. Based on a rigorous analysis, Yotsukura and Sayre (45) have shown that by employing the concepts of cumulative discharge and the orthogonal curvilinear coordinate system, a simple form of convection-diffusion equation can be obtained for steady state two-dimensional mixing in meandering rivers.

Based on the analysis of Yotsukura and Sayre (45), Shen (36) extended the steady state mixing equation to the case of transient mixing in steady-state river flows. This type of formulation eliminated the presence of the transverse velocity term in the convection diffusion equation and mapped the irregular physical domain into a rectangular strip in the new coordinate system. This approach is much more convenient for mathematical treatment than the two-dimension convection-diffusion equation in Cartesian coordinates. Moreover, it also has the advantage of avoiding the cumbersome, if not impractical, numerical solution of velocity field by using available simple simulation formulas for transverse flow distribution.

The numerical solution of convection diffusion equations has been a subject of interest to engineers and mathematicians for their applications in mixing processes in porous media, surface water bodies, and the atmospheric environment. Early finite-difference solutions were obtained by Peaceman and Rachford (26,27), Roberts and Weiss (31), Stone and Brain (41), and many others. Price, et. al. (28) summed up the early experience and discussed the difficulties of oscillations and the undue numerical diffusion in these solutions. Spalding, et. al. (40) proposed the "upwind" difference scheme which incorporated the idea of weighting technique. Shaimir and Harleman (34) developed a combined implicit-explicit finite difference scheme to solve two-dimensional groundwater dispersion problems. In this scheme, the longitudinal convection and diffusion are treated by the Stone-Brain method, whereas transverse dispersion is treated by the alternating direction procedure. In an attempt to solve the two-dimensional convection-diffusion equation in the natural (stream-tube) coordinates, Harden and Shen (15) applied the Shamir-Harleman scheme to transient mixing in natural rivers and verified numerically that the longitudinal diffusion term is negligible. An "overshoot" near the peak of the dispersing front exists in their solution. The size of the time increment and the space grid size required in order to satisfy the convergence criterion is relatively small. This limitation makes the scheme relatively inefficient in simulating field problems. Recently, the Omaha District, U.S. Army Corps of Engineers and the Sutron Corporation (42) developed a finite-difference scheme for simulating two-dimensional mixing in rivers. This model retained with transverse distance as an independent variable in the convection-diffusion equation rather than the cumulative discharge. The numerical method used in this model is an ADI method (30) similar to the method used by Harden and Shen (15) and has similar numerical problems. To avoid the limitation of small time increment, the Sutron report suggested the use of unrealistically large values of the longitudinal diffusion coefficient.

In the last ten years, finite element methods have been applied to dispersion problems. Finite element methods are considered to be more flexible for problems with irregular boundaries and usually allows larger element size in discretizing the solution domain, resulting in savings in computing time and storage. However, in contrast to the finite-difference method, relative little is known about the stability criteria of transient finite-element solutions. At the present time, most of the multi-dimensional finite element models for mixing in surface water bodies are developed using the Galerkin formulation in the Cartesian coordinate system. Triangular elements and linear interpolation functions were used to represent the spatial distribution of unknown variables. The Galerkin finite-element formulation will lead to a system of equations which is symmetric and positive definite. This type of system of equations can be solved by the "skyline" solver. Besides the Galerkin method, there exist other finite element methods such as the Rayleigh-Ritz method and collocation method. Smith, et. al. (39) discussed some advantages of the Galerkin method over the Rayleigh-Ritz method. Almost no study has been done using the collocation finite-element method to simulate multi-dimensional convection-diffusion problems. In the collocation method, the governing equation is exactly satisfied at the collocation points. The system of equations developed in the collocation finite element method is non-symmetric and positive semi-definite, which is more difficult to solve than the system of equations formed in the Galerkin finite element method. However, with new techniques developed for solving systems of non-symmetric equations and the fact that no integration over the spatial domain is required, the collocation method could be more efficient than the Galerkin method especially when the governing equation has variable coefficients. Houstis, et. al. (17) have compared the efficiency of collocation, Galerkin and least square finite element methods for elliptic partial differential equations, and conclude that the collocation method is more efficient for solutions with moderate accuracy.

In the present study a collocation finite-element method using rectangular bi-cubic elements is developed to solve the transient mixing equation in the orthogonal curvilinear (natural) coordinate system (45). This model is verified against analystical solutions. The scheme is applied to a reach of the Missouri River and compared with analytical and finite difference solutions and field measurements. The model is shown to be more efficient than existing finite difference models.

CHAPTER II

PROBLEM FORMULATION

GOVERNING EQUATIONS

As discussed in Chapter I, the orthogonal curvilinear coordinate system, or the natural coordinate system, developed by Chang (6), Fukuoka and Sayre (13), and Yotsukura and Sayre (45), will be used in this study. As shown in Figure 2.1, the natural coordinate system consists of naturally orthogonal longitudinal, transverse, and horizontal coordinate surfaces. The longitudinal and transverse coordinate surfaces are vertical and typically curved and nonparallel. The horizontal coordinate surfaces are all parallel horizontal planes. The longitudinal coordinate surface should be aligned closely with the depth-averaged total velocity vectors. The origin 0 is located at the intersection point of three selected coordinate surfaces. The intersection of horizontal and longitudinal coordinate surfaces forms the x-axis which is positive in the downstream direction. The z-axis is defined as the intersection of the transverse and horizontal coordinate surfaces and are positive to the right. The y-axis is the intersection of longitudinal and transverse coordinate surfaces and is positive in the upward direction.

The horizontal distances measured along different longitudinal (or transverse) coordinate surfaces from one transverse section (or longitudinal) to another are in general not equal. This is due to the curvature in channel alignment and/or variations in width along the channel. As shown in Figure 2.1, lengths of differential elements can be quantified by introducing the metric coefficients m_x and m_z. The differential distances along an arbitrary coordinate surface are:

$$dL_x = m_x dx \quad \ldots\ldots\ldots\ldots \quad (2.1)$$

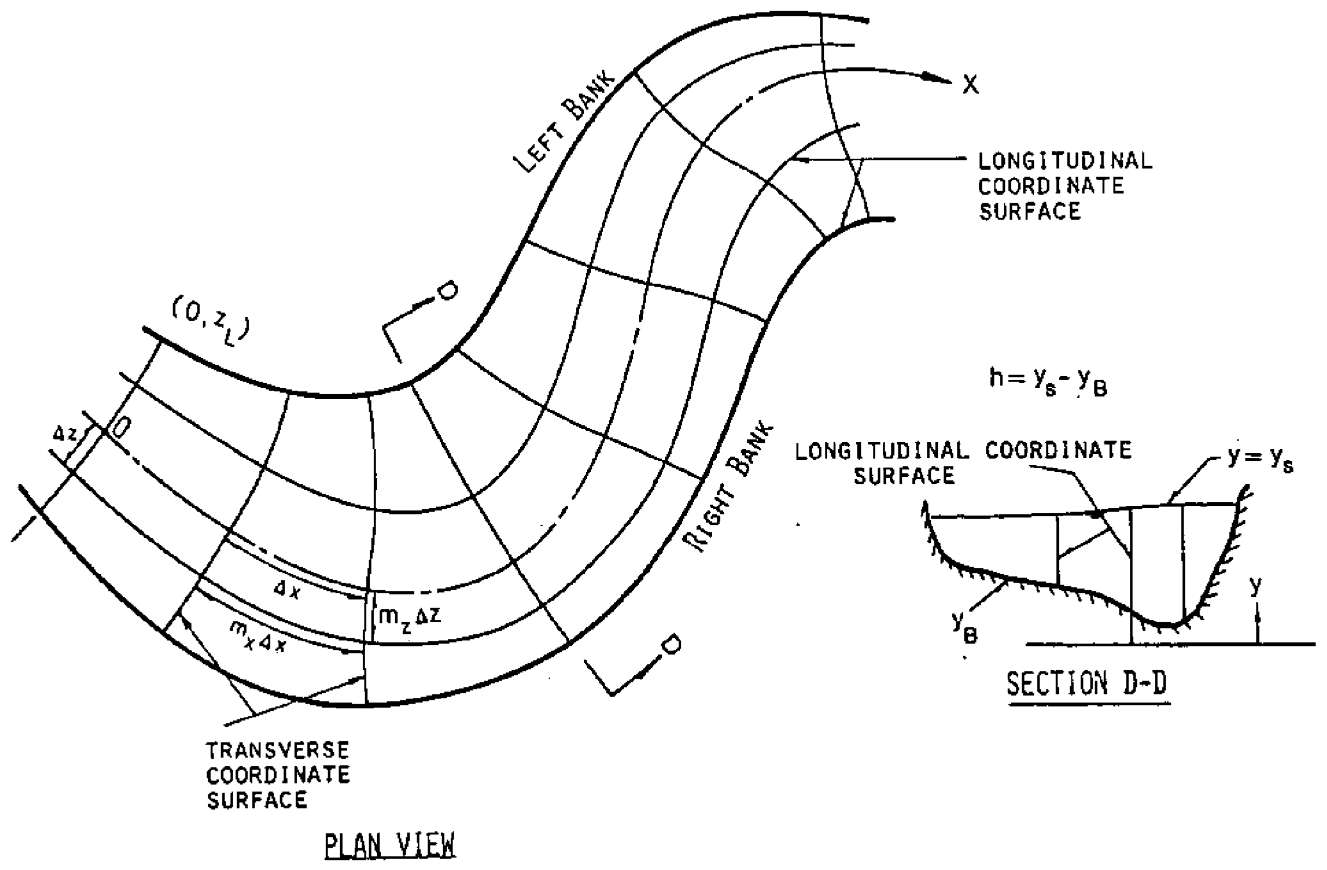


Figure 2.1. Orthogonal Coordinate System for Natural Channels

and

$$dL_z = m_z dz \quad \ldots \quad (2.2)$$

where m_x and m_z are generally a function of both x and z, and their values generally vary between 0.8 and 1.2.

The value of the metric coefficient m_y is always equal unity, since all horizontal surfaces are parallel, while the values of m_x and m_z vary from one point to another except on an x-axis and z-axis where $m_x = 1$ and $m_z = 1$.

In the natural coordinate system, the three-dimensional continuity equation for water and the convection-diffusion equation, with no restriction as to steadiness or uniformity of flow are (6):

$$\frac{\partial}{\partial x}(m_z u_x) + m_x m_z \frac{\partial u_y}{\partial y} + \frac{\partial}{\partial z}(m_x u_z) = 0 \quad \ldots \quad (2.3)$$

and

$$m_x m_z \frac{\partial c}{\partial t} + \frac{\partial}{\partial x}(m_z u_x c) + m_x m_z \frac{\partial(u_y c)}{\partial y} + \frac{\partial(m_x u_z c)}{\partial z} = \frac{\partial}{\partial x}\left(\frac{m_z}{m_x} \varepsilon_x \frac{\partial c}{\partial x}\right) + m_x m_z \frac{\partial}{\partial y}\left(\varepsilon_y \frac{\partial c}{\partial y}\right)$$

$$+ \frac{\partial}{\partial z}\left(\frac{m_x}{m_z} \varepsilon_z \frac{\partial c}{\partial z}\right) + \phi \quad \ldots \quad (2.4)$$

where u_x, u_y and u_z are the local velocity components in x, y, and z direction, ε_x, ε_y and ε_z are the local turbuelent mass diffusivities. c is the local solute concentration, ϕ is a source/sink term which is function of space and time. By integrating Eqs. 2.3 and 2.4 term by term over the depth flow from the bed $Y_B(x,z,t)$ to the surface $Y_s(x,z,t)$, Yotsukura and Sayre (45) have shown that the two dimensional depth integrated continuity and convection-diffusion equations are

$$m_x m_z \frac{\partial h}{\partial t} + \frac{\partial}{\partial x} [h \, \bar{u}_x m_z] + \frac{\partial}{\partial z} [h \, m_x \bar{u}_z] = 0 \quad \ldots\ldots\ldots\ldots\ldots \quad (2.5)$$

and

$$m_x m_z \frac{\partial}{\partial t} (C \, h) + \frac{\partial}{\partial x} (m_z v_x C \, h) + \frac{\partial}{\partial z} (m_x v_z C \, h) =$$

$$\frac{\partial}{\partial x}(\frac{m_z}{m_x} h \, E_x \frac{\partial C}{\partial x}) + \frac{\partial}{\partial z} (\frac{m_x}{m_z} h \, E_z \frac{\partial C}{\partial z}) - \lambda_1 C + \lambda_2 \quad \ldots\ldots\ldots\ldots \quad (2.6)$$

in which, C = depth averaged concentration; v_x, v_z = x,z components of the velocity vectors averaged over the local depth h; E_x, E_z = mixing coefficients that include the combined effects of depth-averaged turbuelent diffusion and convective diffusion; and λ_1 and λ_2 = decay constant and generation function, respectively.

This set of equations are quite general, in that, it is applicable to unsteady, non-uniform flow. By introducing the cumulative discharge, as suggested by Yotsukura and Sayre (45)

$$q_c = \int_{zL}^{z} m_z h v_x dz \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.7)$$

integrating Eq. 2.5 with respect to z from the left bank z_L to z, and substituting into Eq. 2.6, one can chow that the convective-diffusion equation for transient mixing in steady-state flow is

$$\frac{\partial C}{\partial t} + \frac{v_x}{m_x} \frac{\partial C}{\partial x} = \frac{1}{m_x m_z h} [\frac{\partial}{\partial x} (\frac{m_z}{m_x} hE_x \frac{\partial C}{\partial x}) + \frac{\partial}{\partial x} (\frac{m_z}{m_x} hE_x \frac{\partial C}{\partial q_c} \frac{\partial q_c}{\partial x})$$

$$+ \frac{\partial}{\partial q_c} (\frac{m_z}{m_x} hE_x \frac{\partial C}{\partial x}) \frac{\partial q_c}{\partial x} + \frac{\partial}{\partial q_c} (\frac{m_z}{m_x} hE_x \frac{\partial C}{\partial q_c} \frac{\partial q_c}{\partial x}) \frac{\partial q_c}{\partial x}]$$

$$+ \frac{v_x}{m_x} \frac{\partial}{\partial q_c} (m_x h^2 v_x E_z \frac{\partial C}{\partial q_c}) - \lambda_1 C + \lambda_2 \quad \ldots\ldots\ldots\ldots\ldots \quad (2.8)$$

By aligning the longitudinal coordinate surfaces in the direction of the depth-averaged local velocity vectors, and using the relationship $\partial q_c/\partial x = -m_x h v_z$, Eq. 2.8 can be reduced to yield the following equation for the stream-tube model

$$\frac{\partial C}{\partial t} + \frac{v_x}{m_x}\frac{\partial C}{\partial x} = \frac{1}{m_x m_z h}\frac{\partial}{\partial x}\left(\frac{m_z}{m_x} h E_x \frac{\partial C}{\partial x}\right)$$

$$+ \frac{v_x}{m_x}\frac{\partial}{\partial q_c}\left(m_x h^2 v_x E_z \frac{\partial C}{\partial q_c}\right) - \lambda_1 C + \lambda_2 \qquad \ldots\ldots\ldots\ldots\ldots \qquad (2.9)$$

Yotsukura and Sayre (45) have pointed out that the longitudinal mixing term containing E_x can be neglected in practice.

By introducing the following non-dimensional variables:

$$\tau = \frac{t u_o}{\ell_o}, \qquad \eta = \frac{x}{\ell_o}$$

$$\nu = \frac{v_x}{u_o}, \qquad \xi = \frac{q_c}{Q} \qquad \ldots\ldots \qquad (2.10)$$

$$\phi = \frac{h}{R}, \qquad \lambda_1^* = \frac{\lambda_1 \ell_o}{u_o}$$

$$\lambda_2^* = \frac{\lambda_2 \ell_o}{u_o C_o}, \qquad s = \frac{\bar{C}}{C_o}$$

where ℓ_o is a reference length, u_o is the mean velocity, R is the hydraulic radius, Q is the total discharge and C_o is reference concentration, Eq. 2.9 can be transformed to the following non-dimensional form:

$$\frac{\partial s}{\partial \tau} + \frac{\nu}{m_x} \frac{\partial s}{\partial \eta} = \frac{R}{m_x m_z \phi \ell_o} \frac{\partial}{\partial \eta} [D_L \frac{\partial s}{\partial \eta}]$$

$$+ \frac{\ell_o \nu R^3 u_o^2}{m_x Q^2} \frac{\partial}{\partial \xi} [D_T \frac{\partial s}{\partial \xi}] - \lambda_1^* s + \lambda_2^* \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2.11)$$

where $D_L = \frac{m_z \phi E_x}{m_x u_o R}$ and $D_T = \frac{m_x \phi^2 \nu E_z}{u_o R}$

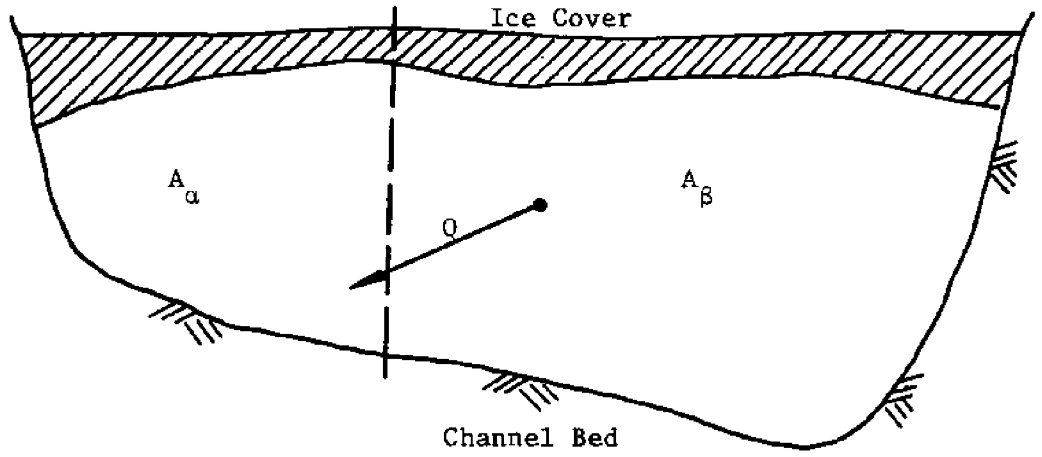
TRANSVERSE FLOW DISTRIBUTION

In order to use the natural coordinate system, it is necessary to know the transverse distribution of the discharge per unit width, q(z) or the cumulative transverse flow distribution. When there is no field measurement available, an anlaytical formula developed by Shen and Ackermann (37) can be used. This formula gives the cumulative discharge as

$$\frac{Q_\alpha}{Q} = [\frac{A_\beta}{A} \cdot \frac{\hat{Q}_\alpha}{Q} + \frac{A_\alpha}{A} (1 - \frac{\hat{Q}_\beta}{Q})] \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.12)$$

in which, Q_α = flow passing the partial cross-sectional area A_α; A_β = partial cross-section area, $A-A_\alpha$; $\hat{Q}_\alpha$ and $\hat{Q}_\beta$ = discharge through partial cross-sectional areas calculated by the formula $\frac{\hat{Q}_\alpha}{Q} = \frac{(A/R^{2/3})_\alpha}{AR^{2/3}}$.

Shen and Ackermann (37) tested this formula against measured data and found that it provides good agreement for both ice covered and free surface flow conditions. The ice covered case is of interest not only because of its application to river thermal conditions in the winter, but also due to the fact that the discharge of pollutants into rivers during the winter low flow period could lead to the worst water quality condition.



$$A = A_\alpha + A_\beta$$

$$Q = Q_\alpha + Q_\beta$$

Figure 2.2. Flow Distribution in a Channel Cross Section

COORDINATE TRANSFORMATION

Values of the different parameters in the physical domain are generally measured in the (x,y,z) coordinate system. To make use of the governing convection-diffusion equation in the natural coordinate system, i.e. in the dimensional x-q_c coordinate system, or in the η-ξ nondimensional coordinate system, all of the parameters, namely, m_x, m_z, E_x, E_z, h, ν and ϕ must be evaluated at the corresponding (x, q_c) points in the new coordinate system. A computerized procedure developed to perform this transformation is described in this section.

Consider a cross-section in natural coordinate system, with given values of the velocity $v_x(x_i,z_j)$ and depth $h(x_i,z_j)$ at different transverse distances in the cross-section as shown in Figure 2.1. The amount of unit-width discharge $q(z_i)$ at a vertical slice can be determined by

$$q(z_j) = v_x(z_j) \cdot h(z_j) \quad \ldots\ldots\ldots\ldots \quad (2.13)$$

For a cross-section located at x_1, the width of the channel is w_i and the total amount of discharge can be determined from the q(z) distribution as shown in Figure 2.3 by

$$Q_{TOT} = \sum_{j=0}^{n-2} \frac{(q(z_{j+1}) + q(z_j))}{2} * (z_{j+1} - z_j) \quad \ldots\ldots\ldots \quad (2.14)$$

in which, n = total number of verticals with measured q.

If $q_c(z)$ is the cumulative discharge at an arbitrary point z where $q_c(z)$ is defined as:

$$q_c(z) = \int_{z_L}^{z} h(\xi)\, v_x(\xi)\, m_\xi d\xi \quad \ldots\ldots\ldots\ldots \quad (2.15)$$

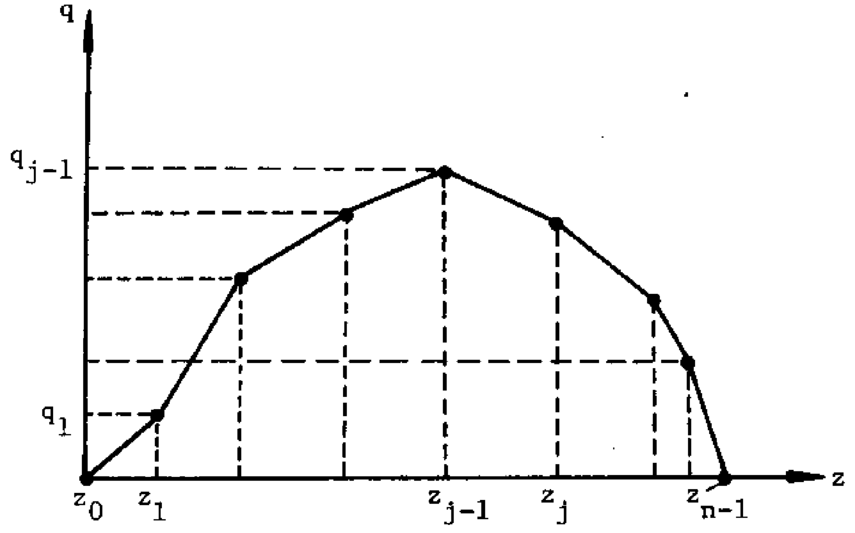


Figure 2.3. Transverse Flow Distribution in a Cross Section

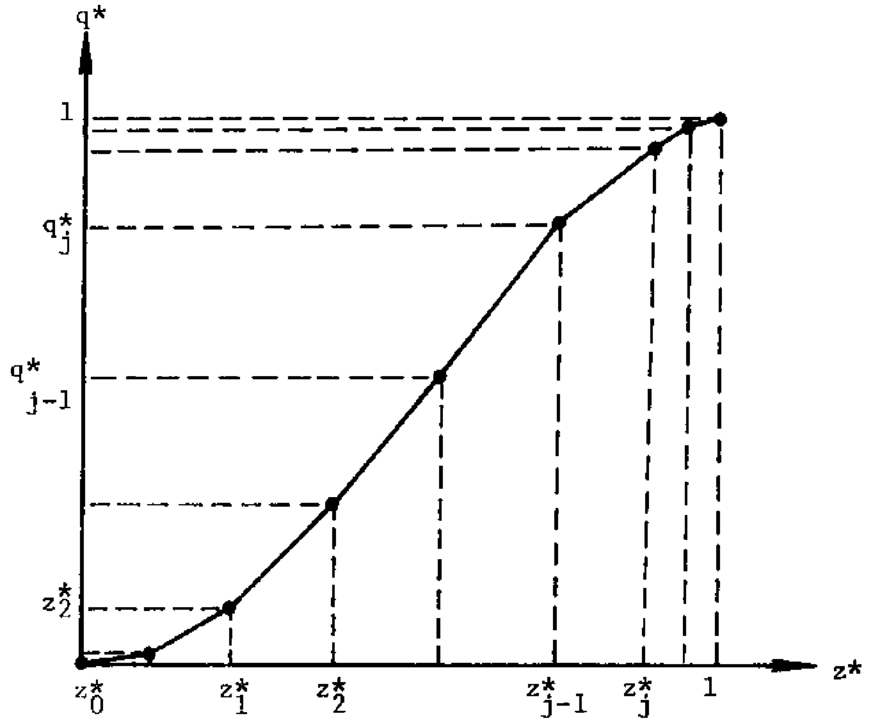


Figure 2.4. Cumulative Discharge Curve

then at $z = z_L$, $q_c(z_L) = 0$, and $q_c(z_R) = Q_{TOT}$. Figure 2.4 shows the relation between $q_c(z)$ and z, and for $z_{j-1} \leq z \leq z_j$ the cumulative discharge is determined by,

$$q_c(z) = [\sum_{n=1}^{j-1} (z_n - z_{n-1})(v_x(z_{n-1}) \cdot h(z_{n-1}) + v_x(z_n) \cdot h(z_n))/2]$$

$$+ [\frac{(v_x(z_j) \cdot h(z_j) - v_x(z_{j-1}) \cdot h(z_{j-1}))}{(z_j - z_{j-1})} (z - z_{j-1})$$

$$+ 2v_x(z_{j-1})\, h(z_{j-1})]\ (z - z_{j-1})/2 \quad \ldots\ldots\ldots\ldots\ldots \quad (2.16)$$

Let $\xi_j^* = \frac{z_j}{w_i}$, and

$$q_c^*(\xi_j^*) = \frac{q_c(\xi_j)}{Q_{TOT}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.17)$$

The values of ξ_j^* and $q_c^*(\xi_j^*)$ are then varied from 0 to 1. If the q_c^* coordinate is divided into K equal segments such that $\Delta q_c^* = 1/K$, with $q_c^*(0) =$ 0 and $q_c^*(1) = 1$, the following relation between ξ_k^* and q_c^* is then obtained for $\xi_{j-1}^* \leq \xi_k^* \leq \xi_j^*$, and $1 \leq k \leq K$.

$$\xi_k^* = \xi_{i-1}^* + \frac{(q_{c_k}^* - q_{c_{j-1}}^*)}{(q_{c_j}^* - q_{c_{j-1}}^*)} (\xi_j^* - \xi_{j-1}^*) \quad \ldots\ldots\ldots\ldots\ldots \quad (2.18)$$

Similarly, the depth at the point ξ_k^*, can be calculated by

$$h(\xi_k^*) = h(\xi_{j-1}^*) + \frac{[(h(\xi_j^*) - h(\xi_{j-1}^*)) \cdot (\xi_k^* - \xi_{j-1}^*)]}{(\xi_j^* - \xi_{j-1}^*)} \quad \ldots\ldots\ldots \quad (2.19)$$

and consequently, $\phi(\xi_k^*)$ can be written as:

$$\phi(\xi_k^*) = \frac{h(\xi_k^*)}{R} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.20)$$

where R is the hydraulic radius.

The velocity $v(\xi_k^*)$ can be determined from

$$v(\xi_k^*) = \frac{q^*(\xi_k^*) \cdot Q_{TOT}}{w_i \cdot h(\xi_k^*)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.21)$$

where the amount of discharge $q^*(\xi_K^*)$ at the boundary of the kth stream-tube is given by,

$$q^*(\xi_k^*) = \frac{1}{2}\left[\frac{\Delta q_c^*}{(\xi_k^* - \xi_{k-1}^*)} + \frac{\Delta q_c^*}{(\xi_{k+1}^* - \xi_k^*)}\right] \quad \ldots\ldots\ldots\ldots \quad (2.22)$$

The value of the transverse mixing coefficients is determined by

$$E_z = \beta\, h\, u_* \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.23$$

where u_* is the shear velocity defined as,

$$u_* = \sqrt{gRS_b} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.24)$$

in which g is the acceleration gravity, S_b is the bed slope and R is the hydraulic radius. Different values of the constant β are given by many authors. Yotsukura, Fischer and Sayre (47) reported a value of about 0.6 for the constant β as observed in the Missouri River.

The value of the longitudinal diffusion coefficient can be estimated by using the following formula,

$$E_x = \alpha\, h\, u_* \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.25)$$

with a value of α equal to 6.0 for this study.

Yotsukura and Sayre (35) suggested the following formulas to evaluate the metric coefficients m_x and m_z,

$$m_x = \frac{L_L}{L} + \left(\frac{L_R - L_L}{L}\right) \cdot \left[\left(\frac{L_k}{w}\right)_i + \left\{\left(\frac{L_k}{w}\right)_{i+1} - \left(\frac{L_k}{w}\right)_i\right\} \cdot \left(\frac{p\Delta x}{L}\right)\right] \quad \ldots\ldots \quad (2.26)$$

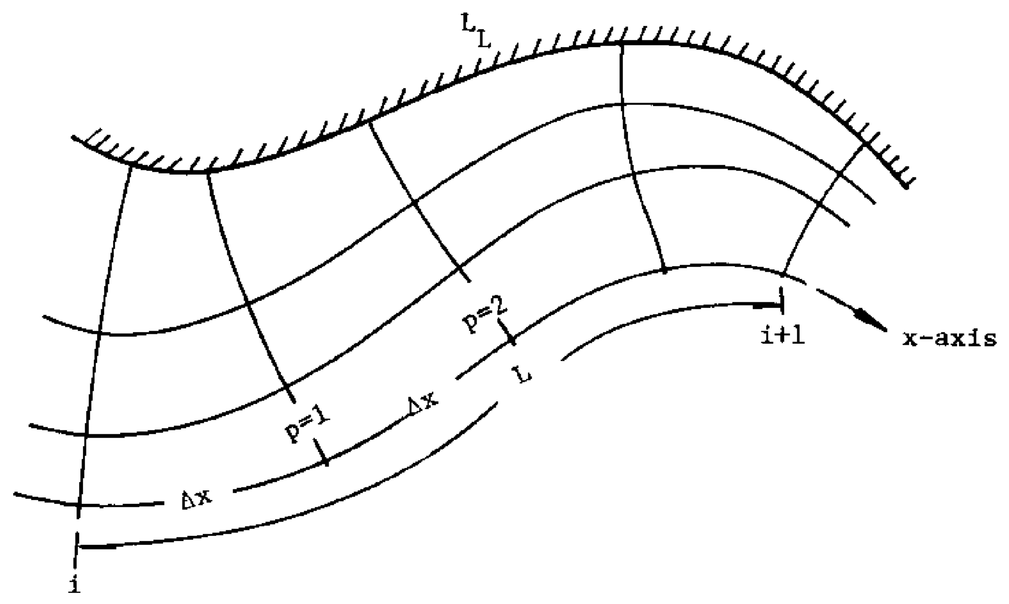


Figure 2.5. Plan View of the Stream Tube System

and $m_z \approx \frac{w_i}{w_o}$ . . . . (2.27)

where the index i refers to the cross sections where velocity and geometry measurements are obtained. L, L_L and L_R are longitidunal distances between the ith and (i+1) cross section measured along the longitudianl coordinate surfaces through the channel center line, left bank and right bank of the natural river, respectively. $(L_k/w)_i$ is the fractional transverse distance from the left bank to the boundary between kth and (k+1)th stream tubes at the ith measurement cross section. p is the number of steps of uniform length (Δx) measured along the x-axis downstream from ith cross section as shown in Figure 2.5.

The procedure discussed can be used to determine all the parameters in the x-q_c coordinate system (or the η-ξ coordinate system), and it has to be repeated for each cross section. These parameters are then interpolated linearly in both the x-direction and the q_c direction to find values of the parameters at any specific points, namely the Gaussian points.

CHAPTER III

THE COLLOCATION FINITE ELEMENT SCHEME

MATHEMATICAL BACKGROUND

The mathematical models for two-dimensional convection-diffusion processes are generally difficult to solve in closed form (i.e. as a finite combination of 'nice' functions such as polynomials, exponentials, sine, cosine, etc...). Let 'C' be an exact solution for problem 'P', and 'A' be an approximate solution. Most of the approximation techniques including collocation method seek an approximation solution 'A' instead of the exact solution 'C'.

Generally, a solution is carried out in three major steps:

(i) Choose n linearly independent functions $\phi_1, \phi_2, \ldots, \phi_n$ and ask that

$$A(x) = \sum_{i=1}^{n} \alpha_i \phi_i (x) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.1)$$

It is worthwhile to note that these n-independent functions $\phi_1, \ldots, \phi_n$ generate a space of finite dimension n, call it S, and those function $\phi_1, \ldots, \phi_n$ are basis functions for S, which means that, the exact solution 'C' belongs to a large 'infinite' dimensional space X of functions, and $S \subset X$. If basis functions $\phi_1, \ldots, \phi_n$ are chosen to be piecewise polynomials, then the solution procedure would become the finite element method, and that choice is responsible for the success of the method. These choices have the common characteristics, that is: each ϕ_i is nonzero over only a small part of the domain Ω, and zero over the remaining 'large' part of Ω.

(ii) Construct a system of n-equations, with respect to the n-unknowns $\alpha_1, \ldots, \alpha_n$. Such systems can be written as,

$$K\vec{\alpha} = \vec{b} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.2)$$

in which,

$K = (k_{ij})_{i,j=1}^{n}$ is a known n x n matrix,

$\vec{b} = (b_j)_{j=1}^{n}$ is a known vector,

$\vec{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_n]^T = (\alpha_j)_{j=1}^{n}$ is a vector of unknown coefficients.

(iii) Solve the resulting system of equations, find $\alpha_1, \alpha_2, \ldots, \alpha_n$, and consequently the approximation solution A(x).

Very often, the elemental matrix K has a structure which is difficult to be stored in a computer, such as in the case of the collocation method. The choice of the method of solving this system (either using a directed method or an iterative method) is responsible for optimizing computer time and memory storage.

THE FINITE ELEMENT COLLOCATION METHOD

Let, L, B be differential operators such that

$$LA(x) = f(x) \quad , \quad x \in \Omega \text{ (domain)} \quad \ldots\ldots\ldots\ldots\ldots \quad (3.3)$$

$$BA(x) = g(x) \quad , \quad x \in \delta\Omega \text{ (boundary)} \quad \ldots\ldots\ldots\ldots \quad (3.4)$$

In the collocation method, the residual LA(x) - f(x) is forced to be zero at a number of points in Ω (interior collocation points = Int.), and the boundary residual BA(x) - g(x) is forced to be zero at a number of points in

$\delta\Omega$ (boundary collocation points = n_b). For a total of n-equations, the total number of collocation points must be equal to n. Accordingly, the n-collocation equations are

$$LA(\sigma_i) = f(\sigma_i) \; ; \text{ at } n_{Int} \text{ points } \sigma_i \epsilon\ \Omega \quad \ldots\ldots\ldots\ldots\ldots \tag{3.5}$$

and

$$BA(\sigma_j) = g(\sigma_j) \; ; \text{ at } n_b \text{ points, } \sigma_j\ \epsilon\ \delta\Omega \quad \ldots\ldots\ldots\ldots\ldots \tag{3.6}$$

where $n_{Int} + n_b = n$. The one-dimensional collocation finite element formulation have been discussed in detail in Ref. 25. In the following paragraphs the one-dimensional formulation will be summarized and generalized to the two-dimensional formulation.

ONE DIMENSIONAL FORMULATION. To define the Hermite cubic elements in one dimensional case, let $\Omega = (a,b)$ and $\bar{\Omega} = \Omega \cup \delta\Omega = [a,b]$. The closed interval $\bar{\Omega}$ is subdivided into N subintervals I_i where

$$I_i = [x_i, x_{i+1}], \; i = 1, \ldots, N$$

and the length of each subinterval is $h_i = x_{i+1} - x_i$, as shown in Figure 3.1 .

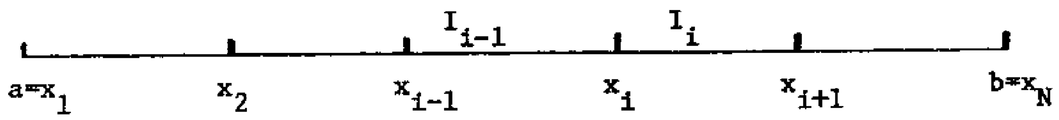


Figure 3-1. Sketch of the element I_i

For Hermite cubic elements, the space S consists of all functions in C' [0,1] which reduce to a cubic over every subinterval. A total of 4N coefficients, or four coefficients per subinterval, are needed. Since two continuity (compatability) conditions are available at each interior node, the number of free coefficients are equal to $n = 2(N+1)$. The n basis functions $\phi_1, \ldots, \phi_n$ can be constructed by associating two functions to each node, as shown in Figure 3.2.

Any cubic polynomial can be written as,

$$P_3(x) = a_o + a_1 x + a_2 x^2 + a_3 x^3$$

For Hermite cubic basis functions, the values of the coefficients a_i, $i = 0,$ 1,2,3 can be determined by satisfying certain requirements for the values of the basis functions, ϕ and its derivatives, at the two end points of each element. The requirements for ϕ_{2i-1} are:

$$\phi_{2i-1}(x_i) = 1 \text{ and } \phi_{2i-1}(x_j) = 0 \quad , \quad j \neq i \qquad \ldots\ldots\ldots\ldots \quad (3.7)$$

$$\left.\begin{array}{l} \phi'_{2i-1}(x_j) = 0 \text{ for all } j \\ \text{and} \\ \phi_{2i-1}(x) = 0 \text{ for } x \notin I_{i-1} \cup I_i \end{array}\right\} \qquad \ldots\ldots\ldots\ldots \quad (3.8)$$

and

$$\left.\begin{array}{l} \phi_{2i-1} \text{ is a cubic polynomial on } I_{i-1} \\ \phi_{2i+1} \text{ is a cubic polynomial on } I_i \end{array}\right\} \qquad \ldots\ldots\ldots\ldots \quad (3.9)$$

The requirements for ϕ_{2i} are the same, except that:

$$\begin{array}{ll} \phi_{2i}(x_j) = 0 & \text{for all } j \\ \phi'_{2i}(x_j) = 1 & \text{for all } j = i \end{array} \qquad \ldots\ldots\ldots\ldots \quad (3.10)$$

and

$$\phi'_{2i}(x_j) = 0 \quad \text{for } j \neq i$$

With these conditions the basis functions can be obtained for each element as

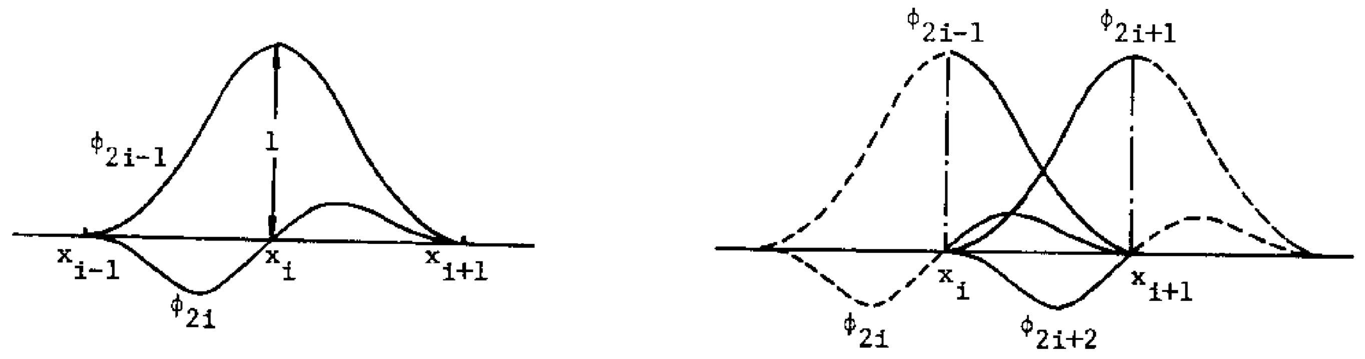


Figure 3.2. Hermite Cubic Basis Functions

$$\phi_{2i-1}(x) = \begin{cases} -2\left(\frac{x-x_i}{h_{i-1}}\right)^3 - 3\left(\frac{x-x_i}{h_{i-1}}\right)^2 + 1 , & x \in I_{i-1} \\ 2\left(\frac{x-x_i}{h_i}\right)^3 - 3\left(\frac{x-x_i}{h_i}\right)^2 + 1 , & x \in I_i \\ 0 & \text{otherwise} \end{cases} \quad \ldots\ldots \quad (3.11)$$

and

$$\phi_{2i}(x) = \begin{cases} \left[\left(\frac{x-x_i}{h_{i-1}}\right)^3 + 2\left(\frac{x-x_i}{h_{i-1}}\right)^2 + \left(\frac{x-x_i}{h_{i-1}}\right)\right] h_{i-1} , & x \in I_{i-1} \\ \left[\left(\frac{x-x_i}{h_i}\right)^2 - 2\left(\frac{x-x_i}{h_i}\right)^2 + \left(\frac{x-x_i}{h_i}\right)\right] h_i , & x \in I_i \\ 0 & \text{otherwise} \end{cases} \quad \ldots \quad (3.12)$$

Since only ϕ_{2i-1}, ϕ_{2i}, ϕ_{2i+1}, and ϕ_{2i+2} are nonzero over I_i, the approximate solution $A \in S$ can be written as:

$$\begin{aligned} A(x) &= \sum_{j=1}^{n} \alpha_j \phi_j (x) \\ &= \sum_{j=2i-1}^{2i+1} \alpha_j \phi_j (x) \end{aligned} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.13)$$

consequently, I_i is an element with four degrees of freedom. The coordinates of the Gaussian points are

$$\sigma_{2i} = \frac{x_i + x_{i+1}}{2} - \frac{1}{\sqrt{3}}\frac{h_i}{2}$$

and $\ldots\ldots\ldots\ldots\ldots\ldots$ (3.14)

$$\sigma_{2i+1} = \frac{x_i + x_{i+1}}{2} + \frac{1}{\sqrt{3}}\frac{h_i}{2}$$

By substituting Eq. 3.13 into the governing differential equation, and using Gaussian points as the interior collocation points, a system of 2N linear equations can be formed. This system of equations include the following elemental matrices.

$$K_i^{(\ell)} = \left\{ D^{\ell} \phi_k (\sigma_{2i}) \right\}_{k=2i-1,\ j=2i}^{2i+1,\ 2i+1} , \text{ for } \ell = 0,1,2 \quad \ldots\ldots\ldots \quad (3.15)$$

or,

$$K_i^{(\ell)} = \begin{bmatrix} \phi_{2i-1}^{(\ell)}(\sigma_{2i}) & \phi_{2i}^{(\ell)}(\sigma_{2i}) & \phi_{2i+1}^{(\ell)}(\sigma_{2i}) & \phi_{2i+2}^{(\ell)}(\sigma_{2i}) \\ \phi_{2i-1}^{(\ell)}(\sigma_{2i+1}) & \phi_{2i}^{(\ell)}(\sigma_{2i+1}) & \phi_{2i+1}^{(\ell)}(\sigma_{2i+1}) & \phi_{2i+2}^{(\ell)}(\sigma_{2i}) \end{bmatrix}$$

in which, D^{ℓ} is defined as the ℓth derivative of a function, $D^{\ell} = \frac{d^{\ell}}{dx^{\ell}}$.

By using Eqs. 3.11 and 3.12 the following form of the collocation elemental matrices for the element I_i can be obtained,

$$K_{(x)}^{(0)} = \begin{bmatrix} \alpha & h_i\beta & 1-\alpha & -h_i\bar{\beta} \\ 1-\alpha & h_i\bar{\beta} & \alpha & -h_i\beta \end{bmatrix} \quad ;$$

$$\alpha = \frac{9 + 4\sqrt{3}}{18} \quad , \beta = \frac{3 + \sqrt{3}}{36} \quad , \bar{\beta} = \frac{3 - \sqrt{3}}{36} \quad \ldots\ldots\ldots\ldots \quad (3.16)$$

$$K_{(x)}^{(1)} = \begin{bmatrix} -1/h_i & \sqrt{3}/6 & 1/h_i & -\sqrt{3}/6 \\ -1/h_i & -\sqrt{3}/6 & 1/h_i & \sqrt{3}/6 \end{bmatrix} \quad \ldots\ldots\ldots\ldots \quad (3.17)$$

and

$$K_{(x)}^{(2)} = \begin{bmatrix} -\alpha''/h_i^2 & -\beta''/h_i & \alpha''/h_i^2 & -\bar{\beta}''/h_i \\ \alpha''/h_i^2 & \beta''/h_i & -\alpha/h_i^2 & \beta''/h_i \end{bmatrix} \quad ;$$

$$\alpha'' = 2\sqrt{3} \ , \ \beta'' = \sqrt{3} + 1, \ \bar{\beta}'' = \sqrt{3} - 1 \quad \ldots\ldots\ldots\ldots \quad (3.18)$$

With the 2N equations obtained for the interior collocation points and two equations from boundary conditions, the 2(N+1) unknown coefficients of A can be determined for each time step.

<u>TWO DIMENSIONAL FORMULATION</u>. A two-dimensional formulation can be developed by generalizing the one-dimensional formulation. In the two-dimensional case, the region Ω is divided into a finite number of rectangular elements. Bicubic Hermite basis functions, which are products of two Hermite cubic piecewise polynomials of one variable, are used. Associated with each node there are four basis functions:

$$\begin{aligned}
B_{ij}^{(0)}(x,y) &= \phi_{2i-1,2j-1}(x,y) \equiv \phi_{2i-1}(x)\ \phi_{2j-1}(y) \\
B_{ij}^{(1)}(x,y) &= \phi_{2i,2j-1}(x,y) \equiv \phi_{2i}(x)\ \phi_{2j-1}(y) \qquad \ldots\ldots\ldots\ldots \qquad (3.19) \\
B_{ij}^{(2)}(x,y) &= \phi_{2i-1,2j}(x,y) \equiv \phi_{2i-1}(x)\ \phi_{2j}(y)
\end{aligned}$$

and

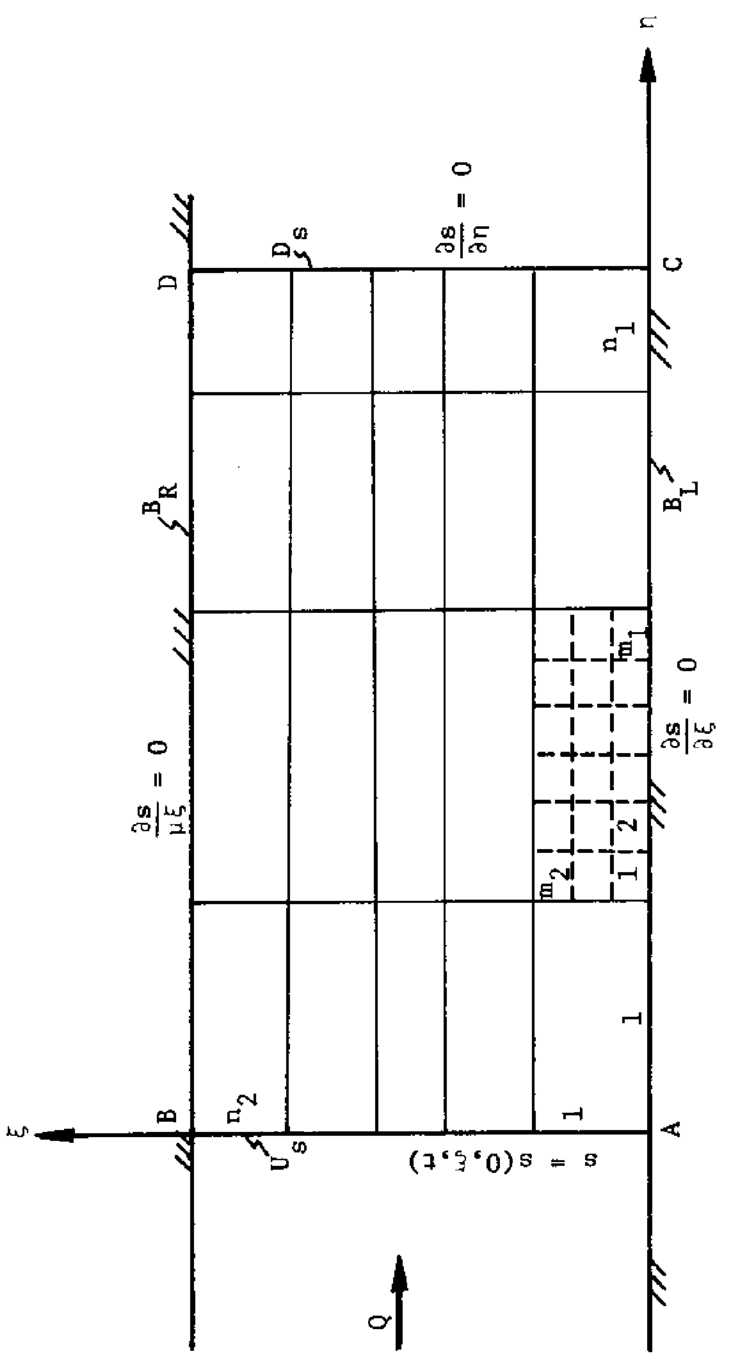
$$B_{ij}^{(1,2)}(x,y) = \phi_{2i,2j}(x,y) \equiv \phi_{2i}(x)\ \phi_{2j}(y)$$

The approximate solution can be written as,

$$A(x,y) = \sum_{\substack{k=i,i+1 \\ \ell=j,j+1}} \ \sum_{\substack{r=2k-1,2k \\ t=2\ell-1,2\ell}} \alpha_{rt}\ \phi_r(x)\ \phi_t(y) \qquad \ldots\ldots\ldots \qquad (3.20)$$

Each rectangular element has sixteen degrees of freedom, four for each node.

The boundary value problem for predicting concentration distribution in rivers is defined by the governing differential equations, and boundary and initial conditions in the two-dimensional connected domain Ω and on the boundary $\delta\Omega$, Fig. 3.3.



ξ
η
A
B
C
D
B_R
B_L
D_s
U_s
n_1
n_2
m_1
m_2
1
2
Q
$\frac{\partial s}{\mu\xi} = 0$
$\frac{\partial s}{\partial \eta} = 0$
$\frac{\partial s}{\partial \xi} = 0$
$s = s(0,\xi,t)$

Figure 3.3. Discretization of the Solution Domain

$$\frac{\partial s}{\partial \tau} + f_1(\eta,\xi)\,\frac{\partial s}{\partial \eta} = f_2(\eta,\xi)\,\frac{\partial}{\partial \eta}\left[D_L\,\frac{\partial s}{\partial \eta}\right] +$$

$$f_3(\eta,\xi)\,\frac{\partial}{\partial \xi}\left[D_T\,\frac{\partial s}{\partial \xi}\right] - f_4(\eta,\xi,\tau)s +$$

$$f_5(\eta,\xi,\tau) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.21)$$

in which, $f_1(\eta,\xi) = \nu/m_x$; $f_2(\eta,\xi) = R/(m_x m_z \phi)$; $f_3(\eta,\xi) = (\ell_o \nu\, R^3\, u_o^2)/(m_x Q^2)$; $f_4(\eta,\xi,\tau) = \lambda_1^*$; $f_5(\eta,\xi,\tau) = \lambda_2^*$; $D_L = (m_z \phi E_x)/(m_x u_o R)$; and $D_T = (m_x \phi^2 \nu E_z)/(u_o R)$.

$$s(0,\xi,\tau) = g_1(\xi,\tau) \quad ; \quad \text{for } \tau \geq 0 \quad \ldots\ldots\ldots\ldots \quad (3.22)$$

$$\frac{\partial s}{\partial n} = 0 \quad ; \quad \text{on } B_L,\ B_R,\ D_s \text{ for } \tau \geq 0 \quad \ldots\ldots\ldots\ldots \quad (3.23)$$

and,

$$s(\eta,\xi,0) = 0 \text{ on } \Omega \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.24)$$

A solution of the problem in the space domain is accomplished by applying the approximate solution A to Equations 3.21 to 3.24. It is of interest to mention that the two-dimensional convection-diffusion equation in the η-ξ coordinate system is a linear, non-symmetric, elliptic partial differential equation. The non-symmetry arises from the convection terms which have been the principal source of difficulty in numerical solutions of the problems of this type.

In order to be able to use different element sizes in different regions of the solution domain, the domain Ω is first divided into n_1 and n_2 divisions along the η-axis and the ξ-axis, respectively. These divisions, which are not necessary equal, are then further divided into m_1 and m_2 equal subdivisions with lengths h_i and k_i, respectively. Such divisions yield a total of N rectangular elements.

$$N = N_x \cdot N_y = \sum_{i=1}^{n_1} (m_1)_i \cdot \sum_{j=1}^{n_2} (m_2)_j \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.25)$$

The total number of elements along the η-direction is N_x, and that along the ξ-direction is equal to N_y. Since there are four functions associated with each node, it is required to construct M equations to evaluate the $M = 4(N_x+1)(N_y+1)$ coefficients of the approximate solution

$$A(\eta,\xi,\tau) = \sum_{i=1}^{M} \alpha_i(\tau)\ \phi_i(\eta,\xi) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.26)$$

In terms of tensor products and elemental matrices, the following expressions are obtained from the approximate solution $A(\eta,\xi,\tau)$.

$$s(\eta,\xi,\tau) = [K_\eta^{(0)} \otimes K_\xi^{(0)}]\ \vec{\alpha}(\tau)$$

$$\frac{\partial s}{\partial t}(\eta,\xi,\tau) = [K_\eta^{(0)} \otimes K_\xi^{(0)}]\ \frac{\partial}{\partial t}\ \vec{\alpha}(\tau)$$

$$\frac{\partial s}{\partial \eta}(\eta,\xi,\tau) = [K_\eta^{(1)} \otimes K_\xi^{(0)}]\ \vec{\alpha}(\tau) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.27)$$

$$\frac{\partial s}{\partial \xi}(\eta,\xi,\tau) = [K_\eta^{(0)} \otimes K_\xi^{(1)}]\ \vec{\alpha}(\tau)$$

$$\frac{\partial^2 s}{\partial \eta^2}(\eta,\xi,\tau) = [K_\eta^{(2)} \otimes K_\xi^{(0)}]\ \vec{\alpha}(\tau)$$

$$\frac{\partial^2 s}{\partial \xi^2}(\eta,\xi,\tau) = [K_\eta^{(0)} \otimes K_\xi^{(2)}]\ \vec{\alpha}(\tau)$$

where each $[\ \otimes\ ]$ represents 4 x 16 matrix of given coefficient and $\vec{\alpha}(\tau)$ is a 16 x 1 vector of unknown functions of time to be determined.

By assembling the preceeding relations, the governing equation, Eq. 3.21 becomes,

$$[H]\ \left(\frac{\partial \vec{\alpha}}{\partial \tau}\right) = [K]\ \vec{\alpha}(\tau) + f_5(\eta,\xi,\tau) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.28)$$

where

$$[H] = [K_\eta^{(0)} \otimes K_\xi^{(0)}]$$

$$[K] = (-f_1(\eta,\xi) + f_2(\eta,\xi) \frac{\partial D_L}{\partial \eta}) \cdot [K_\eta^{(1)} \otimes K_\xi^{(0)}] + f_2(\eta,\xi)\, D_L \cdot$$

$$[K_\eta^{(2)} \otimes K_\xi^{(0)}] + f_3(\eta,\xi)\, D_T \cdot [K_\eta^{(0)} \otimes K_\xi^{(2)}] +$$

$$f_3(\eta,\xi) \frac{\partial D_T}{\partial \xi} \cdot [K_\eta^{(0)} \otimes K_\xi^{(1)}] - f_4(\eta,\xi,\tau) \cdot [K_\eta^{(0)} \otimes K_\xi^{(0)}]$$

At $\tau=0$, the initial values of $\vec{\alpha}(\tau)$, $(\vec{\alpha}(\tau)_o)$, can be obtained from the initial condition $A(\eta,\xi,0) = 0$. Matrices [H] and [K] are constructed from elemental matrices. Various methods can be used to obtain the relationship between $(\alpha(\tau)_o)$ and $(\alpha(\tau)_1)$, the solution at $\Delta\tau$. In the present study, the following implicit formula is used to approximate Eq. 3.28,

$$[H](\theta(\frac{\partial\alpha}{\partial t})_1 + (1-\theta)(\frac{\partial\alpha}{\partial t})_o) = [K](\theta\ \alpha(\tau)_1 + (1-\theta)\ \alpha(\tau)_o) + f_5(\eta,\xi,\tau) \quad . \ . \ (3.29)$$

where θ is a scalar parameter. When $\theta = 1/2$, this becomes the well-known Crank-Nicolson method. The forward and backward differencing in time yields,

$$[H](\frac{\partial\vec{\alpha}}{\partial\tau})_1 \approx [H](\frac{\vec{\alpha}(\tau)_1 - \vec{\alpha}(\tau)_{1/2}}{\Delta\tau/2}) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.30)$$

and,

$$[H](\frac{\partial\vec{\alpha}}{\partial\tau})_0 \approx [H](\frac{\vec{\alpha}(\tau)_{1/2} - \vec{\alpha}(\tau)_0}{\Delta\tau/2}) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.31)$$

Substituting Equations 3.30 and 3.31 into Eq. 3.29 one gets

$$[H]((\frac{\partial\vec{\alpha}}{\partial\tau})_1 + (\frac{\partial\vec{\alpha}}{\partial\tau})_0) \approx [H] \cdot \frac{2}{\Delta t} (\vec{\alpha}(\tau)_1 - \vec{\alpha}(\tau)_0)$$

$$\approx [K](\vec{\alpha}(\tau)_1 + \vec{\alpha}(\tau)_0) + 2\ f_5(\eta,\xi,\tau) \quad \ldots\ldots \quad (3.32)$$

which can be generalized to

$$[A]\ \vec{\alpha}(\tau)_n = [D]\ \vec{\alpha}(\tau)_{n-1} + 2\ f_5(\eta,\xi,\tau) \quad \ldots\ldots\ldots\ldots\ldots \quad (3.33)$$

where

$$[A] = \frac{2}{\Delta\tau}\ [H] - [K]$$

$$[D] = \frac{2}{\Delta\tau}\ [H] + [K]$$

This system of equations together with boundary conditions will be used to solve for coefficients $\alpha(\tau)_n$, in order to find the approximate solution $A(\eta,\xi,\tau)$. It should be noted that Eq. 3.33 is a system of linear equations in which matrix [A] and [D] are non-symmetrical, not positive definite and sparse in general.

Boundary conditions can be formulated through the use of Eq. 3.27. At the upstream boundary, Eqs. 3.22 and 3.27 give

$$\begin{aligned} g_1(\xi,\tau) &\simeq A(\eta,\xi,\tau) \\ &= [K_\eta^{(0)} \otimes K_\xi^{(0)}]\ \vec{\alpha}(\tau) \\ &= \{[\phi_{2k-1}(\eta)\ \phi_{2k}(\eta)\ \phi_{2k+1}(\eta)\ \phi_{2k+2}(\eta)] \\ &\quad \otimes [\phi_{2\ell-1}(\xi)\ \phi_{2\ell}(\xi)\ \phi_{2\ell+1}(\xi)\ \phi_{2\ell+2}(\xi)]\}\ \vec{\alpha}(\tau) \quad \ldots\ldots \quad (3.34) \end{aligned}$$

in which,

$$\vec{\alpha}(t) = [\alpha_{2k-1,2\ell-1}(\tau) \quad \alpha_{2k-1,2\ell}(\tau) \quad \alpha_{2k-1,2\ell+1}(\tau) \quad \alpha_{2k-1,2\ell+2}(\tau)$$

$$\alpha_{2k,2\ell-1}(\tau) \quad \alpha_{2k,2\ell}(\tau) \quad \alpha_{2k,2\ell+1}(\tau) \quad \alpha_{2k,2\ell+2}(\tau)$$

$$\alpha_{2k+1,2\ell-1}(\tau) \quad \alpha_{2k+1,2\ell}(\tau) \quad \alpha_{2k+1,2\ell+1}(\tau) \quad \alpha_{2k+1,2\ell+2}(\tau)$$

$$\alpha_{2k+2,2\ell-1}(\tau) \quad \alpha_{2k+2,2\ell}(\tau) \quad \alpha_{2k+2,2\ell+1}(\tau) \quad \alpha_{2k+2,2\ell+2}(\tau)]^T$$

Since along the boundary, $\eta = \eta_k$, $\phi_{2k}(\eta_k) = \phi_{2k+1}(\eta_k) = \phi_{2k+2}(\eta_k) = 0$, and $\phi_{2k-1}(\eta_k) = 1$, Eq. 3.34 can be reduced to

$$K_\xi^{(0)}\, \vec{\alpha}(\tau) = g_1(\xi,\tau) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.35)$$

in which,

$$\vec{\alpha}(\tau) = [\alpha_{2k-1,2\ell-1}(\tau) \quad \alpha_{2k-1,2\ell}(\tau) \quad \alpha_{2k-1,2\ell+1}(\tau) \quad \alpha_{2k-1,2\ell+2}(\tau)]^T$$

At the two corner points (η_k,ξ_ℓ), since $\phi_{2\ell-1}(\xi_\ell) = 1$ and $\phi_{2\ell}(\xi_\ell) = \phi_{2\ell+1}(\xi_\ell) = \phi_{2\ell+2}(\xi_\ell) = 0$, Eq. 3.34 can be reduced further to

$$g_1(\xi_\ell,\tau) = \phi_{2k-1}(\eta_k)\, \phi_{2\ell-1}(\xi_\ell)\, \alpha_{2k-1,2\ell-1}(\tau)$$

$$= \alpha_{2k-1,2\ell-1}(\tau) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.36)$$

For left and right banks, Eqs. 3.23 and 3.27 gives,

$$\frac{\partial s}{\partial \xi}(\eta,\xi,\tau) = 0 \doteq \frac{\partial A}{\partial \xi}(\eta,\xi,\tau)$$

$$= [K_\eta^{(0)} \otimes K_\xi^{(1)}]\, \vec{\alpha}(\tau)$$

$$= \{[\phi_{2k-1}(\eta)\ \phi_{2k}(\eta)\ \phi_{2k+1}(\eta)\ \phi_{2k+2}(\eta)] \otimes$$

$$[\phi'_{2\ell-1}(\xi)\ \phi'_{2\ell}(\xi)\ \phi'_{2\ell+1}(\xi)\ \phi'_{2\ell+2}(\xi)]\}\, \vec{\alpha}(\tau) \quad \ldots\ldots \quad (3.37)$$

Noticing that along the banks, $\phi'_{2\ell}(\xi_\ell) = 1$ and $\phi'_{2\ell-1}(\xi_\ell) = \phi'_{2\ell+1}(\xi_\ell) = \phi'_{2\ell+2}(\xi_\ell) = 0$, Eq. 3.37 can be reduced to

$$K_\eta^{(0)}\, \vec{\alpha}(\tau) = 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.38)$$

in which,

$$\vec{\alpha}(\tau) = [\alpha_{2k-1,2\ell}\ \alpha_{2k,2\ell}\ \alpha_{2k+1,2\ell}\ \alpha_{2k+2,2\ell}]^T.$$

At the downstream boundary, Eqs. 3.23 and 3.27 gives

$$\begin{aligned}\frac{\partial s}{\partial \eta}(\eta,\xi,\tau) = 0 \simeq \frac{\partial A}{\partial \eta}(\eta,\xi,\tau) \\ &= [K_\eta^{(1)} \otimes K_\xi^{(0)}]\, \vec{\alpha}(\tau) \\ &= \{[\phi'_{2k-1}(\eta)\ \phi'_{2k}(\eta)\ \phi'_{2k+1}(\eta)\ \phi'_{2k+2}(\eta)] \otimes \\ &\quad [\phi_{2\ell-1}(\xi)\ \phi_{2\ell}(\xi)\ \phi_{2\ell+1}(\xi)\ \phi_{2\ell+2}(\xi)]\}\, \vec{\alpha}(\tau) \quad \ldots\ldots \quad (3.39)\end{aligned}$$

Since along the downstream boundary, $\eta = \eta_k$, $\phi'_{2k}(\eta_k) = 1$, and $\phi'_{2k-1}(\eta_k) = \phi'_{2k+1}(\eta_k) = \phi'_{2k+2}(\eta_k) = 0$, Eq. 3.39 can be reduced to

$$K_\xi^{(0)}\, \vec{\alpha}(\tau) = 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.40)$$

in which,

$$\vec{\alpha}(\tau) = [\alpha_{2k,2\ell-1}(\tau)\ \alpha_{2k,2\ell}(\tau)\ \alpha_{2k,2\ell+1}(\tau)\ \alpha_{2k,2\ell+2}(\tau)]^T$$

At the two corner nodes, (η_k,ξ_ℓ), since $\phi_{2\ell-1}(\xi_\ell) = 1$ and $\phi_{2\ell}(\xi_\ell) = \phi_{2\ell+1}(\xi_\ell) = \phi_{2\ell+2}(\xi_\ell) = 0$, Eq. 3.40 can be further reduced to

$$\phi'_{2k}(\eta_k)\ \phi_{2\ell-1}(\xi_\ell)\ \alpha_{2k,2\ell-1}(\tau) = \alpha_{2k,2\ell-1}(\tau) = 0$$

Equations 3.35 , 3.38, and 3.40 are to be evaluated at the Gaussian points for each element along respective boundaries.

SOLUTION OF THE LINEAR SYSTEM OF EQUATIONS

The linear system of equations resulting from the collocation formulation can be written in the following form:

$$A\vec{\alpha} = \vec{b} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \qquad (3.41)$$

where A is an N x N non-symmetric matrix, and $\vec{\alpha}$ and $\vec{b}$ are vectors of length N. If N is small or A is dense, then the standard algorithm for solving this system is 'Gaussian elimination' with partial pivoting (provided that A is well conditioned). The simulation of two-dimensional mixing in natural rivers generally requires a large number of elements. This requirement together with the bicubic collocation formulation results in a large sparse system of equations. The standard technique becomes not feasible for this type of problem.

To overcome this problem, one can either apply one of the iterative techniques, such as the frontal technique, the successive over relaxation, or the Gauss-Siedel iterative method; or use a direct method, such as Gaussian elimination or matrix inversion (4). Since an iterative method cannot be applied to a system of equations with zero entries along the diagonal, direct methods are preferred if care is taken for the large number of zeros that exist in the system. Recently, several methods have been developed using sparse Gaussian elimination to solve systems like Eq. 3.41 (9,10,14).

Herein, the algorith of Sherman (38), which is considered as a refinement of the other algorithms, for solving linear systems and performing Gaussian elimination with column interchanges will be used. Briefly, this algorithm can be summarized as follows,

(a) The linear system of equations, Eq. 3.41 is used to obtain a factorization of the form

$$AQ = LU \qquad \ldots\ldots\ldots\ldots\ldots\ldots \qquad (3.42)$$

where L is lower triangular, U is upper triangular and Q is a permutation matrix corresponding to the column interchanges.

(b) Once this factorization has been obtained, one can find $\vec{\alpha}$ by solving

$$L\vec{y} = \vec{b}$$

and $\qquad \ldots\ldots\ldots\ldots\ldots\ldots \qquad (3.43)$

$$UQ^T\vec{\alpha} = \vec{y}$$

The data structure for the sparse matrix that is both compact and easily accessible is the important feature of this algorithm. All calculations are carried out in three one-dimensional arrays, two of them list the column indices and numerical value of the nonzero matrix entries, the third array is a set of row pointers.

The algorithm is "stable" numerically since the application of standard Gaussian elimination with row interchanges is stable. The number of operations required by this algorithm depends strongly on the number of nonzero entries of sparse matrix A. Concerning computer time, this algorithm works quite efficiently and as far as the authors are aware are better than any other algorithms available (38).

ERROR ANALYSIS

The objective of any numerical technique is generally to obtain sufficiently accurate approximations with minimum effort. The efficiency of an approximation method with respect to its use in computation is measured by the "Truncation Error," (defined as the amount by which the exact solution fails to satisfy the difference equation) and is denoted by T.

To analyze the error encountered when applying the collocation finite element method with rectangular Hermite bicubic elements to the nonsteady two-dimensional convection-diffusion equation, consider first Eq. 3.29 with $\theta = \frac{1}{2}$,

$$[H]\ \{(\frac{\partial\alpha(\tau)}{\partial\tau})_{n+1} + (\frac{\partial\alpha(\tau)}{\partial\tau})_{n}\} = [K]\ \{(\alpha(\tau))_{n+1} + (\alpha(\tau))_{n}\} + 2f_5(\eta,\xi) \quad . \ . \ (3.44)$$

If a forward or backward difference scheme is used, the result will produce a local truncation error of order $O(\Delta t)$, where Δt is the time increment. To devise a procedure with local truncation error of order $O(\Delta\tau^2)$, Smith, et. al. (39), suggested to use a forward and backward difference in time which gives the most accurate result for this type of problem. This can be done by using the Taylor series in time which yields

$$(\frac{\partial\alpha(\tau)}{\partial\tau})_{(n+1/2)} = \frac{(\alpha(\tau)_{n+1} - (\alpha(\tau))_n}{\Delta\tau} - \frac{1}{48}\ (\Delta\tau)^2\ [\frac{\partial^3\alpha(\tau_0)}{\partial t^3} + \frac{\partial^3\alpha(\tau_1)}{\partial t^3}] \quad . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \quad (3.45)$$

where $\tau_0 \varepsilon\ (\tau,\tau + \frac{\Delta\tau}{2})$, $\tau_1 \varepsilon\ (\tau - \frac{\Delta\tau}{2}, \tau)$.

This method is a second order method and the local truncation error is of order $O(\Delta\tau)^2$ i.e.,

$$T = \frac{1}{48}\ (\Delta\tau)^2\ (\frac{\partial^3\alpha(\tau_0)}{\partial t^3} + \frac{\partial^3\alpha(\tau_1)}{\partial t^3}) \quad . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \quad (3.46)$$

provided the solution of the differential equation satisfies the usual differentiability condition. The error bound can be written as

$$\text{ERROR BOUND} = |T| \leq \frac{1}{24} \Delta\tau^2 M_1 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.47)$$

where $M_1 = \max \left|\frac{\partial^3 \alpha(\overset{o}{\tau})}{\partial t^3}\right|$; $\overset{o}{\tau} \in [\tau - \frac{\Delta\tau}{2} , \tau + \frac{\Delta\tau}{2}]$.

The error encountered when applying a finite element method in space can be analyzed by using Taylor's theorem in two variables. In general, suppose that $C(\eta,\xi)$ is the exact solution for the problem 'P' and all of its partial derivatives of order less than or equal to (m+1) are continuous in the domain Ω.

$$\Omega = \{(\eta,\xi)|a \leq \eta \leq b,\ c \leq \xi \leq d\}$$

and let $(\eta_o,\xi_o) \in \Omega$. Then for every point $(\eta,\xi) \in \Omega$ there exists a point $(\overset{o}{\eta},\overset{o}{\xi}) \in \Omega$ with

$$C(\eta,\xi) = P_m(\eta,\xi) + R_{m+1}(\eta,\xi) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.48)$$

where

$$P_m(\eta,\xi) = C(\eta_o,\xi_o) + [(\eta-\eta_o) \frac{\partial c}{\partial \eta} (\eta_o,\xi_o) + (\xi-\xi_o) \frac{\partial c}{\partial \xi} (\eta_o,\xi_o)]$$

$$+ [\frac{(\eta-\eta_o)^2}{2} \frac{\partial^2 C}{\partial \eta^2} (\eta_o,\xi_o) + (\eta-\eta_o)\ (\xi-\xi_o) \frac{\partial^2 C}{\partial \xi} (\eta_o,\xi_o) + \frac{(\xi-\xi_o)^2}{2} \frac{\partial^2 C}{\partial \xi^2} (\eta_o,\xi_o)]$$

$$+ \ldots\ \ldots + [\frac{1}{m!} \sum_{j=0}^{m} \binom{m}{j} (\eta-\eta_o)^{m-j} (\xi-\xi_o)^j \frac{\partial^m C(\eta_o \xi_o)}{\partial \eta^{m-j} \partial \xi^j}] \quad \ldots \quad (3.49)$$

and

and

$$R_{m+1}(\eta,\xi) = \frac{1}{(m+1)!} \sum_{j=0}^{m+1} \binom{m+1}{j} (\eta-\eta_o)^{m+1-j} (\xi-\xi_o)^j \frac{\partial^{m+1} C(\overset{o}{\eta},\overset{o}{\xi})}{\partial\eta^{m+1-j}\partial\xi^j} \quad \ldots\ldots \quad (3.50)$$

P_m is a Taylor polynomial of degree m in two variables for the function C about (η_o,ξ_o) and R_{m+1} (η,ξ) is the truncation error associated with $P_m(\eta,\xi)$.

Using the collocation method with Hermite bicubic elements the remainder term or the truncation error can be written as

$$R_4(\eta,\xi) = \frac{1}{4!} \sum_{j=0}^{4} \binom{4}{j} (\Delta\eta)^{4-j} (\Delta\xi)^j \frac{\partial^4 C(\overset{o}{\eta},\overset{o}{\xi})}{\partial\eta^{4-j}\partial\xi^j} \quad \ldots\ldots\ldots\ldots \quad (3.51)$$

where, $\Delta\eta = (\eta-\eta_o)$ and $\Delta\xi = (\xi-\xi_o)$.

Assuming that the mesh size (element length) in the η-direction is the same as that in the ξ-direction, Eq. 3.51 becomes,

$$R_4(\eta,\xi) = (\Delta\eta)^4 \frac{1}{4!} \sum_{j=0}^{4} \binom{4}{j} \frac{\partial^4 C(\overset{o}{\eta},\overset{o}{\xi})}{\partial\eta^{4-j}\partial\xi^j} \quad \ldots\ldots\ldots\ldots\ldots \quad (3.52)$$

which means that the error is of order $O(\Delta\eta)^4$, provided that all the previous conditions for smoothness of the function $C(\eta,\xi)$ is atisfied. The error bound can be written as

$$|R_4(\eta,\xi)| \leq \frac{1}{24} (\Delta\eta)^4 M_2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.53)$$

in which,

$$M_2 = \max \left|\left\{\frac{\partial^4 C(\eta_1,\xi_1)}{\partial\eta^4}, \frac{\partial^4 C(\eta_1,\xi_1)}{\partial\eta^3 \eta\xi}, \frac{\partial^4 C(\eta_1,\xi_1)}{\partial\eta^2\partial\xi^2}, \frac{\partial^4 C(\eta_1,\xi_1)}{\partial\eta\partial\xi^3}\right.\right.$$

$$\left.\left.\frac{\partial^4 C(\eta_1,\xi_1)}{\partial\xi^4}\right\}\right|$$

and $(\eta_1,\xi_1) \in \Omega \ni \eta_1 \in [\eta,\Delta\eta,\eta+\Delta\eta]$, $\xi_1 \in [\xi-\Delta\eta,\xi+\Delta\eta]$

Let E_T be the error when applying finite differencing in time and finite element in space. From Eqs. 3.51 and 3.52 one can then write,

$$\max_{\eta,\xi,\tau} |E_T| = \max_{\eta,\xi,\tau} | T + R_4(\eta,\xi)| \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.54)$$

$$\leq C_1(\Delta\tau)^2 + C_2\ (\Delta\eta)^4 \leq C_3[\Delta\tau^2 + \Delta\eta^4]$$

where C_1 and C_2 are independent of $\Delta\tau$ and $\Delta\eta$, and C_3 is the $\max\{C_1,C_2\}$.

Equation 3.54 shows that the truncation error when using the collocation method with rectangular Hermite bicubic elements is of order $O[\Delta\tau^2 + \Delta\eta^4]$. To make use of the higher order of accuracy, $(\Delta\tau)$ should be chosen in a way such that,

$$\Delta\tau \leq C_4(\Delta\eta)^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.55)$$

where C_4 is a constant, independent of $\Delta\tau$ and $\Delta\eta$. Consequently, Eq. 3.54 becomes,

$$\max_{\eta,\tau} |E_T| \leq C_5(\Delta\eta)^4 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.56)$$

in which, C_5 is independent of $\Delta\tau$ and $\Delta\eta$. Eq. 3.56 shows that the order of convergence is equal to 4.

Numerically, the following formula can be used to check for the order of convergence K.

$$K \simeq \ell n \left(E_{T_{\Delta\eta}} / E_{T_{\Delta\eta/2}}\right) / \ell n 2 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.57)$$

where $E_{T_{\Delta\eta}}$, $E_{T_{\Delta\eta/2}}$ are errors for element sizes $\Delta\eta$ and $\Delta\eta/2$ at a same $\Delta\tau$ value.

CHAPTER IV

MODEL VERIFICATION AND APPLICATION

In this chapter the numerical scheme developed in Chapter III is verified against exact analytical solutions. The order of convergence of the scheme is verified numerically. The scheme is also used to numerically simulate mixing in a reach of the Missouri River and compared with a steady state analytical solution and field data.

MODEL VERIFICATION

Two two-dimensional initial boundary value problems with analytical solutions are used to verify the numerical scheme. These two boundary value problems and their analytical solutions are:

PROBLEM I

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} - \frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f_1 \quad \dots \dots \dots \dots (4.1)$$

in which,

$$f_1 = -2t[x^2(\frac{x}{3} - \frac{1}{2}) + y^2(\frac{y}{3} - \frac{1}{2}) + t^2(x^2 - 3x + 2 - 2y)].$$

$$u(0,y,t) = t^2(\frac{y^2}{2} - \frac{y^3}{3}) \quad \dots \dots \dots \dots (4.2)$$

$$\frac{\partial u}{\partial x}(1,y,t) = 0 \quad \dots \dots \dots \dots (4.3)$$

$$\frac{\partial u}{\partial y}(x,0,t) = 0 \quad \dots \dots \dots \dots (4.4)$$

$$\frac{\partial u}{\partial y}(x,1,t) = 0 \quad \dots \dots \dots \dots (4.5)$$

and $u(x,y,0) = 0$, $0 \leq x \leq 1$ and $0 \leq y \leq 1$. . . . . (4.6)

The exact solution is,

$$u(x,y,t) = t^2(\frac{x^2}{2} + \frac{y^2}{2} - \frac{x^3}{3} - \frac{y^3}{3}) \quad \dots \dots \dots \dots (4.7)$$

PROBLEM II

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} - \frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f_2 \quad \ldots\ldots \quad (4.8)$$

in which,

$$f_2 = -[x^2(\frac{x}{3} - \frac{1}{2}) + y^2(\frac{y}{3} - \frac{1}{2}) + t\ (x^2 - 3x + 2 - 2y)].$$

$$u(0,y,t) = t(\frac{y^2}{2} - \frac{y^3}{3}) \quad \ldots\ldots \quad (4.9)$$

$$\frac{\partial u}{\partial x}(1,y,t) = 0 \quad \ldots\ldots \quad (4.10)$$

$$\frac{\partial u}{\partial y}(x,0,t) = 0 \quad \ldots\ldots \quad (4.11)$$

$$\frac{\partial u}{\partial y}(x,1,t) = 0 \quad \ldots\ldots \quad (4.12)$$

and $u(x,y,0) = 0$; $0 \leq x \leq 1$ and $0 \leq y \leq 1$. . . . . (4.13)

The exact solution is,

$$u(x,y,t) = t(\frac{x^2}{2} + \frac{y^2}{2} - \frac{x^3}{3} - \frac{y^3}{3}) \quad \ldots\ldots \quad (4.14)$$

To test the degree of accuracy and the rate of convergence, the model was run for different values of time increments (Δt) with rectangular elements. Table 4-1 shows the calcualted order of convergence for Problem I for various element sizes and time steps. Calculated values of K using Eq. 3.57, as shown in Table 4-1, are approximately equal to 4.0, which verifies the conclusion obtained from Eq. 3.56.

Numerical solutions for problem II for various element sizes and time steps are also obtained and compared with analytical solutions. The maximum absolute error is of order 10^{-7}.

TABLE 4-1
ORDER OF CONVERGENCE (K) FOR COLLOCATION METHOD

$\Delta t/(\Delta x)^2 = 2$				$\Delta t/(\Delta x)^2 = 4$			
Δt	Δx	E_T	K	Δt	Δx	E_T	K
0.5	0.5	2.13×10^{-2}	--	--	--	--	--
0.125	0.25	1.33×10^{-3}	4.01	0.4	0.33	1.20×10^{-2}	--
0.03	0.125	8.29×10^{-5}	4.02	0.1	0.165	7.61×10^{-4}	3.99

APPLICATION OF THE NUMERICAL MODEL

A field study of the mixing characteristics of Missouri River by Yotsukuna, Fischer, and Sayre (47) for a six-mile reach below Blair, Nebraska, shown in Fig. 4.1. This particular reach has two mild alternating curves. At the time of field test, the river discharge was 34,100 cfs, the average depth was 9 feet, and the width ranged from 500 to 700 feet. The average velocity was 5.7 fps, and the shear velocity was estimated to be 0.24 fps. Detailed depth and velocity data were obtained at two cross sections as indicated in Fig. 4.1. The field test consisted of continuously injecting a tracer into the river and measuring the steady state concentration at downstream cross sections. Yotsukura and Cobb (46) has obtained the steady state concentration distribution by using a two-dimensional analytical solution by assuming variations of E_z over a cross section are negligible.

The collocation finite-element solution was applied to the Missouri River for a reach extending from 8730 ft station to the 12,000 ft station. The distribution of the velocity and the depth at different points in the transverse coordinate for the first two cross sections are listed in Table 4.2. The corresponding values in x-q_c coordinates are determined by the subroutine APPR. The velocity and depth were taken to be equal to zero at the channel boundaries. Eqs. 2.23 and 2.24 are used to calculate values of the longitudinal and transverse dispersion coefficients E_x and E_z with $\alpha = 6.0$ and $\beta = 0.6$. Metric coefficients m_x and m_z were taken to be equal to unity. A value of 5280 ft for the reference length ℓ_o is used. The upstream boundary concentration used in the sample calculation increases linearly from zero to a steady state boundary concentration distribution, over a period of $10\Delta t$. The steady state concentration distribution at the

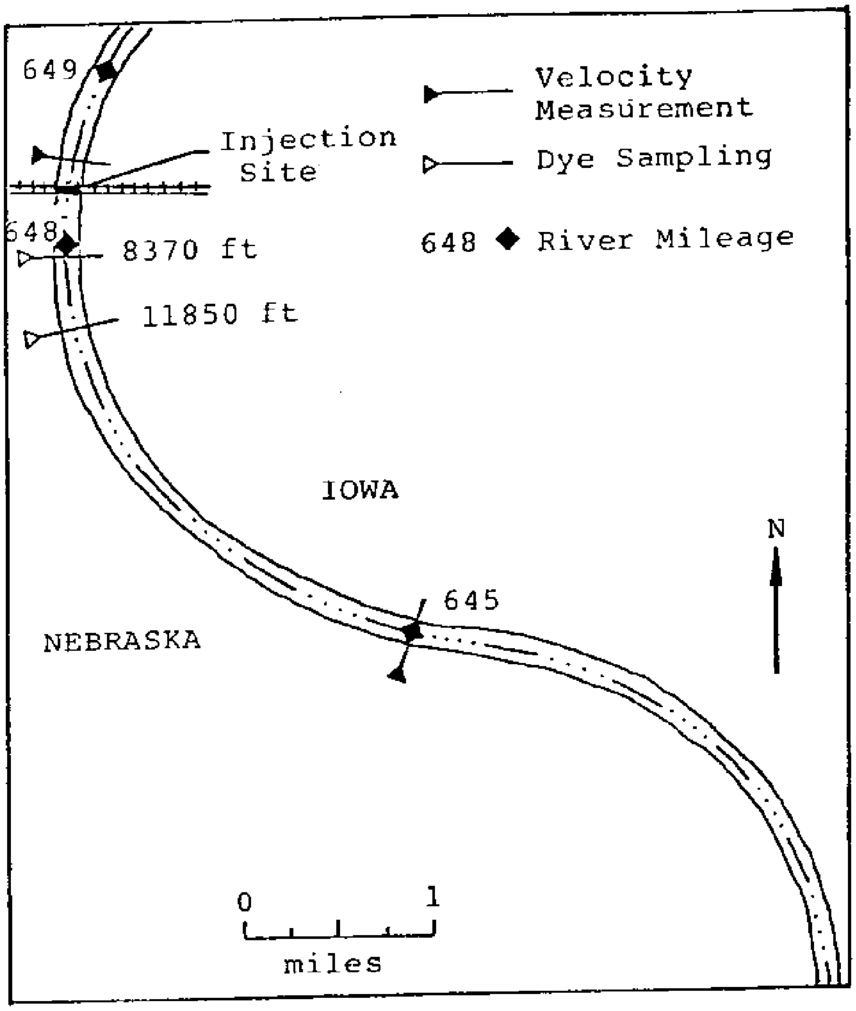


Figure 4.1 Missouri River Near Blair, Nebraska

TABLE 4-2
TRANSVERSE DISTRIBUTIONS OF VELOCITY AND DEPTH
AT THE FIRST TWO CROSS SECTIONS IN MISSOURI RIVER.

POINT NO.	x = 8730 ft.			x = 11,850 ft.		
	(Z) COORD.	(V) VELOCITY	(h) DEPTH	(Z) COORD.	(V) VELOCITY	(h) DEPTH
1	0	0.0	0.0	0	0.0	0.0
2	25	3.36	3.50	25	3.36	3.36
3	60	6.88	10.25	60	6.88	6.88
4	90	8.27	13.50	90	8.27	8.27
5	120	8.27	13.50	120	8.27	8.27
6	155	8.07	13.00	155	8.07	8.07
7	185	7.86	12.50	185	7.86	7.86
8	225	7.54	11.75	225	7.54	7.54
9	270	6.99	10.50	270	6.99	6.99
10	320	6.07	8.50	320	6.07	6.07
11	385	4.82	6.00	385	4.82	4.82
12	415	4.82	6.00	4.15	4.82	4.82
13	435	4.82	6.00	435	4.82	4.82
14	490	3.98	4.50	490	3.98	3.98
15	525	3.52	3.75	525	3.52	3.52
16	560	3.20	3.25	560	3.20	3.20
17	600	0.0	0.0	600	0.0	0.0

upstream boundary was adapted from the analytical concentration distribution given by Yotsukura and Cobb (36) for the station corresponding to 8730 ft. This concentration distribution is illustrated in Figure 4.2.

The model was run for different values of time increments with different combinations of N_x and N_y. Numerical solutions for the concentration distribution were obtained until the steady state condition was reached, for all cases. Table 4.3 summarized calculated steady state concentration distribution at x = 11850 ft. for $q_c^* = 0.2, 0.4, 0.6$ and 0.8.

According to the analysis given in Chapter III, for an appropriately chosen value of $\Delta\tau$, the maximum error can be estimated by

$$\max|E_T| = C\cdot|\max\{\Delta\eta\Delta\xi^2,\ \Delta\eta^2\Delta\xi^2,\ \Delta\eta^3\Delta\xi,\ \Delta\eta^4,\ \Delta\xi^4\}|$$

Based on this, the error for the case $N_x = N_y = 10$, which is the most accurate case, is of $0(10^{-4})$. For other combinations of N_x and N_y in Table 4-3, the same accuracy can be obtained, except for cases with $N_x = 3$ and cases with $N_y = 3$. Errors for these cases are of $0(10^{-3})$ and $0(10^{-2})$, respectively. Numerical results presented in Table 4-3 generally agree with the error analysis. Cases with $N_y = 3$ are in general least accurate. This is consistent with the error analysis. An additional reason responsible for the inaccuracy in cases with $N_y = 3$ is the linear interpolation used to approximate the transverse concentration distribution at the upstream boundary. It is of interest to note that the use of larger time steps has relative small effect on the accuracy of the solution.

Table 4-4 represents the number of time steps and the computer time required to reach the steady state. This result shows that the computing time required for each time step is the same for a given combination of N_x and N_y values, even though Δt values are different. Computing time required for each time step varies from 2.4 sec. for $N_x = 3$, $N_y = 10$ to 15 sec. for

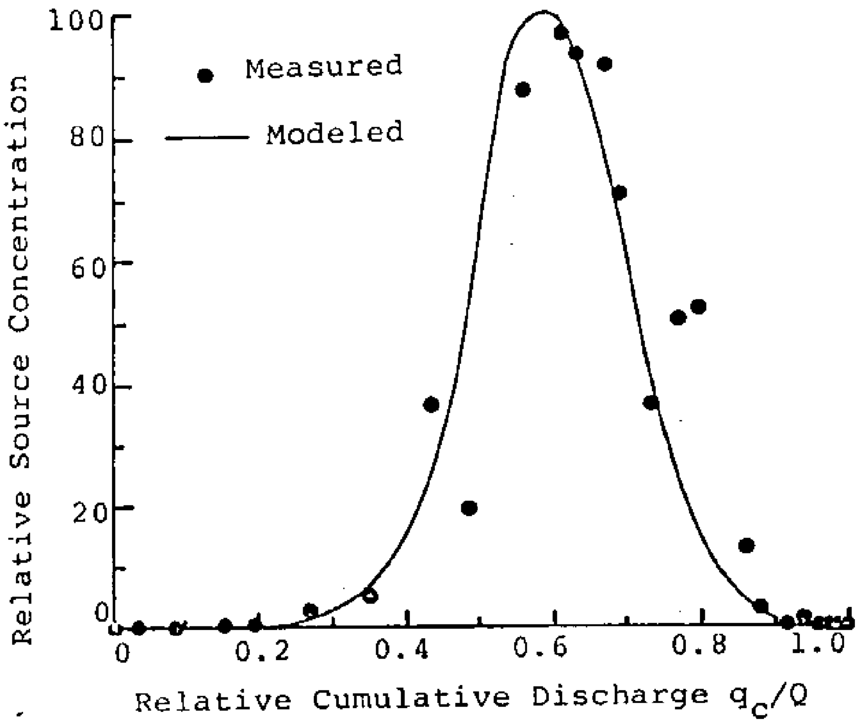


Figure 4.2. Source Concentration Distribution

TABLE 4-3.

Transverse distribution of relative concentration at cross section 2 (11,875 ft.) at different values of relative comulative discharge (q_c^*) for the different values of N_x, N_y and Δt at steady state.

[min η=0, min ξ=0.0, max η=0.619 and max ξ=1.0].

Δt (sec.)	N_x/N_y	$s(q_c^*=0.2)$	$s(q_c^*=0.4)$	$s(q_c^*=0.6)$	$s(q_c^*=0.8)$
10	3/10	0.0283	0.2845	0.7167	0.1957
	5/10	0.0284	0.2897	0.7405	0.1850
	8/10	0.0285	0.2905	0.7362	0.1853
	10/3	0.0120	0.2856	0.7394	0.2319
	10/8	0.0275	0.2865	0.7255	0.2103
	10/10	0.0283	0.2898	0.7371	0.1856
25	3/10	0.0288	0.2897	0.7328	0.1877
	5/10	0.0292	0.2921	0.7350	0.1851
	8/10	0.0286	0.2906	0.7359	0.1855
	10/3	0.0120	0.2868	0.7420	0.2339
	10/8	0.0275	0.2868	0.7254	0.2099
	10/10	0.0284	0.2901	0.7370	0.1854
50	3/10	0.0291	0.2910	0.7334	0.1862
	5/10	0.0289	0.2910	0.7345	0.1860
	8/10	0.0285	0.2902	0.7359	0.1857
	10/3	0.0120	0.2868	0.7416	0.2343
	10/8	0.0275	0.2868	0.7260	0.2098
	10/10	0.0284	0.2901	0.7376	0.1856
100	3/10	0.0292	0.2914	0.7334	0.1862
	5/10	0.0289	0.2909	0.7345	0.1860
	8/10	0.0285	0.2903	0.7358	0.1857
	10/3	0.0118	0.2867	0.7316	0.2344
	10/8	0.0275	0.2868	0.7251	0.2099
	10/10	0.0284	0.2900	0.7366	0.1856
200	3/10	0.0292	0.2914	0.7344	0.1862
	5/10	0.0289	0.2910	0.7345	0.1860
	8/10	0.0286	0.2906	0.7384	0.1857
	10/3	0.0121	0.2864	0.7405	0.1823
	10/8	0.0275	0.2868	0.7257	0.2099
	10/10	0.0284	0.2898	0.7362	0.1855

$N_x = 10$, $N_y = 10$. Table 4-4 also provides some information for selecting N_x, N_y and Δt values with which a required degree of accuracy can be acheived with a minimal computing time. By comparing the computing time required by the present model with that of the finite difference model developed by Harden and Shen (15), one can see that for $\Delta t = 10$ seconds, the computing time required by the finite-difference model is about fifteen times longer. This is due to the fact that the order of the accuracy of the finite-difference model is $O(\Delta\tau, \Delta\eta^2, \Delta\xi^2)$.

Figure 4.3 shows the transverse concentration distribution at the 11,875 ft station after steady state conditions were reached for different values of Δt, N_x and N_y. Those values are compared with values obtained in the previous studies (47). The various figures indicate that the predicted values are in an excellent agreement with those of the analytical model (46), the finite-difference solution and the measured data.

Figure 4.4 shows the concentration profiles along the stream tube boundary corresponding to $q_c^* = 0.6$ at various time levels. The "overshoot" near the peak of the dispersing front presented in the finite difference solutions did not appear in the present solution.

TABLE 4-4.

The CPU time and the number of time steps required to reach the steady state.

Δt (sec.) \ N_x/N_y		3/10	5/10	8/10	10/3	10/8	10/10
10	CPU Time (Min.)	3.4	6.4	14.4	4.8	17.0	18.2
	No. of Time Steps	85	80	80	80	75	70
25	CPU Time (Min.)	2.4	3.5	8.5	3.6	9.5	12.5
	No. of Time Steps	60	50	50	50	50	50
50	CPU Time (Min.)	2.0	3.15	8.0	5.1	9.5	10.0
	No. of Time Steps	50	45	50	50	45	40
100	CPU Time (Min.)	2.4	4.2	7.8	3.6	11.4	13.75
	No. of Time Steps	60	60	60	60	60	55
200	CPU Time (Min.)	3.4	5.95	14.45	5.1	15.2	21.25
	No. of Time Steps	85	85	85	80	80	65

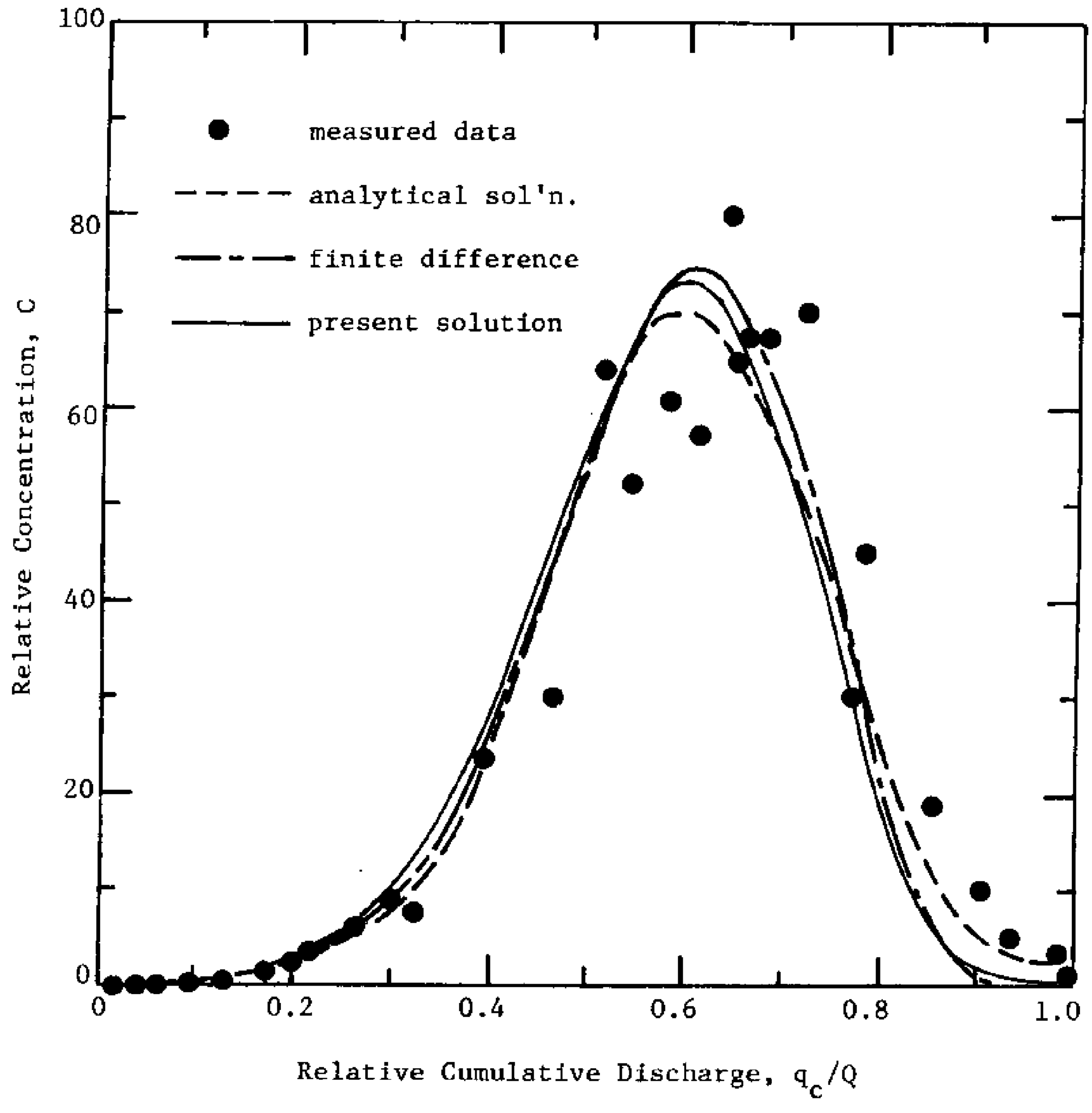


Figure 4.3. Comparison of Steady State Solutions at x = 11875 ft.

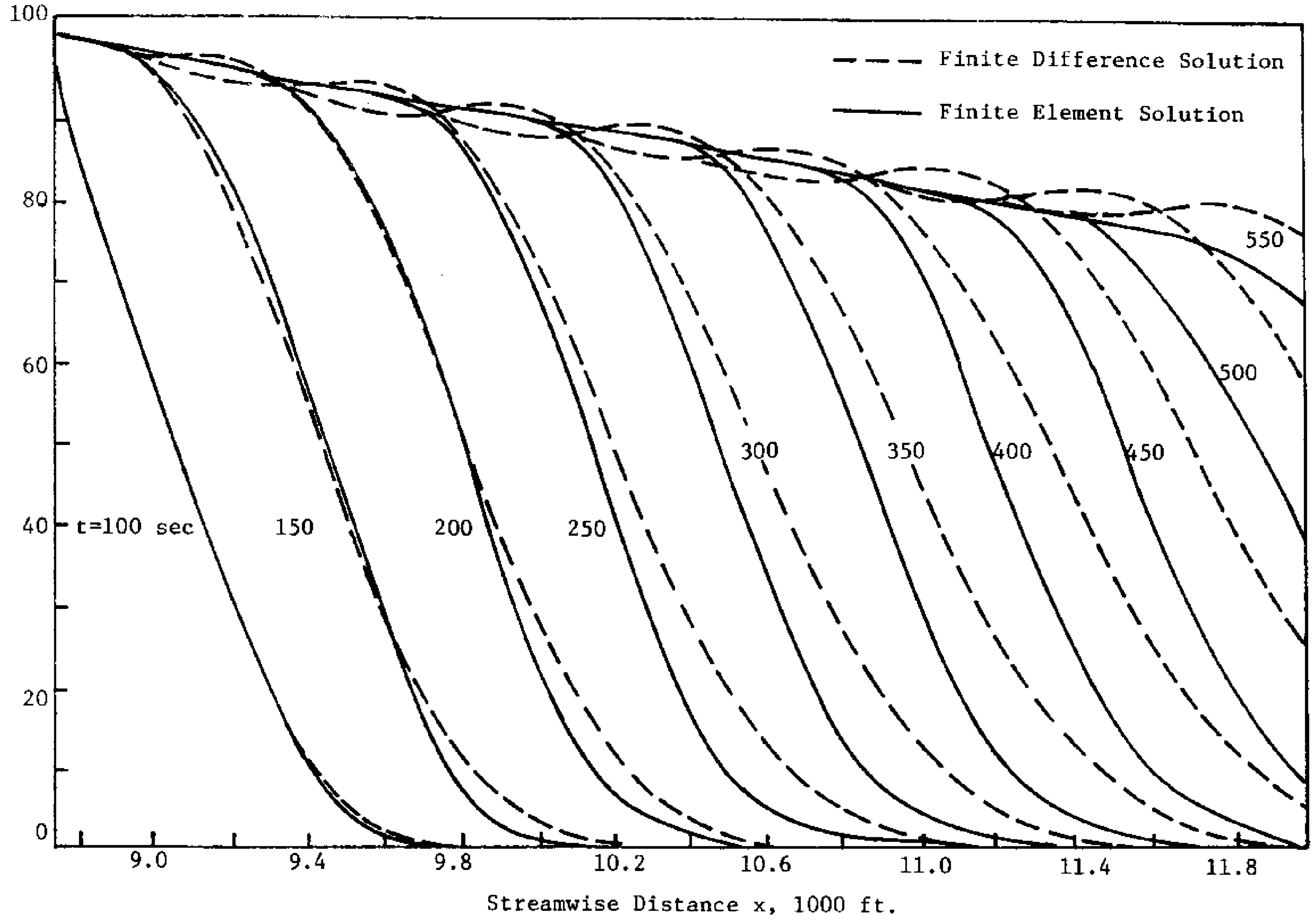


Figure 4.4. Comparison of Concentration Profiles Along the Stream Tube Boundary $q_c/Q = 0.6$

CHAPTER V

CONCLUSIONS

A collocation finite-element model using rectangular bi-cubic elements to solve the transient mixing equation in a natural coordinate system for rivers is developed. This numerical model can be used to predict transverse concentration distributions in a natural river when the concentration becomes reasonably uniform over the depth. The model is compared with an existing finite-diffrence model based on a combined implicit/explicit scheme and is found to be more efficient and stable. The order of convergence of the collocation finite-element scheme is analyzed to provide a guideline for choosing appropriate element sizes and time steps. The efficiency of the model can be further improved when more efficient techniques for solving systems of linear equations become available.

REFERENCES

1. Banks, R.B., "Distribution of BOD and DO in Rivers and Lakes," Journal of the Environmental Engineering Division, ASCE, Vol. 102, No. EE2, Proc. Paper 12028, Apr. 1976, pp. 265-280.

2. Brian, P.L.T., "A Finite Difference Method of High Order Accuracy for the Solution of Three-Dimensional Transient Heat Conduction Problem," Journal A.I.Ch.E., Vol. 7, No. 3, September 1961, pp. 367-370.

3. Briggs, D.A., Mudson, D.O., "Analytical Models for One and Two Layer Systems in Rectangular Basin," Ralph M. Parsons Laboratory for Water Resources and Hydrodynamics, M.I.T., Department of Civil Engineering, Report No. 172, October 1973.

4. Burden, R.A., Faires, J.D., Reynolds, A.C., "Numerical Analysis," Prindle, Weber & Schmidt, Incorp., 1978, 579 pp.

5. Carlson, R.E. and Hall, C.A., "Ritz Approximation to Two-Dimensional Boundary Value Problems," Numer. Math. 18, pp. 171-181, 1971.

6. Chang, Y.C., "Lateral Mixing in Meandering Channels," Ph.D. Dissertation, Department of Mechanics and Hydraulics, University of Iowa, Iowa City, 1971.

7. Chung, T.J., "Finite Element Analysis in Fluid Dynamics," McGraw-Hill International Book Company, 1978.

8. Connor, J.J., "Finite Element Modeling of Two-Dimensional Hydrodynamic Circulation," Ralph M. Parsons Laboratory for Water Resources, M.I.T., Department of Civil Engineering, Report No. 172, October 1973.

9. Curtis, A.R. and Reid, J.K., "The Solution of Large Sparse Unsymmetric Systems of Linear Equations," Information Processing, 71(1971), 1240-1245.

10. Eisental, S.C., Schultz, M.H. and Sherman, A.H., "Considerations in the Design of Software for Sparse Gaussian Elimination," Proceedings of the Symposium on Sparse Matrix Computations at Argonne National Laboratory, Sept. 1975, pp. 263-273.

11. Fischer, H.B., "Transverse Mixing in a Sand Bed Channel," U.S. Geological Survey Professional Paper, 575-D, pp. D267-272, 1967.

12. Fix, G.J., "An Analysis of the Finite Element Method," Prentice Hall, Inc., England Cliffs, N.J., 1973.

13. Fukuoka, S. and Sayre, W.W., "Longitudinal Dispersion in Sinuous Channels," Journal of the Hydraulics Division, ASCE, Vol. 99, No. HY1, 1973, pp. 195-217.

14. Gustavons, F.G., "Two Fast Algorithms for Sparse Matrices, Multiplication and Permuted Transposition," ACM Transaction on Mathematical Software, Vol. 4, No. 3, September 1978, pp. 250-270.

15. Harden, O.T., Shen, H.T., "Numerical Simulation of Mixing in Natural Rivers," Journal of the Hydraulics Division, ASCE, Vol. 105, No. HY4, April 1979.

16. Herrmann, L.R., Scott, V.H. and Guymon, L.G., "A General Numerical Solution of Two-Dimensional Diffusion-Convection Equation by the Finite Element Method," Water Resources Research, Vol. 6, No. 6, December 1970.

17. Houstis, E.N., Lynch, R.E., Rice, J.R., and Papatheodorou, T.S., "Evaluation of Numerical Methods for Elliptic Partial Differential Equations," Journal of Computational Physics, Vol. 27, No. 3, June 1978, pp. 323-350.

18. Irons, B.M., "A Frontal Solution Program for Finite Element Analysis," International Journal for Numerical Methods in Engineering, Vol. 2, 5-32, 1972.

19. Jirka, G.H., Abraham, G., Harlemann, D.R.F., "An Assessment of Techniques for Hydrothermal Prediction," Report No. 203, R.M. Parsons Lab., Massachusetts Institute of Technology, Cambridge, Mass., July 1975, 403 pp.

20. Jobson, H.E., "Vertical Mass Transfer in Open Channel Flow," Ph.D. Dissertation, Department of Civil Engineering, Colorado State University, Fort Collins, Colorado, December 1968.

21. Lawson, D.W., "Improvement in the Finite Difference Solution of Two-Dimensional Dispersion Problems," Water Resources Research, Vol. 7, No. 3, June 1971.

22. Leimkuhler, W.F., "A Two-Dimensional Finite Element Dispersion Model," Thesis submitted in partial fulfillment of the requirements for the degree of Civil Engineering at the Massachusetts Institute of Technology, Sept. 1974, 80 pp.

23. Loziuk, L.A., Anderson, J.C. and Bleytschko, T., "Transient Hydrothermal Analysis of Small Lakes," Journal of the Power Division, ASCE, Vol. 99, No. PO2, Proc. Paper 10150, Nov. 1973, pp. 349-364.

24. Norrie, D.H. and Vries, G.D., "The Finite Element Method," Academic Press, New York, 1973.

25. Papaspyropoulos, G.T., "A Collocation Finite-Element Solution of the Convection-Diffusion Equation," M.S. Thesis, Department of Civil and Environmental Engineering, Clarkson College of Technology, Mar. 1981.

26. Peaceman, D.W. and Rachford, H.H., Jr., "Numerical Calculations of Multidimensional Miscrible Displacement," Journal of the Society of Petroleum Engineers, Vol. 2, No. 4, Dec. 1962, pp. 327-339.

27. Peaceman, D.W. and Rachford, H.H., Jr., "The Numerical Solution of Parabolic and Elliptic Differential Equations," Journal of the Society of Industrial and Applied Mathematics, Vol. 3, No. 1, March 1955, pp. 28-41.

28. Price, H.S., Cavendish, J.C. and Varga, R.S., "Numerical Methods of High-Order Accuracy for Diffusion-Convection Equation," Soc. Petrol. Eng. J., 243, 1968, pp. 293-303.

29. Rastogi, A.K. and Rodi, W., "Prediction of Heat and Mass Transfer in Open Channels," Journal of Hydraulics Division, ASCE, Vol. 104, No. HY3, Proc. Paper 13639, March 1978, pp. 397-420.

30. Remson, I., et. al., "Numerical Methods in Subsurface Hydrology," Wiley-Interscience, New York, 1971, pp. 208-219.

31. Roberts, K.V. and Weiss, N.O., "Convective Difference Schemes," Math. Comp. 20, 1966, pp. 272-297.

32. Rose, D.J. and Bunch, J.R., "The Role of Partitioning in Numerical Solution of Sparse Systems," Plenum Press, New York, 1972, pp. 177-187.

33. Sayre, W.W. and Yeh, T.P., "Transverse Mixing Characteristics of the Missouri River Downstream from the Cooper Nuclear Station," IIHR Report 145, Iowa Institute of Hydraulics Research, Unaiversity of Iowa, Iowa City, Iowa, 1973.

34. Shamir, U. and Harleman, D.R.F., "Numerical and Analytical Solutions of Dispersion Problems in Homogeneous and Layered Aquifers," Hydrodynamics Laboratory Report 89, M.I.T., Department of Civil Engineering, May 1966.

35. Shapiro, A., Pinder, G.E., "New Collocation Method for Solution of the Convection-Dominated Transport Equation," Water Resources Research, Vol. 15, No. 5, October 1979.

36. Shen, H.T., "Transient Mixing in River Channels," Journal of the Environmental Engineering Division, ASCE, Vol. 104, No. EE2, June 1978.

37. Shen, H.T., Ackermann, N.L., "Wintertime Flow Distribution in River Channels," Journal of the Hydraulics Division, ASCE, Vol. 106, No. HY5, Proc. Paper 15427, May 1980, pp. 805-817.

38. Sherman, A.H., "Algorithms of Sparse Gaussian Elimination with Partial Pivoting," ACM Transactions on Mathematical Software, Vol. 4, No. 4, December 1978, pp. 330-338.

39. Smith, I.M., Farraday, R.V., and O'Connor, B.A., "Rayleigh-Ritz and Galerkin Finite Element for Diffusion-Convection Problems," Water Resources Research, Vol. 9, No. 3, June 1973, pp. 593-605.

40. Spalding, D.B., Stephenson, P.L. and Taylor, R.G., "A Novel Finite Difference Formulation for Differential Expressions Involving Both First and Second Derivatives," Int. J. Num. Meth. Engng., 4, pp. 551-559, 1972.

41. Stone, H.L. and Brian, P.L.T., "Numerical Solutions of Convective Transport Prolbems," Journal of A.I.Ch.E., Vol. 9, No. 5, Sept. 1963, pp. 681-688.

42. Sutron Corporation, "Mathermatical Model of the Lateral and Longitudinal Mixing Processes in Open Channels," MRD Sediment Series No. 16, U.S. Army Engineering Division, Missouri River, Corps of Engineers, Omaha, Nebraska, May 1979.

43. U.S. Nuclear Regulatory Commission, "Estimating Aquatic Dispersion of Effluents from Accidental and Routine Reactor Releases for the Purpose of Implementing, Appendix I," Regulatory Guide 1.113, Washington, D.C., April 1977, 55 pp.

44. Williams, B.J., and Hinwood, J.B., "Two-Dimensional Mathematical Water Quality Model," Journal of the Environmental Engineering Division, ASCE, Vol. 102, No. EE1, Proc. Paper 11947, Feb. 1976, pp. 149-163.

45. Yotsukura, N. and Sayre, W.W., "Transverse Mixing in Natural Channels," Water Resources Research, Vol. 12, No. 4, August 1976, pp. 695-704 and Vol. 13, No. 2, April 1977, pp. 495-497.

46. Yotsukura, N. and Cobb, E.D., "Transverse Diffusion of Solutes in Natural Streams," U.S. Geological Survey Professional Paper, 582-C, 1972.

47. Yotsukura, N., Fisher, H.B. and Sayre, W.W., "Measurements of Mixing Characteristics of the Missouri River Between Sioux City, Iowa and Plattsmouth, Nebraska," U.S. Geological Survey Water-Supply Paper, 1899-G, 1970.

APPENDIX

USER'S MANUAL AND COMPUTER PROGRAM

This appendix describes the computer program which solves the convection-diffusion equation for transient two-dimensional mixing in natural rivers using collocation finite-element method.

Main variable names used in this FORTRAN program are as follows:

NX	=	Number of elements in η-direction
NY	=	Number of elements in ξ-direction
IA1	=	Number of Gaussian points in η-direction
IA2	=	Number of Gaussian points in ξ-direction
IA3	=	Total number of interior points, boundary points and corners, i.e. IA3 = 4*(Nx+1)* (NY+1)
K	=	Number of segments in ξ-direction
ELZERO	=	(ℓ_o) reference length
BF	=	Average width of the channel
R	=	Hydraulic radius
S	=	Bed slope
DT	=	Time increment
LTEST	=	Control parameter for calling APPR subroutine to perform the interpolation procedure
UZERϕ	=	Reference velocity (u_o)
CØ	=	Reference concentration (C_o)
USTAR	=	Shear velocity (u_*)
NCCS	=	Number of control cross section
ICDN	=	Control parameter for the mesh size of the elements in ξ-direction
N1	=	Integer array contains the number of elements in each control cross section

XCR	=	Dimensional x-coordinates of each cross section
AK	=	Real array contains ξ-coordinates of the nodes in ξ-direction
RPKC	=	Source concentration at the Gaussian points upstream
SEGI	=	Gaussian points longitudinally
SEGJ	=	Gaussian points transversely
XCOR	=	η-coordinate of the nodal points longitudinally
YCOR	=	ξ-coordinates of the nodal points transversely
F1,F2,F3, F4,F5,F6, F7	=	Input values for the coefficients of the P.D.E. (required if LTEST = 1)
MR	=	Control parameter for calling APRINT subroutine
QTOT	=	Total discharge
NNT	=	Number of time steps
KIN	=	Control parameter for source distribution
TPEAK	=	Time for source concentration to reach peak
T	=	Dimensional time
TAU	=	Non-dimensional time
DTAU	=	Non-dimensional time increment
B	=	Solution at time T
B1	=	Solution at time T + DT
NM	=	Number of points other than nodal points where concentration is to be evaluated
XX	=	x-coordinates of the points where concentration is to be evaluated
YY	=	ξ-coordinates of the points where concentration is to be evaluated
LPRINT	=	Control parameter to printing procedure
AM	=	One-dimensional array contains the actual nonzero entries of the resulting system
IAP	=	Row pointers of AM

IRR	=	Order of rows of AM
ICL	=	Order of columns of AM
IC	=	Inverse of ICL
ITEMP	=	Interger array for internal use
MAX	=	Maximum number of off-diagonal nonzero entries of the upper triangular matrix which may be stored
WORK	=	The right hand side of the linear system of equations
RTEMP	=	Real array for internal use
Wϕ	=	Width of the channel upstream
ICD	=	Control parameter for metric coefficient values
CEMX	=	Constant value for m_x
CEMZ	=	Constant value for m_z
NPTS	=	Number of data points at the ith cross section (for approximation purposes)
V	=	Velocity at the given points
Z	=	z-coordinates of the given points
H1	=	Depth at each given point
Q	=	Discharge at the given points
QC	=	Cumulative discharge at the given points
QCK	=	Cumulative discharge at the kth segment
ALPHA	=	Elder's coefficient α
BETA	=	Elder's coefficient β
AL	=	Length of the central line at the ith control cross section (L)
ALR	=	Length of the right side of the ith control cross section (L_R)
ALL	=	Length of the left side of the ith control cross section (L_L)
VN	=	Velocity at the Gaussian points

HN	=	Depth at the Gaussian points
FHIN	=	Non-dimensional depth at the Gaussian points (ϕ)
EMXN	=	Metric coefficient at the Gaussian points (m_x)
EMZN	=	Metric coefficient at the Gaussian points (m_z)
EPSXN	=	Longitudinal diffusion coefficient at the Gaussian points (ε_x)
EPSZN	=	Transverse diffusion coefficient at the Gaussian points (ε_z)
FF1,FF2, FF3,FF4, FF5,FF6, FF7	=	Real arrays contain the coefficients of the P.D.E.
CMXϕ	=	Elemental matrix ($K_x^{(0)}$)
CMX1	=	Elemental matrix ($K_x^{(1)}$)
CMX2	=	Elemental matrix ($K_x^{(2)}$)
CMYϕ	=	Elemental matrix ($K_y^{(0)}$)
CMY1	=	Elemental matrix ($K_y^{(1)}$)
CMY2	=	Elemental matrix ($K_y^{(2)}$)
W	=	Width of cross section (i)

The MAIN program reads input data and controls the execution of the other subroutine subprograms. All variable names and arrays which start with the alphabets I, J, K, L, M and N are INTEGERs. A flow chart which outlines the program data input structure is presented on pages 66-69. The following is a list of subroutines and their functions:

(1) APRINT: Used to evaluate the concentration at the different points other than the nodal points. In the η-direction the coordinates of the points must be read as an input data. In ξ-direction the points are fixed with coordinates (0, 0.05 m), m=0,...,20. This subroutine is called as follows:

```
CALL APRINT(NX,NY,IA1,IA2,IA3,XCOR,XX,NM,YCOR,YY)
```

(2) GAUSSP: Calculate the coordinates of the Gaussian points in η and ξ-direction. The calling statement is,

```
CALL GAUSSP(NX,NY,IA1,IA2,IA3,XCOR,YCOR)
```

(3) APPR: The interpolation procedure is performed in this subroutine. Mainly, it reads the input data for all variablex in x-z coordinates, transform those data to η-ξ coordinates, apply piecewise linear approximation in both directions and finally find the new values of all parameters at the Gaussian points. The calling statement is,

```
CALL APPR(NX,NY,IA1,IA2,IA3,K1,NCS,XCR,QTOT)
```

(4) VALUE1: Used to find the values of the different variable coefficients in the main P.D.E. The calling statement is

```
CALL VALUE1(NX,NY,IA1,IA2,IA3,QTOT)
```

(5) CINPUT: Calculates the source distribution upstream for time step T second. The calling statement is

```
CALL CINPUT(C0,T,TPEAK,IA1,IA2,RPKC,KIN,TAU)
```

(6) ELMAT: Used to calculate the elemental collocation matrices CMX0, CMX1, CMX2, CMY0, CMY1, CMY2. The calling statement is

```
CALL ELMAT(NX,NY,IA1,IA2,IA3,I,J)
```

(7) CORNER: Build up the equation results when applying the boundary conditions at the four corner points. The calling statement is,

```
CALL CORNER(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
```

(8) BOUNDX: Construct the equation results when applying the boundary conditions at the Gaussian points (left and right bank). The calling statement is,

```
CALL BOUNDX(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
```

(9) BOUNDY: Build up the equation results when applying the boundary conditions upstream and downstream at the Gaussian points. The calling statement is,

```
CALL BOUNDY(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
```

(10) AINTP: Construct the equation results when applying the P.D.E. at the interior points for each element. The calling statement is,

```
CALLAINTP(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
```

(11) SUBC: This subroutine is called by AINTP. It performs some intermediate calculations required by AINTP subroutine. The calling statement is,

```
CALL SUBC(I,J,NX,NY,IA1,IA2,IA3,ICODE,DTAU)
```

(12) TENSRP: Called by SUBC subroutine. It performs the tensor product of two matrices each of dimension 2x4. The calling statement is,

```
CALL TENSRP(AKX,AKY,AKE,ICODE,CONST)
```

(13) AADD: Called by SUBC subroutine to accumulate the values of the resulting matrix of dimension 4x16 which is used by AINTP subroutine. The calling statement is,

```
CALL AADD(NX,NY,IA1,IA2,IA3,ICODE,AKE,JJ)
```

(14) FV1: Functional subprogram used to assign values at downstream boundary.

(15) FV2: Assign values at left and right bank.

(16) ALAM1: Assign values for $\lambda_1^* (\eta,\xi,\tau)$.

(17) ALAM2: Assign values for $\lambda_2^* (\eta,\xi,\tau)$.

(18) GELLM: Subroutine subprogram used to solve the linear system of equations. The calling statement is,

```
CALL GELLM(N,MAX,IERR,ITEMP,RTEMP)
```

(19) PREORD: Used for re-ordering rows anc columns of the matrix AM. The calling statement is,

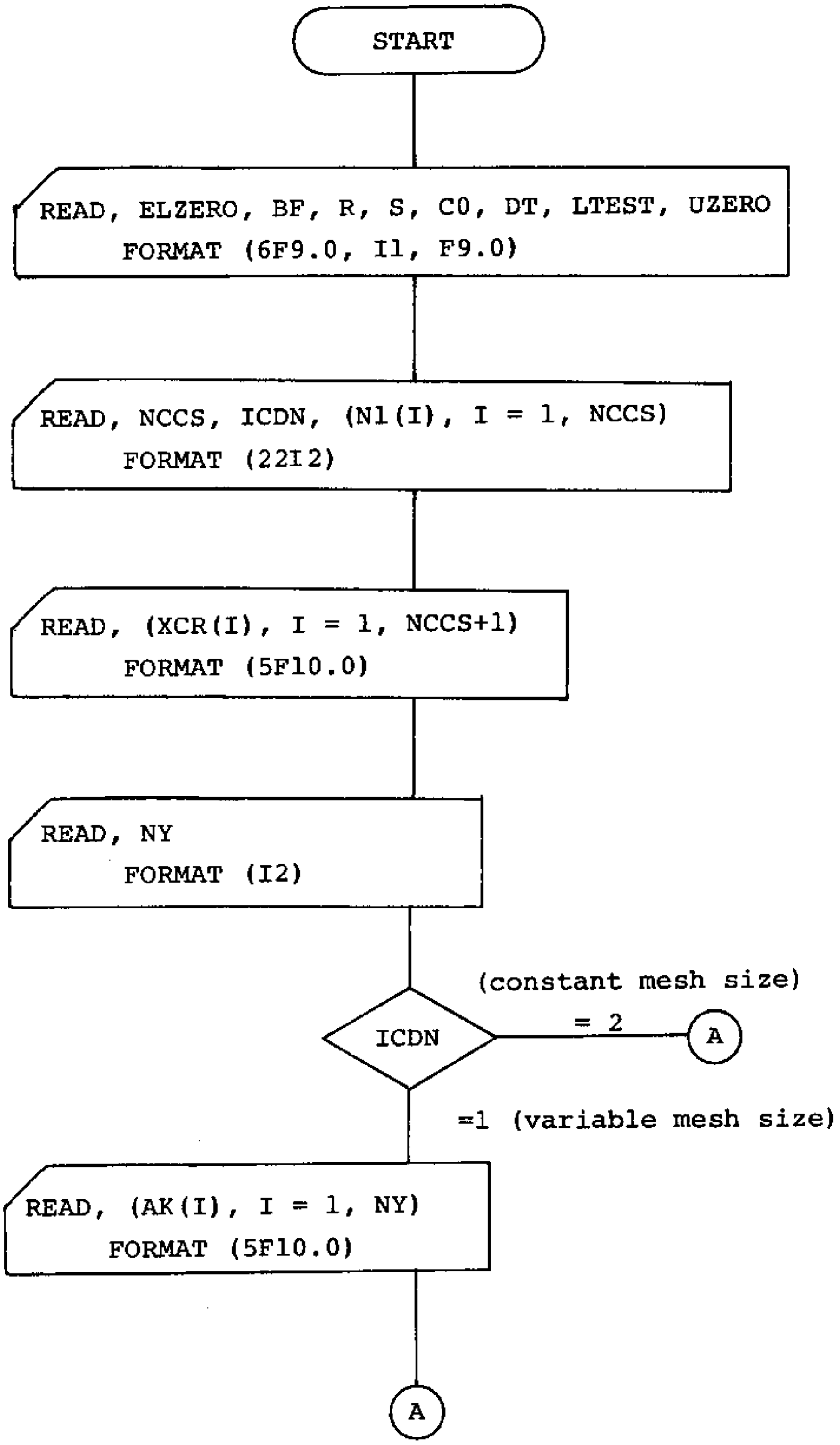
```
CALL PREORD(N)
```

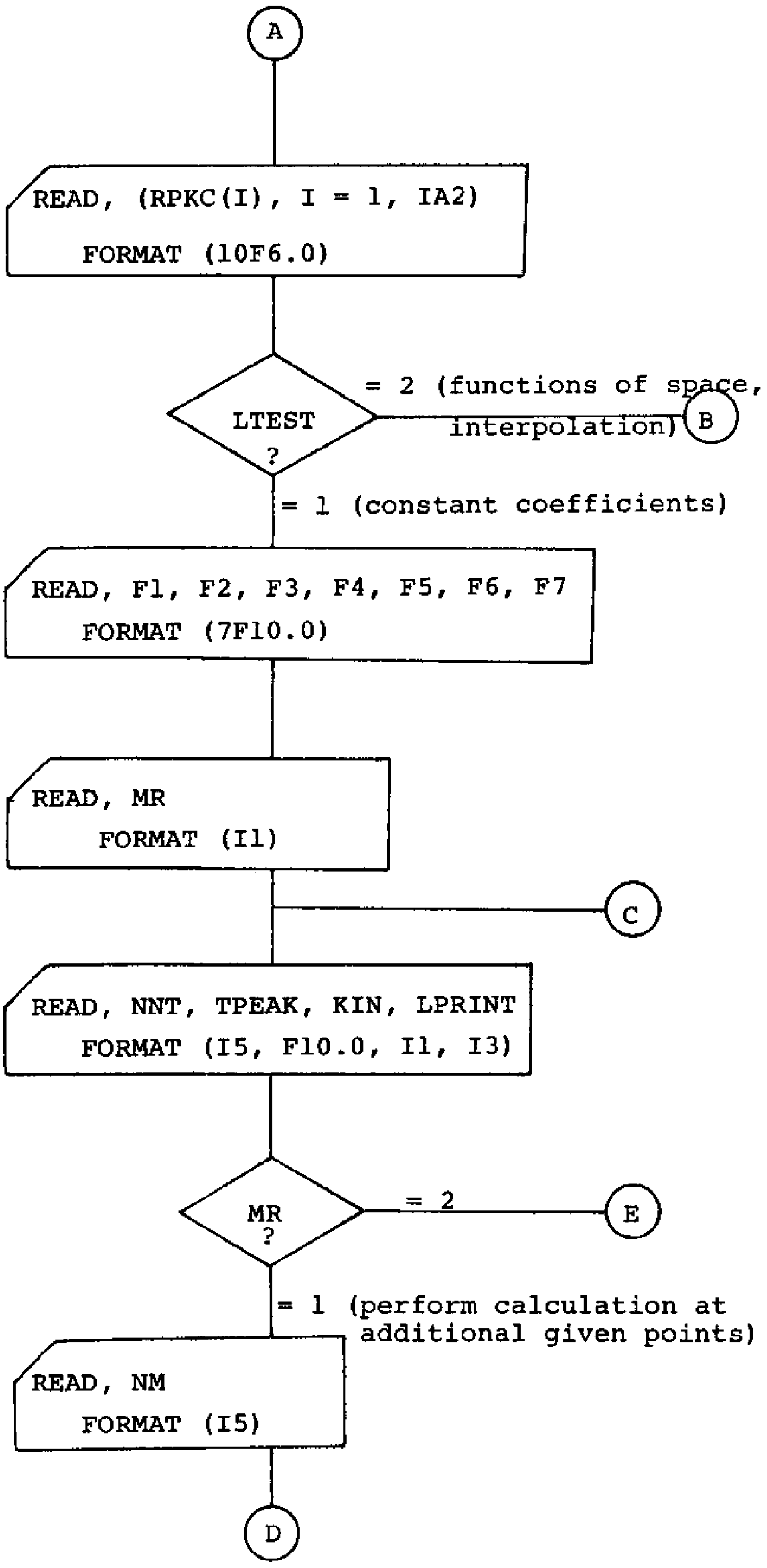
(20) GAUSPV: Called by GELLM. Perform all operations for factorization the matrix AM to upper and lower triangular matrices and solve the linear system by using Gaussian elimination with partial pivoting (using column interchanges).

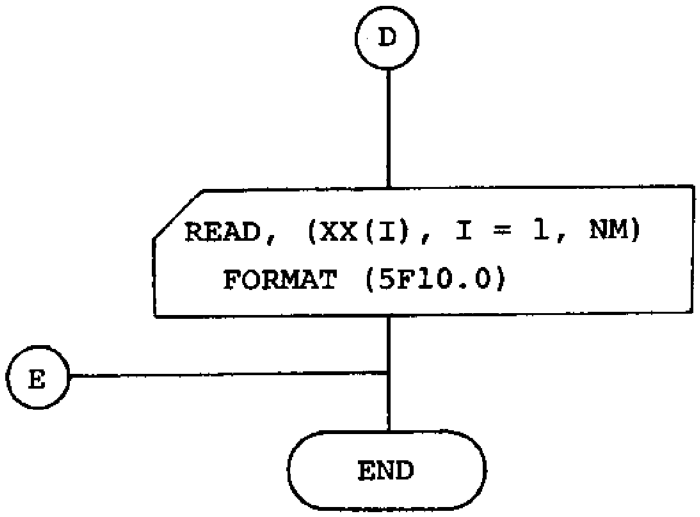
Memory size required to run this program is 250K bytes using single precision arithmetic or 500K bytes when using double precision arithmetic.

The maximum number of elements that can be used is 100, 10 elements along each coordinate, for example. This can be increased by changing the dimensions of the arrays.

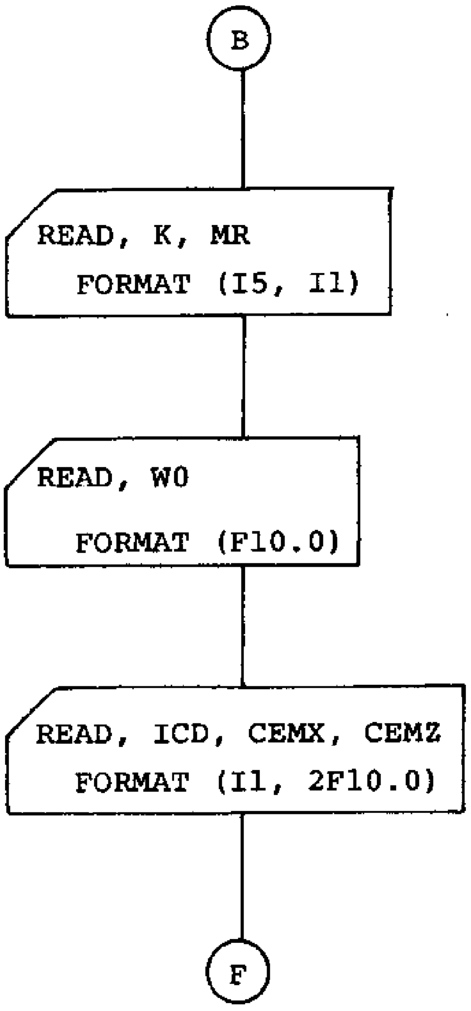
FLOW CHART FOR DATA INPUT




A
READ, (RPKC(I), I = 1, IA2)
FORMAT (10F6.0)
LTEST ?
= 2 (functions of space, interpolation)
B
= 1 (constant coefficients)
READ, F1, F2, F3, F4, F5, F6, F7
FORMAT (7F10.0)
READ, MR
FORMAT (I1)
C
READ, NNT, TPEAK, KIN, LPRINT
FORMAT (I5, F10.0, I1, I3)
MR ?
= 2
E
= 1 (perform calculation at additional given points)
READ, NM
FORMAT (I5)
D



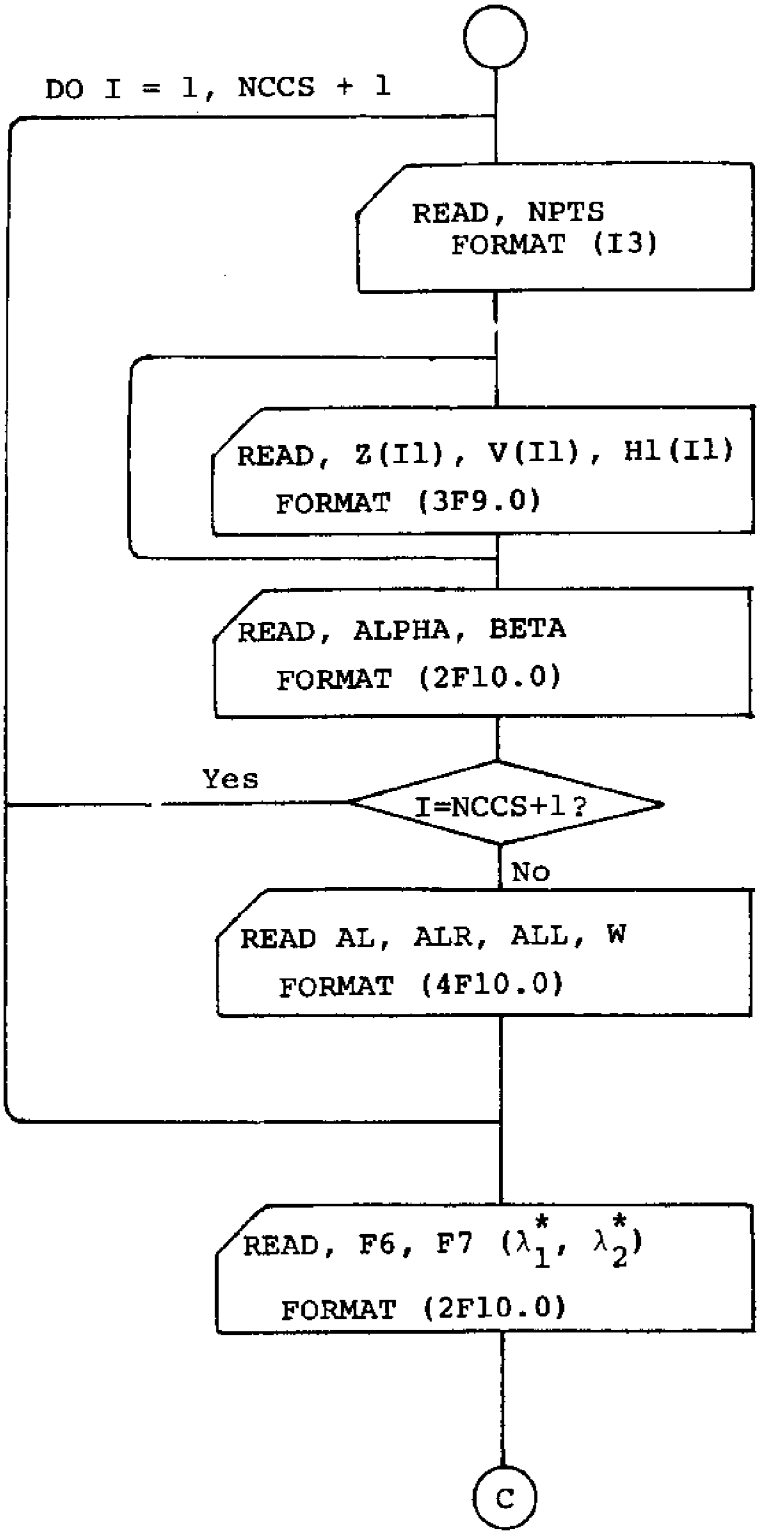
```
READ, (XX(I), I = 1, NM)
  FORMAT (5F10.0)
```



```
READ, K, MR
  FORMAT (I5, I1)
```

```
READ, W0
  FORMAT (F10.0)
```

```
READ, ICD, CEMX, CEMZ
  FORMAT (I1, 2F10.0)
```


DO I = 1, NCCS + 1
READ, NPTS
FORMAT (I3)
READ, Z(I1), V(I1), H1(I1)
FORMAT (3F9.0)
READ, ALPHA, BETA
FORMAT (2F10.0)
Yes
I=NCCS+1?
No
READ AL, ALR, ALL, W
FORMAT (4F10.0)
READ, F6, F7 (λ_1^*, λ_2^*)
FORMAT (2F10.0)
C

PROGRAM LISTING

```
/INCLUDE OSJE
SYSTEM='VS1'.RETURN
//CX01H705 JOB CX01.CLASS=A.REGION=500K
/*JOBPARM TIME=30.LINES=8
/*ROUTE  PRINT MUSIC
/*ROUTE  PUNCH MUSIC
//  EXEC GIFORTG.PARM.GO='SIZE=500000.MAP.LIST.PRINT'.
//  REGION.GO=500K
//FORT.SYSIN  DD  *
C.....................MAIN       PROGRAM......................
C                                                            .
C                   ..........................               .
C                   .THIS PROGRAM IS USED TO SO.             .
C                   .-LVE TWO DIMENSIONAL DISS-.             .
C                   .PERSION PROBLEM  USING THE.             .
C                   .FINITE ELEMENT   METHODS  .             .
C                   ..........................               .
C                                                            .
C                                                            .
C.............................................................
C                                                            .
C            ::::::::::::::::::::::::::::::::::::::          .
C            :PREPARED BY: YAHIA   S.    HALABI     :        .
C            :CLARKSON  COLLEGE  OF TECHNOLOGY      :        .
C            :MATH.   AND COMPUTER SCIENCE DEPT.    :        .
C            ::::::::::::::::::::::::::::::::::::::          .
C                                                            .
C.............................................................
C
      IMPLICIT REAL*8(A-H.O-Z)
      INTEGER*2  LOC.IAP.IRR.ICL.IC.ITEMP
      DIMENSION N1(20).XCR(22).RPKC(22).XCOR(11).YCOR(11)
      DIMENSION RTEMP(16000).ITEMP(16418)
      DIMENSION    C(22.22).WORK(484).XX(20).YY(21)
      DIMENSION    AK(10).H(10).SEGI(22).SEGJ(22)
      DIMENSION    VN(22.22).HN(22.22).FHIN(22.22).EMXN(22.22)
      DIMENSION    EMZN(22.22).EPSXN(22.22).EPSZN(22.22)
      DIMENSION    FF1(22.22).FF2(22.22).FF3(22.22).FF4(22.22).
     1          FF5(22.22).FF6(22.22).FF7(22.22)
      DIMENSION    B(484).B1(484).AM(6724)
      DIMENSION    CMX0(2.4).CMX1(2.4).CMX2(2.4).CMY0(2.4).
     1CMY1(2.4).CMY2(2.4).CC(4.16).CC1(4.16)
      COMMON LOC(6724).IAP(488).IRR(484).ICL(484).IC(484)
      COMMON C.WORK.AK.H.SEGI.SEGJ.USTAR.BF.R.UZERO.ELZERO.
     1VN.HN.FHIN.EMXN.EMZN.EPSXN.EPSZN.FF1.FF2.FF3.FF4.FF5.FF6.
     1FF7.B.B1.AM.CMX0.CMX1.CMX2.CMY0.CMY1.CMY2.CC.CC1
C------
C READ THE REFF. LENGTH
C------
      MAX=16000
      READ(5.1)ELZERO.BF.R.S.CO.DT.LTEST.UZERO
1     FORMAT(6F9.3.I1.F9.3)
      USTAR=(32.2*R*S)**.5
      WRITE(6.10)ELZERO.BF.R.S.CO.DT.LTEST.UZERO.USTAR
10    FORMAT(1H1.//.5X.'ELZERO='.F10.3.5X.'AVERAGE WIDTH='.F10.3.
     1           /.5X.'HYDRULIC RADIOUS='.F10.3.5X.'BED SLOPE='.
     1           F10.4./.5X.'REF. CONCENTRATION='.F10.4.5X.
     1'TIME INCREMENT='.F10.3./5X.'LTEST ='.I2.5X.
     1'UZERO='.F10.4./5X.'USTAR='.F10.5)
C-------
C READ NUMBER OF CONTROL CROSS SECTIONS.NO. OF ELEMENTS IN EACH
```

```
C  CONTROL CROSS SECTION.CODE FOR NUMBER OF ELEMENS VERTICALLY.
C   1-FEED THE LENGTH OF ELEMENTS.2- CONSIDER EQUALLY LENGTH.
C-------
      READ(5,15)NCCS,ICDN,(N1(I),I=1,NCCS)
15    FORMAT(22I2)
C-------
C NCS IS THE NUMBER OF CROSS SECTIONS
C NX,NY ARE THE NUMBER OF ELEMENTS IN X AN QC COORD.
C-------
      NCS=NCCS+1
      NX=0
      DO 20 I=1,NCCS
20    NX=NX+N1(I)
      WRITE(6,25)NCCS,ICDN,NX,(N1(I),I=1,NCCS)
25    FORMAT(//5X,' NUMBER OF CONTROL CROSS SECTIONS=',I2,
     1/5X,' ICDN=',I2,/5X,'TOTAL NO. OF ELEMENTS IN X-DIRECTION',
     1'=',I2,/5X,'NO. OF ELEMENTS IN EACH CONTROL CROSS SECTION=',
     1/,5X,20(I2,1X))
C------
C READ THE DIMENTIONAL X- COORDINATES OF EACH CROSS SECTION
C------
      READ(5,30)(XCR(I),I=1,NCS)
30    FORMAT(5F10.3)
      WRITE(6,35)(XCR(I),I=1,NCS)
35    FORMAT(/5X,'DIMENSIONAL X-COORD.',/5X,4(5F10.3,/,5X))
      AB=XCR(1)
      DO 1909 I=2,NCS
1909  XCR(I)=XCR(I)-XCR(1)
      XCR(1)=0.D0
C------
C CALCULATE THE NON-DIMENSIONAL X-COORD.
C------
      DO 40 I=1,NCS
40    XCR(I)=XCR(I)/ELZERO
C------PRINT THE NON-DIMENSIONAL X-COORD.
      WRITE(6,45)(XCR(I),I=1,NCS)
45    FORMAT(/5X,'NON-DIMENSIONAL X-COORD.',/5X,4(5F10.3/,5X))
C------READ THE NUMBER OF ELEMENTS IN QC-COORD.
      READ(5,15)NY
      WRITE(6,50)NY
50    FORMAT(/5X,' NUMBER OF ELEMENTS NY=',I2)
      GO TO (55,65),ICDN
55    READ(5,60)(AK(I),I=1,NY)
60    FORMAT(5F10.5)
      GO TO 75
65    DO 70 I=1,NY
70    AK(I)=1./DFLOAT(NY)
75    WRITE(6,80)(AK(I),I=1,NY)
80    FORMAT(5X,'ELEMENT SIZE IN QC-DIRECTION'/5X,10(6F10.5,/,5X))
C------CALCULATE THE COORD. OF ELEMENTS IN QC-DIRECTION.
      YCOR(1)=0.D0
      L=NY+1
      DO 85 I=2,L
85    YCOR(I)=YCOR(I-1)+AK(I-1)
      YCOR(NY+1)=1.D0
      WRITE(6,90)(YCOR(I),I=1,L)
90    FORMAT(5X,'QC-COORD. OF NODAL POINTS'/5X,5(6F10.4,/,5X))
      IA2=2*NY+2
C------
C READ INITIAL CONDITION UP-STREAM.
C------
```

```
      READ(5,95)(RPKC(I),I=1,IA2)
95    FORMAT(10F6.2)
      WRITE(6,100)(RPKC(I),I=1,IA2)
100   FORMAT(5X,'INITIAL CONDITION'/5X,6(6F10.5,/,5X))
C------
C------
      L=NY+1
C------
C CALCULATE THE X-COORD. AND MESH SIZE OF EACH ELEMENTS IN X-COORD.
C------
      L1=1
      XCOR(1)=XCR(1)
      DO 110 I=1,NCCS
      A1=(XCR(I+1)-XCR(I))/DFLOAT(N1(I))
      L2=L1+N1(I)-1
      DO 105 I1=L1,L2
      H(I1)=A1
105   XCOR(I1+1)=XCOR(I1)+A1
      XCOR(NX+1)=XCR(NCS)
      L1=L2+1
110   CONTINUE
      L1=NX+1
      WRITE(6,115)(XCOR(I),I=1,L1)
115   FORMAT(/5X,'XCOR-OF NODAL POINTS'/5X,6(6F12.5/,5X))
      WRITE(6,120)(H(I),I=1,NX)
120   FORMAT(/5X,'MESH SIZE OF EACH ELEMENT IN X-DIRECTION'
     1/5X,6(5F12.5,/))
      IA1=2*NX+2
      IA3=IA1*IA2
C------
C FIND THE COORD. OF GAUSSIAN POINTS
C------
      CALL GAUSSP(NX,NY,IA1,IA2,IA3,XCOR,YCCR)
      WRITE(6,121)
121    FORMAT(/5X,'GAUSSIAN POINTS HORIZENTALLY')
      WRITE(6,122)(SEGI(L),L=1,IA1)
122   FORMAT(5X,10(6F10.5,/,5X))
      WRITE(6,123)
123    FORMAT(/5X,'GUASSIAN POINTS VERTICALLY')
      WRITE(6,122)(SEGJ(L),L=1,IA2)
C------------
C TEST IF CONSTANT COEF. FOR P.D.E WILL BE READ,
C  OR APPROXIMATION TECHNIQUE OPTION.
C LTEST=1,READ THE CONSTANT COEF.
C LTEST=2, DO APPROXIMATION TECHNIQUE.
C------------
C..........                    ....................
      GO TO (91,92),LTEST
91    READ(5,651)F1,F2,F3,F4,F5,F6,F7
651   FORMAT(7F10.4)
      WRITE(6,93)F6,F7
93    FORMAT(5X,'NON-DIMENSIONAL -(LAMDA1)',F10.5,
     1/5X,'NON-DIMENSIONAL -(LAMDA2)',F10.5)
      READ(5,599)MR
599   FORMAT(I1)
      WRITE(6,641)MR
      GO TO 1136
92    CONTINUE
C------
C APPROXIMATE THE PARAMETERS IN THE NEW COORDINATES
C  READ THE NUMBER OF SEGMENTS TRAVERSLY (K)
```

```
C------
      READ(5,125)K,MR
125   FORMAT(I5,I1)
      WRITE(6,641)MR
641   FORMAT(5X,'OPTION CODE FOR APRINT SUB. IS. MR=',I1)
C--MR=1 -READ NM,XX,AND CALL APRINT
C--MR=2 -NON OF THE ABOVE
C  WHERE MR IS OPTION FOR MORE APPROX. AT ADDITIONAL POINTS
      K1=K+1
      CALL APPR(NX,NY,IA1,IA2,IA3,K1,NCS,XCR,QTOT)
C------
C FIND THE VALUES OF FUNCTIONS IN  (P. D. E.)
C------
      READ(5,130)F6,F7
130   FORMAT(2F10.5)
      WRITE(6,135)QTOT,F6,F7
135   FORMAT(1H1,///,5X,'TOTAL DISCHARGE=',F12.3,
     1/5X,'LAMDA1=',F10.4,2X,' , LAMDA2=',F10.4)
      CALL VALUE1(NX,NY,IA1,IA2,IA3,QTOT)
1136  CONTINUE
C-----WRITE(6,136)
136   FORMAT(10X,'FF1',10X,'FF2',10X,'FF3',10X,'FF4',10X,'FF5',
     1/2X,70(1H:))
      DO 138 I=1,IA1
      DO 138 J=1,IA2
      GO TO (1380,1381),LTEST
1380  CONTINUE
      FF1(I,J)=F1
      FF2(I,J)=F2
      FF3(I,J)=F3
      FF4(I,J)=F4
      FF5(I,J)=F5
1381  CONTINUE
C-----WRITE(6,137)I,J,FF1(I,J),FF2(I,J),FF3(I,J),FF4(I,J),FF5(I,J)
137   FORMAT(1X,I3,',',I3,F7.4,4(3X,F10.4))
138   CONTINUE
C------
C READ THE NUMBER OF TIME STEPS,TPEAK,INITIAL CONDITION CODE
C  AND LPRINT(PRINTING COUNT CODE).
C------
      READ(5,140)NNT,TPEAK,KIN,LPRINT
140   FORMAT(I5,F10.3,I1,I3)
      WRITE(6,145)NNT,TPEAK,KIN,LPRINT
145   FORMAT(5X,'NUMBER OF TIME STEPS=',I5,5X,'TPEAK=',F10.3,
     1/5X,'KIN=',I1,10X,'PRINT EVERY ',I3,' TIME STEP.')
C------
C I N I T I A L I Z A T I O N
C------
      T=0.D0
      TAU=0.D0
      DTAU=DT*UZERO/ELZERO
      DO 150 I=1,IA3
      B(I)=0.D0
150   B1(I)=0.D0
C------
C L O O P   F O R   T I ME   S T E P S ...............
C::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C-----
CREAD THE COORD. OF POINTS AT WHICH CONCENTRATION
C IS TO BE EVALUATED.
C-----
```

```
      GO TO(719,717),MR
719   CONTINUE
      READ(5,997)NM
997   FORMAT(I5)
      READ(5,996)(XX(I),I=1,NM)
996   FORMAT(5F10.4)
      WRITE(6,998)(XX(I),I=1,NM)
998   FORMAT(5X,'X-COORD. OF POINTS AT WHICH CONCENTRATION',
     1'  IS TO BE EVALUATED',/5X,5(6F10.3/5X))
      DO 1221 I=1,NM
1221  XX(I)=XX(I)-AB
      YY(1)=0.D0
      DO 995 I=2,21
995   YY(I)=YY(I-1)+0.05D0
      YY(21)=1.D0
      DO 994 I=1,NM
994   XX(I)=XX(I)/ELZERO
      WRITE(6,993)(XX(I),I=1,NM)
993   FORMAT(5X,'NON-DIMENSIONAL COORD.'/5X,6(6F10.4/5X))
717   CONTINUE
      DO 886 I=1,IA1
      DO 886 J=1,IA2
886   C(I,J)=0.D0
      WRITE(6,889)
889   FORMAT(1H1)
      DO  300  ML =1,NNT
      DO 116 IP=1,IA3
      ICL(IP)=IP
      IC(IP)=IP
116   IRR(IP)=IP
      DO 155 I=1,IA3
      WORK(I)=0.D0
155   CONTINUE
      L=NX*NY*64+16*NX+16*NY+4
      IAP(1)=1
      IAP(IA3+1)=L+1
      DO 141 I=1,L
141   AM(I)=0.D0
      TAU=TAU+DTAU
      T=T+DT
C------
C EVALUATE  INITAL  CONDITIONS
C------
      CALL CINPUT(CO,T,TPEAK,IA1,IA2,RPKC,KIN,TAU)
C------
C CALCULATE THE COEFF. IN P.D.E. (FF6,FF7)
C------
      DO 160 I=1,IA1
      DO 160 J=1,IA2
      A1=SEGI(I)
      A2=SEGJ(J)
      ELZ=ELZERO
      UZR=UZERO
      FF6(I,J)=ALAM1(A1,A2,TAU,ELZ,UZR,F6)
160   FF7(I,J)=ALAM2(A1,A2,TAU,ELZ,UZR,CO,F7)
C------
C INITIALIZE  ROW NUMBER,FIND THE ENTRIES IN THE BIG
C MATRICES RESULTING AT  1- CORNER
C                        2-BOUNDARY OF X(LEFT AND RIGHT BANKS)
C                        3-BOUNDARY OF Y-UP AND DOWN STREAM
C                        4-AT THE ENTERIOR POINTS.
```

```
C------------------------
C------
      LN=0
      IORD=0
      LM=0
      CALL CORNER(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
      CALL BOUNDX(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
      CALL BOUNDY(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
      CALL AINTP(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
C------
C------------------------
C------
C------
C SOLVE THE RESULTING SYSTEM  AM*(B1)=WORK FOR (B1)
C USING GAUSSIAN ELIMINATION METHOD.
C------
      CALL PREORD(IA3)
      CALL GELLM(IA3,MAX,IERR,ITEMP,RTEMP)
      DO 839 I=1,IA3
839   WORK(I)=B1(I)
C------
C CHECK  FOR CONVERGENCE
C------
      IF(ML-ML/LPRINT*LPRINT)1179,305,1179
305   CONTINUE
       WRITE(6,3671)IERR
3671   FORMAT(5X,'IERR = ',I10)
      WRITE(6,310)T,TAU
310   FORMAT(1H0,4X,'COEFFICIENTS   AFTER  ',F10.2,
     1'          ','  ,TAU=',F10.5/,5X,65(1H=)/,
     115(10F10.4/))
      L1=-1
      L2=NY+1
      MM=2*IA2
      DO 1234 I=1,L2
      L1=L1+2
1234  WRITE(6,1245)(B1(J),J=L1,IA3,MM)
1245  FORMAT(5X,10(6F10.4,/,5X))
      GO TO (1178,1179),MR
1178  CONTINUE
      CALL APRINT(NX,NY,IA1,IA2,IA3,XCOR,XX,NM,YCOR,YY)
1179  CONTINUE
      DO 315 I1=1,IA3
315   B(I1)=B1(I1)
300   CONTINUE
999   STOP
      END
      SUBROUTINE APRINT(NX,NY,IA1,IA2,IA3,XCOR,XX,NM,YCOR,YY)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2 LOC,IAP,IRR,ICL,IC
      DIMENSION XCOR(11),XX(20),CNEW(30,30),C(22,22),WORK(484)
     1,YCOR(11)
      DIMENSION AK(10),H(10),SEGI(22),SEGJ(22),FI(4),FJ(4),YY(21)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ
      DO 99 I=1,30
      DO 99 J=1,30
99    CNEW(I,J)=0.D0
      DO 1 I=1,NM
      DO 11 II=1,NX
      IF((XX(I).GE.XCOR(II)).AND.(XX(I).LE.XCOR(II+1)))GO TO 10
```

```
11     CONTINUE
       WRITE(6,130)XX(I)
130    FORMAT(10X,F10.6,' IS WRONG X-COORD.')
       RETURN
10     XL1=(XX(I)-XCOR(II))/H(II)
       XL2=(XX(I)-XCOR(II+1))/H(II)
       FI(1)=2.D0*(XL1**3)-3.D0*(XL1**2)+1.D0
       FI(2)=((XL1**3)-2.C0*(XL1**2)+XL1)*H(II)
       FI(3)=-2.D0*(XL2**3)-3.D0*(XL2**2)+1.D0
       FI(4)=((XL2**3)+2.C0*(XL2**2)+XL2)*H(II)
       DO 100 K=1,21
       DO 111 KK=1,NY
       IF((YY(K).GE.YCOR(KK)).AND.(YY(K).LE.YCOR(KK+1)))GC TO 13
111    CCNTINUE
       WRITE(6,131)YY(K)
131    FORMAT(10X,F10.6,' IS WRONG Y-COORD.')
       RETURN
13     YL1=(YY(K)-YCOR(KK))/AK(KK)
       YL2=(YY(K)-YCOR(KK+1))/AK(KK)
       FJ(1)=2.D0*(YL1**3)-3.D0*(YL1**2)+1.D0
       FJ(2)=((YL1**3)-2.D0*(YL1**2)+YL1)*AK(KK)
       FJ(3)=-2.D0*(YL2**3)-3.D0*(YL2**2)+1.C0
       FJ(4)=((YL2**3)+2.D0*(YL2**2)+YL2)*AK(KK)
       DO 106 MN1=1,4
       LN=2*(MN1-1)*(NY+1)+2*(KK-1)+4*(NY+1)*(II-1)
       DO 106 MN2=1,4
       LN=LN+1
       CNEW(I,K)=CNEW(I,K)+(FI(MN1)*FJ(MN2))*WORK(LN)
106    CONTINUE
100    CONTINUE
 1     CONTINUE
       WRITE(6,107)
107    FORMAT(///,5X,'RELATIVE CONCENTRATICN'/5X,65(1H=))
       DO 108 J=1,21
108    WRITE(6,109)(CNEW(I,J),I=1,NM)
109    FORMAT(5X,15(6F10.4/,5X))
       WRITE(6,110)
110    FORMAT(5X,65(1H=))
       RETURN
       END
C-----------------------------------------------------------------
C THIS SUBROUTINE IS USEC TC CALCULATE THE GAUSSIAN  POINTS     :
C IN X COORDINATE  AND  OC  CCORDINATES.                        :
C-----------------------------------------------------------------
       SUBROUTINE GAUSSP(NX,NY,IA1,IA2,IA3,XCOR,YCOR)
       IMPLICIT REAL*8(A-H,C-Z)
       INTEGER*2  LOC,IAP,IRR,ICL,IC
       DIMENSION XCOR(11),YCOR(11)
       DIMENSION    C(22,22),WORK(484)
       DIMENSION    AK(10),H(10),SEGI(22),SEGJ(22)
       COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
       COMMON C,WCRK,AK,H,SEGI,SEGJ
       X1=XCOR(1)
       X2=XCOR(1)
       SEGI(1)=X1
       DO 1 I=1,NX
       X2=X2+H(I)
       SEGI(2*I)=(X1+X2)/2.C0-H(I)/(2.C0*CSQRT(3.D0))
       SEGI(2*I+1)=(X1+X2)/2.D0+H(I)/(2.C0*DSQRT(3.D0))
       X1=X2
1      CCNTINUE
```

```
      SEGI(IA1)=XCOR(NX+1)
C------
C------
      Y1=YCOR(1)
      Y2=YCOR(1)
      SEGJ( 1)=Y1
      DO 2  J=1,NY
      Y2=Y2+AK(J)
      SEGJ(2*J)=(Y1 + Y2)/2.D0-AK(J)/(2.D0*DSQRT(3.D0))
      SEGJ(2*J+1)=(Y1+Y2)/2.D0+AK(J)/(2.D0*DSQRT(3.D0))
      Y1=Y2
2     CONTINUE
      SEGJ(IA2)=YCOR(NY+1)
      RETURN
      END
C-----------------------------------------------------------------
C THIS SUBROUTINE IS USED TO APPROXIMATE THE DATA (SET)  :
C OF PARAMETERS LINEARLY,HORIZENTALLY AND TRAVERSELY     :
C , FIRST STEP IS TO TRANSFORM THE VALUES FROM X-Z COOR.:
C  TO THE CORRESPONDING VALLES IN X-QC COORDINATES.      :
C-----------------------------------------------------------------
      SUBROUTINE APPR(NX,NY,IA1,IA2,IA3,K1,NCS,XCR,QTOT)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION XCR(22),Z(22),V(22),H1(22),QC(22),Q(22),
     1QK(22),ZK(22,22),VK(22,22),HK(22,22),EMX(22,22),
     1EMZ(22,22),FHI(22,22),EPSX(22,22),EPSZ(22,22)
      DIMENSION QCK(22,22),WV(22,22),WH(22,22),WP(22,22),
     1WEMX(22,22),WEMZ(22,22),WEPX(22,22),WEPZ(22,22)
C------
      DIMENSION C(22,22),WORK(484)
      DIMENSION AK(10),H(10),SEGI(22),SEGJ(22)
      DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
      DIMENSION EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,
     1ELZERO,VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN
C------
      VFUN(A1,A2,A3,A4,A5)=A1+(A2-A1)/A3*(A4-A5)
C------
      WRITE(6,1)
1     FORMAT(1H1,//,5X,'APPROXIMATIOON  TECHNIQUE')
      READ(5,2)W0
2     FORMAT(2F10.3)
      KK=K1-1
      WRITE(6,3)KK,W0
3     FORMAT(/5X,'NO. OF SEGMENTS TRAVERSELY=',I5,
     1       /5X,'WIDTH OF UP-STREAM          =',F10.3)
C------
C READ ICD CODE ,CEMX,CEMZ
C ICD=1--EMX(I,J)=CEMX,EMZ(I,J)=CEMZ
C------
      READ(5,71)ICD,CEMX,CEMZ
71    FORMAT(I1,2F10.3)
      WRITE(6,72)ICD,CEMX,CEMZ
72    FORMAT(5X,'ICD=',I2,3X,'CEMX=',F10.3,3X,'CEMZ=',F10.3)
C------
      DO 4  I=1,NCS
C------
C READ NUMBER OF POINTS AT (I) CROSS SECTION
C------
```

```
      READ(5,5)NPTS
 5    FORMAT(I3)
      WRITE(6,6)XCR(I),NPTS
 6    FORMAT(5X,'ETA=',F10.3,5X,'NO. OF POINTS=',I3)
      WRITE(6,7)
 7    FORMAT(5X,'Z',10X,'V',10X,'H1',6X,'Q',/,2X,33(1H:))
C-----READ Z -COORD.,DIMENSIONAL VELOCITY AND DEPTH
      DO 8 I1=1,NPTS
      READ(5,9)Z(I1),V(I1),H1(I1)
 9    FORMAT(3F9.3)
      AZN=Z(NPTS)
      QK(1)=0.D0
C-----CALCULATE DIMENSIONAL Q
      Q(I1)=V(I1)*H1(I1)
 8    WRITE(6,18)Z(I1),V(I1),H1(I1),Q(I1)
 18   FORMAT(3X,F6.2,4X,F7.3,4X,2F7.3,4X,2F7.3)
C-----CALCULATE COMULATIVE DISCHARGE AT EACH PCINT
      QC(1)=0.C0
      L=NPTS
      DO 10 I1=2,L
 10   QC(I1)=QC(I1-1)+(Q(I1-1)+Q(I1))*(Z(I1)-Z(I1-1))/2.C0
      IF(I .EQ. 1) AW1=QC(NPTS)
      QTOT=AW1
C-----CALCULATE THE NON-DIMENTIONAL Z AND QC
      DO 11 I1=1,NPTS
      Z(I1)=Z(I1)/AZN
 11   QC(I1)=QC(I1)/QTOT
C-----EVALUATE QCK(I,J) AT THE DIFFERENT SEGMENTS ,ZK,FHI,VK,HK
      DQ=1.C0/DFLOAT(KK)
      L=K1
      DO 12 I1=1,L
 12   QCK(I,I1)=(I1-1)/DFLOAT(KK)
      ZK(I,1)=0.D0
      FHI(I,1)=0.D0
      HK(I,1)=0.D0
      VK(I,1)=0.D0
      DO 55 I1=2,L
      DO 14 J=2,NPTS
      IF((QCK(I,I1).GT.QC(J-1)).AND.(QCK(I,I1).LE.QC(J)))GO TO 17
      GO TO 14
 17   ZK(I,I1)=Z(J-1)+(QCK(I,I1)-QC(J-1))/(QC(J)-QC(J-1))*
     1(Z(J)-Z(J-1))
      HK(I,I1)=H1(J-1)+(H1(J)-H1(J-1))/(Z(J)-Z(J-1))*(ZK(I,I1)-Z(J-1))
      IF (I1.EQ.2) GO TO 14
      QK(I1-1)=.5*(DQ/(ZK(I,I1-1)-ZK(I,I1-2))+DQ/(ZK(I,I1)-ZK(I,I1-1)))
      QD=QK(I1-1)*QTOT/AZN
      FHI(I,I1-1)=HK(I,I1-1)/R
      VK(I,I1-1)=QD/(HK(I,I1-1)*UZERO)
      GO TO 54
 14   CONTINUE
 54   FHI(I,L)=0.D0
      HK(I,L)=0.D0
      VK(I,L)=0.D0
 55   QK(L)=0.D0
C-----PRINT THE VALUES IN THE NEW COORDINATES
      WRITE(6,15)
 15   FORMAT(5X,'ZK',10X,'VK',5X,'HK',6X,'FHI',10X,'QCK',
     1/,55(1H:))
      DO 16 I1=1,L
 16   WRITE(6,23)ZK(I,I1),VK(I,I1),HK(I,I1),FHI(I,I1),QCK(I,I1)
 23   FORMAT(3X,F6.2,4X,F7.2,4X,2F7.2,4X,F7.2,5X,F6.2/)
```

```
      READ(5,632)ALPHA,BETA
632    FORMAT(2F10.5)
      WRITE(6,635)ALPHA,EETA
635   FORMAT(2X,'ALPHA=',F10.3,'  BETA=',F10.3)
C------
CCALCULATE THE METRIC   EMX ,EMZ,  AND EPSX,EPSZ
C------
      IF(I .EQ. NCS)GO TO 30
      READ(5,25)AL,ALR,ALL,W
25    FORMAT(5F10.3)
      WRITE(6,26)I,AL,ALR,ALL,W
26    FORMAT(2X,'CROSS SECTION ',I2,/,
     12X,'CENTRAL LENGTH=',F10.3,/,
     12X,'RIGHT SIDE LENGTH=',F10.3,/,
     12X,'LEFT SIDE LENGTH=',F10.3,/,
     12X,'WIDTH OF SEGMENT=',F10.3)
30    CONTINUE
      WRITE(6,28)
28    FORMAT(//,5X,'EMX',8X,'EMZ',8X,'EPSX',8X,'EPSZ'/4X,42(1H:))
      M=K1
      DO 24 I1=1,M
      IF(ICD-1)77,88,77
88    EMX(I,I1)=CEMX
      EMZ(I,I1)=CEMZ
      GO TO 89
77    CONTINUE
      IF(I1-1)35,35,36
35    EMX(I,I1)=ALL/AL
      GO TO 27
36    EMX(I,I1)=(ALL/AL)+((ALR-ALL)/AL)*(ZK(I,I1)
     1 -ZK(I,I1-1))/W*BF
27    EMZ(I,I1)=W/WO
89    EPSX(I,I1)=ALPHA*HK(I,I1)*USTAR
      EPSZ(I,I1)=BETA*HK(I,I1)*USTAR
      WRITE(6,29)EMX(I,I1),EMZ(I,I1),EPSX(I,I1),EPSZ(I,I1)
29    FORMAT(5X,F6.4,5X,F6.4,5X,F7.4,5X,F7.5)
24    CONTINUE
4     CONTINUE
C------
C APPROXIMATE LINEARLY IN OC- DIRECTION
C------
      LL=IA2-1
      DO 200 I=1,NCS
      WV(I,1)=VK(I,1)
      WV(I,IA2)=VK(I,K1)
      WH(I,1)=HK(I,1)
      WH(I,IA2)=HK(I,K1)
      WP(I,1)=FHI(I,1)
      WP(I,IA2)=FHI(I,K1)
      WEMX(I,1)=EMX(I,1)
      WEMX(I,IA2)=EMX(I,K1)
      WEMZ(I,1)=EMZ(I,1)
      WEMZ(I,IA2)=EMZ(I,K1)
      WEPX(I,1)=EPSX(I,1)
      WEPX(I,IA2)=EPSX(I,K1)
      WEPZ(I,1)=EPSZ(I,1)
      WEPZ(I,IA2)=EPSZ(I,K1)
      DO 200 J1=2,LL
      DO 170 J=1,KK
      A1=SEGJ(J1)
      IF((SEGJ(J1) .GE. QCK(I,J)) .AND. (SEGJ(J1).LE.QCK(I,J+1))
     1) GO TO 169
      GO TO 170
169   WV(I,J1)=VFUN(VK(I,J),VK(I,J+1),DQ,A1,CCK(I,J))
      WH(I,J1)=VFUN(HK(I,J),HK(I,J+1),DQ,A1,QCK(I,J))
      WP(I,J1)=VFUN(FHI(I,J),FHI(I,J+1),DQ,A1,QCK(I,J))
      WEMX(I,J1)=VFUN(EMX(I,J),EMX(I,J+1),DQ,A1,QCK(I,J))
      WEMZ(I,J1)=VFUN(EMZ(I,J),EMZ(I,J+1),DQ,A1,QCK(I,J))
      WEPX(I,J1) =VFUN(EPSX(I,J),EPSX(I,J+1),DQ,A1,QCK(I,J))
      WEPZ(I,J1)  =VFUN(EPSZ(I,J),EPSZ(I,J+1),DQ,A1,QCK(I,J))
      J=KK
170   CONTINUE
200   CONTINUE
      GO TO 171
3423  CONTINUE
      WRITE(6,152)
152    FORMAT(10X,'WV',6X,'WH',6X,'  WP',6X,'WEMX',6X,'WEMZ',
     16X,'WEPX',6X,'WEPZ')
      DO 154 I=1,NCS
      DO 154 J=1,IA2
154    WRITE(6,183)I,J,WV(I,J),WH(I,J),WP(I,J),WEMX(I,J),
     1WEMZ(I,J),WEPX(I,J),WEPZ(I,J)
C------
C APPROXIMATE HORIZENTALLY
C------
171   CONTINUE
      LD=NCS-1
      LL=IA1-1
      DO 190 I=1,IA2
      VN(1,I)=WV(1,I)
      VN(IA1,I)=WV(NCS,I)
      HN(1,I)=WH(1,I)
      HN(IA1,I)=WH(NCS,I)
      FHIN(1,I)=WP(1,I)
      FHIN(IA1,I)=WP(NCS,I)
      EMXN(1,I)=WEMX(1,I)
      EMXN(IA1,I)=WEMX(NCS,I)
      EMZN(1,I)=WEMZ(1,I)
      EMZN(IA1,I)=WEMZ(NCS,I)
      EPSXN(1,I)=WEPX(1,I)
      EPSXN(IA1,I)=WEPX(NCS,I)
      EPSZN(1,I)=WEPZ(1,I)
      EPSZN(IA1,I)=WEPZ(NCS,I)
      DO 190 J1=2,LL
      DO 180 J=1,LD
      A1=SEGI(J1)
      IF((SEGI(J1).GE.XCR(J)).AND.(SEGI(J1).LE.XCR(J+1))
     1)     GO TO 269
      GO TO 180
269   A3=XCR(J+1)-XCR(J)
      VN(J1,I)=VFUN(  WV(J,I),WV(J+1,I),A3,A1,XCR(J))
      HN(J1,I)=VFUN(  WH(J,I),WH(J+1,I),A3,A1,XCR(J))
      FHIN(J1,I)=VFUN(WP(J,I),WP(J+1,I),A3,A1,XCR(J))
      EMXN(J1,I)=VFUN(WEMX(J,I),WEMX(J+1,I),A3,A1,XCR(J))
      EMZN(J1,I)=VFUN(WEMZ(J,I),WEMZ(J+1,I),A3,A1,XCR(J))
      EPSXN(J1,I)=VFUN(WEPX(J,I),WEPX(J+1,I),A3,A1,XCR(J))
      EPSZN(J1,I)=VFUN(WEPZ(J,I),WEPZ(J+1,I),A3,A1,XCR(J))
      J=LD
180   CONTINUE
190    CONTINUE
      GO TO 1181
4876  CONTINUE
```

```
      WRITE(6,181)
 181  FORMAT(1H1,'  I, J',4X,'VN',6X,'HN',6X,'FHIN',6X,
     1'EMXN',4X,'EMZN',6X,'EPSXN',5X,'EPSZN')
      DO 182 I=1,IA1
      DO 182 J=1,IA2
      WRITE(6,183)I,J,VN(I,J),HN(I,J),FHIN(I,J),
     1EMXN(I,J),EMZN(I,J),EPSXN(I,J),EPSZN(I,J)
 182  CONTINUE
 183  FORMAT(1X,I2,',',I2,1X,F7.4,2X,F7.4,1X,F7.4,3X,
     1F7.4,3X,F7.4,3X,F7.4,5X,F7.4)
 1181 CONTINUE
      RETURN
      END
C------------------------------------------------------------------
C THIS SUBROUTINE IS USED TO FIND THE VALUE OF THE FUNCTIONS   :
C FF1,FF2,FF3,FF4 AND FF5   IN P. D. E.                         :
C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
      SUBROUTINE VALUE1(NX,NY,IA1,IA2,IA3,QTOT)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION C(22,22),WORK(484)
      DIMENSION AK(10),H(10),SEGI(22),SEGJ(22)
      DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
      DIMENSION EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
      DIMENSION FF1(22,22),FF2(22,22),FF3(22,22),FF4(22,22),
     1        FF5(22,22),FF6(22,22),FF7(22,22)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,ELZERO,
     1VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN,FF1,FF2,FF3,FF4,FF5,FF6,
     1FF7
C------
      DO 1 I=1,IA1
      DO 1 J=1,IA2
      FF1(I,J)=VN(I,J)/EMXN(I,J)
      FF2(I,J)=EPSXN(I,J)/((EMXN(I,J)**2)*ELZERO*UZERO)
      FF3(I,J)=0.D0
      A1=ELZERO*(VN(I,J)**2)*(R*R)*(UZERO)/QTOT
      A2=A1*(FHIN(I,J)**2)*EPSZN(I,J)/QTOT
      FF4(I,J)=A2
      FF5(I,J)=0.D0
1     CONTINUE
      RETURN
      END
C------------------------------------------------------------------
C THIS SUBROUTINE IS USED TO FIND THE INITIAL CONDITIONS       :
C  UP-STREAM AT TIME STEP (T) SECOND.                           :
C------------------------------------------------------------------
      SUBROUTINE CINPUT(CO,T,TPEAK,IA1,IA2,RPKC,KIN,TAU)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION RPKC(22)
      DIMENSION C(22,22)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C
      IF(T .GT. TPEAK) GO TO 320
      GO TO(1,2,1,1),KIN
  1   DO 300 J=1,IA2
300   C(1,J)=(RPKC(J)/TPEAK)*T/CO
      RETURN
2     DO 3 J=1,IA2
3     C(1,J)=CO*(1.D0-DEXP(-TAU))
```

```
      RETURN
320   DO 340 J=1,IA2
340   C(1,J)=C(1,J)
      RETURN
      END
C-----------------------------------------------------------------------
C THIS SUBROUTINE IS USED TC CALCULATE THE ELEMENTAL MATRICES      :
C ...CMX0,CMC1,CMX2,   CMY0,CMY1,CMY2,                               :
C-----------------------------------------------------------------------
      SUBROUTINE ELMAT(NX,NY,IA1,IA2,IA3,I,J)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION C(22,22),WORK(484)
      DIMENSION AK(10),H(10),SEGI(22),SEGJ(22)
      DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
      DIMENSION EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
      DIMENSION FF1(22,22),FF2(22,22),FF3(22,22),FF4(22,22)
     1,        FF5(22,22),FF6(22,22),FF7(22,22)
      DIMENSION B(484),E1(484),AM(6724)
      DIMENSION CMX0(2,4),CMX1(2,4),CMX2(2,4),CMY0(2,4)
     1,        CMY1(2,4),CMY2(2,4)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,
     1ELZERO,VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN,FF1,FF2,FF3,FF4,
     1FF5,FF6,FF7,B,B1,AM,CMX0,CMX1,CMX2,CMY0,CMY1,CMY2
      ALPHA=(9.D0+4.D0*DSQRT(3.D0))/18.D0
      BETA=(3.D0+DSQRT(3.D0))/36.D0
      BETH=(3.D0-DSQRT(3.D0))/36.D0
      CMX0(1,1)=ALPHA
      CMX0(1,2)=H(I)*BETA
      CMX0(1,3)=1.D0-ALPHA
      CMX0(1,4)=-H(I)*BETH
      CMX0(2,1)=1.D0-ALPHA
      CMX0(2,2)=H(I)*BETH
      CMX0(2,3)=ALPHA
      CMX0(2,4)=-H(I)*BETA
C-----
      CMY0(1,1)=ALPHA
      CMY0(1,2)=AK(J)*BETA
      CMY0(1,3)=1.D0-ALPHA
      CMY0(1,4)=-AK(J)*BETH
      CMY0(2,1)=1.D0-ALPHA
      CMY0(2,2)=AK(J)*BETH
      CMY0(2,3)=ALPHA
      CMY0(2,4)=-AK(J)*BETA
C-----
      CMX1(1,1)=-1.D0/H(I)
      CMX1(1,2)=DSQRT(3.D0)/6.D0
      CMX1(1,3)=-CMX1(1,1)
      CMX1(1,4)=-CMX1(1,2)
      CMX1(2,1)=CMX1(1,1)
      CMX1(2,2)=CMX1(1,4)
      CMX1(2,3)=CMX1(1,3)
      CMX1(2,4)=CMX1(1,2)
C-----
      CMY1(1,1)=-1.D0/AK(J)
      CMY1(1,2)=DSQRT(3.D0)/6.D0
      CMY1(1,3)=-CMY1(1,1)
      CMY1(1,4)=-CMY1(1,2)
      CMY1(2,1)=CMY1(1,1)
      CMY1(2,2)=CMY1(1,4)
      CMY1(2,3)=CMY1(1,3)
      CMY1(2,4)=CMY1(1,2)
C-----
      ALPHA=2.D0*DSQRT(3.D0)
      BETA=DSQRT(3.D0)+1.D0
      BETH=DSQRT(3.D0)-1.D0
      CMX2(1,1)=-ALPHA/(H(I)**2)
      CMX2(1,2)=-BETA/H(I)
      CMX2(1,3)=-CMX2(1,1)
      CMX2(1,4)=-BETH/H(I)
      CMX2(2,1)=CMX2(1,3)
      CMX2(2,2)=-CMX2(1,4)
      CMX2(2,3)=CMX2(1,1)
      CMX2(2,4)=-CMX2(1,2)
C-----
      CMY2(1,1)=-ALPHA/(AK(J)**2)
      CMY2(1,2)=-BETA/AK(J)
      CMY2(1,3)=-CMY2(1,1)
      CMY2(1,4)=-BETH/AK(J)
      CMY2(2,1)=CMY2(1,3)
      CMY2(2,2)=-CMY2(1,4)
      CMY2(2,3)=CMY2(1,1)
      CMY2(2,4)=-CMY2(1,2)
      RETURN
      END
C-----------------------------------------------------------------
C THIS SUBROUTINE IS USED TO FIND THE VALUE OF THE ENTRIES:
C OBTAINED AT THE CORNER OF THE FIELD.                        :
C-----------------------------------------------------------------
      SUBROUTINE CORNER(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION C(22,22),WORK(484)
      DIMENSION AK(10),H(10),SEGI(22),SEGJ(22)
      DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
      DIMENSION EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
      DIMENSION FF1(22,22),FF2(22,22),FF3(22,22),FF4(22,22),
     1        FF5(22,22),FF6(22,22),FF7(22,22)
      DIMENSION B(484),B1(484),AM(6724)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,U2ERO,ELZERO,
     1VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN,FF1,FF2,FF3,FF4,FF5,FF6,
     1FF7,B,B1,AM
C-----
      LM=0
      LN=0
      A1=SEGI(IA1)
      A2=SEGJ(1)
      A3=SEGJ(IA2)
C-----FIRST CORNER
      DO 6 II=1,4
      LM=LM+1
      GO TO (1,2,3,4),II
1     I=1
      J=1
      WORK(LM)=C(1,1)
      GO TO 5
C-----SECOND CORNER
2     I=1
      J=NY+1
      WORK(LM)=C(1,IA2)
      GO TO 5
C-----THIRD CORNER
3       CONTINUE
      LM1=(2*(NX+1)-2)*IA2+2
      WORK(LM)=FV1(A1,A2,DTAU)
      GO TO 16
C-----FOURTH CORNER
4       CONTINUE
      LM1=(2*(NX+1)-2)*IA2+(2*(NY+1))
      WORK(LM)=FV1(A1,A3,CTAU)
      GO TO 16
5     CONTINUE
      LM1=(2*(I-1))*(2*(NY+1))+2*J-1
16    LN=LN+1
      IORD=IORD+1
      IAP(IORD)=LN
      AM(LN)=1.D0
      LOC(LN)=LM1
6     CONTINUE
      RETURN
      END
C-----------------------------------------------------------------
C THIS SUBROUTINE IS USED TO FIND THE COEFF. WHEN APPLYING:
C THE BOUNDARY CONDITION AT LEFT AND RIGHT BANK           :
C-----------------------------------------------------------------
      SUBROUTINE BOUNDX(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION C(22,22),WORK(484)
      DIMENSION AK(10),H(10),SEGI(22),SEGJ(22)
      DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
      DIMENSION EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
      DIMENSION FF1(22,22),FF2(22,22),FF3(22,22),FF4(22,22),
     1        FF5(22,22),FF6(22,22),FF7(22,22)
      DIMENSION B(484),B1(484),AM(6724)
      DIMENSION CMX0(2,4),CMX1(2,4),CMX2(2,4),CMY0(2,4),
     1        CMY1(2,4),CMY2(2,4)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,ELZERO,
     1VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN,FF1,FF2,FF3,FF4,FF5,FF6,
     1FF7,B,B1,AM,CMX0,CMX1,CMX2,CMY0,CMY1,CMY2
C------
      DO 3 NT=1,2
      GO TO (6,7),NT
C::::::::::::::::::------------RIGHT  BANK
6     J=1
      J1=1
      M1=1
      GO TO 8
C::::::::::::::::::------------LEFT  BANK
7     J=NY+1
      J1=NY
      M1=IA2
8     CONTINUE
      DO 3 I=1,NX
      A3=SEGJ(M1)
      A1=SEGI(2*I)
      A2=SEGI(2*I+1)
C-----CALCULATE THE ELEMENTAL MATRIX FOR ELEMENT(I,J)
      CALL ELMAT(NX,NY,IA1,IA2,IA3,I,J1)
      DO 3 JJ=1,2
      IORD=IORD+1
      IAP(IORD)=LN+1
      LM=LM+1
      DO 4 K=1,4
      LM1=(K-1)*IA2+2*J+4*(I-1)*(NY+1)
      LN=LN+1
      AM(LN)=CMX0(JJ,K)
      LOC(LN)=LM1
4     CONTINUE
      GO TO (9,10),JJ
9     WORK(LM)=FV2(A1,A3,DTAU)
      GO TO 3
10    WORK(LM)=FV2(A2,A3,DTAU)
3     CONTINUE
      RETURN
      END
C------------------------------------------------------------------
C THIS SUBROUTINE IS USED TO FIND THE COEFF.  OBTAINED          :
C WHEN  APPLYING THE CONDITIONS UP AND DOWN STREAM.             :
C------------------------------------------------------------------
      SUBROUTINE BOUNDY(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION C(22,22),WORK(484)
      DIMENSION AK(10),H(10),SEGI(22),SEGJ(22)
      DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
      DIMENSION EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
      DIMENSION FF1(22,22),FF2(22,22),FF3(22,22),FF4(22,22),
     1        FF5(22,22),FF6(22,22),FF7(22,22)
      DIMENSION B(484),B1(484),AM(6724)
      DIMENSION CMX0(2,4),CMX1(2,4),CMX2(2,4),CMY0(2,4),
     1        CMY1(2,4),CMY2(2,4)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,ELZERO,
     1VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN,FF1,FF2,FF3,FF4,FF5,FF6,
     1FF7,B,B1,AM,CMX0,CMX1,CMX2,CMY0,CMY1,CMY2
C------
      DO 2 NT=1,2
      GO TO (5,6),NT
C:::::::::::::::::::::::----UP-STREAM CONDITION
5     CONTINUE
      I1=1
      M1=1
      GO TO 7
C:::::::::::::::::::::::----DOWN-STREAM CONDITION
6     CONTINUE
      I1=NX
      M1=IA1
7     CONTINUE
      AA1=SEGI(M1)
      DO 2 J=1,NY
      AA2=SEGJ(2*J)
      AA3=SEGJ(2*J+1)
      CALL ELMAT(NX,NY,IA1,IA2,IA3,I1,J)
      DO 2 JJ=1,2
      IORD=IORD+1
      IAP(IORD)=LN+1
      LM=LM+1
      DO 3 K=1,4
      A1=CMY0(JJ,K)
      GO TO (8,9),NT
```

```
8        LM1=2*(J-1)+K
        IF(K .GT. 1)GO TO 10
        GO TO (13,15),JJ
13      WORK(LM)=C(1,2*J)
        GO TO 10
15      WORK(LM)=C(1,2*J+1)
        GO TO 10
9       CONTINUE
        LM1=(2*NX+1)*IA2+2*(J-1)+K
        IF(K .GT. 1)GO TC 10
        GO TO (16,17),JJ
16      WORK(LM)=FV1(AA1,AA2,DTAU)
        GO TO 10
17      WORK(LM)=FV1(AA1,AA3,DTAU)
10      LN=LN+1
        AM(LN)=A1
        LOC(LN)=LM1
3       CONTINUE
2       CONTINUE
        RETURN
        END
C--------------------------------------------------------------
CTHIS SUBROUTINE IS USED TO FIND THE COEFF.  WHEN        :
C  APPLYING THE P. D. E. AT ENTERIOR POINTS.              :
C--------------------------------------------------------------
        SUBROUTINE AINTP(NX,NY,IA1,IA2,IA3,DTAU,LM,LN,IORD)
        IMPLICIT REAL*8(A-H,O-Z)
        INTEGER*2  LOC,IAP,IRR,ICL,IC
        DIMENSION C(22,22),WORK(484)
        DIMENSION AK(10),H(10),SEGI(22),SEGJ(22),AA(4)
        DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
       1,      EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
        DIMENSION FF1(22,22),FF2(22,22),FF3(22,22),FF4(22,22),
       1        FF5(22,22),FF6(22,22),FF7(22,22)
        DIMENSION B(484),B1(484),AM(6724)
        DIMENSION CMX0(2,4),CMX1(2,4),CMX2(2,4),CMY0(2,4),
       1        CMY1(2,4),CMY2(2,4),CC(4,16),CC1(4,16)
        COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
        COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,ELZERO
       1,VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN,
       1FF1,FF2,FF3,FF4,FF5,FF6,FF7,B,B1,AM,CMX0,CMX1,CMX2,CMY0,
       1CMY1,CMY2,CC,CC1
C-----
        DO 1 J=1,NY
        DO 1 I=1,NX
        CALL ELMAT(NX,NY,IA1,IA2,IA3,I,J)
C::::::ICODE-CODE TO DEFFERENTIATE BETWEEN THE COEFF. OBTAINED
C::::::          FOR MATRIX  AM    OR  (D)
C:::::: ICODE=1---FOR    (AM)
C::::::        =2---FOR    (D )
C------
        ICODE=1
        DO  3 I3=1,4
        DO  3 J3=1,16
3        CC(I3,J3)=0.D0
        CALL SUBC(I,J,NX,NY,IA1,IA2,IA3,ICODE,DTAU)
        ICODE=2
        DO 4 I4=1,4
        AA(I4)=0.D0
        DO 4 J4=1,16
4        CC1(I4,J4)=0.D0
```

```
       CALL SUBC(I,J,NX,NY,IA1,IA2,IA3,ICCCE,CTAU)
       ICN=LM
       DO 2 KT=1,4
       IORD=IORD+1
       IAP(IORD)=LN+1
       IH=0
       LM=LM+1
       DO 2 II=1,4
       LM1=2*(II-1)*(NY+1)+2*(J-1)+4*(NY+1)*(I-1)
       DO 2 JJ=1,4
       IH=IH+1
       LM1=LM1+1
       LN=LN+1
       AM(LN)=CC(KT,IH)
       LOC(LN)=LM1
       AA(KT)=AA(KT)+CC1(KT,IH)*B(LM1)
 2     CONTINUE
       WORK(ICN+1)=2.D0*FF7(2*I,2*J)+AA(1)
       WORK(ICN+2)=2.D0*FF7(2*I,2*J+1)+AA(2)
       WORK(ICN+3)=2.D0*FF7(2*I+1,2*J)+AA(3)
       WORK(ICN+4)=2.D0*FF7(2*I+1,2*J+1)+AA(4)
 1     CONTINUE
       RETURN
       END
C----------------------------------------------------------------
C THIS SUBROUTINE IS USED TC PERFORM DIFFERENT CPERATIONS:
C. PERFORMING FIRST-TENSOR PRODUCT.EVALUATE ENTRIES.           :
C----------------------------------------------------------------
       SUBROUTINE SUBC(I,J,NX,NY,IA1,IA2,IA3,ICCCE,CTAU)
       IMPLICIT REAL*8(A-H,O-Z)
       INTEGER*2  LOC,IAP,IRR,ICL,IC
       DIMENSION AKX(2,4),AKY(2,4),AKE(4,16),CONST(4)
       DIMENSION C(22,22),WORK(484)
       DIMENSION AK(10),H(10),SEGI(22),SEGJ(22)
       DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
      1,EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
       DIMENSION FF1(22,22),FF2(22,22),FF3(22,22),FF4(22,22)
      1       ,FF5(22,22),FF6(22,22),FF7(22,22)
       DIMENSION B(484),B1(484),AM(6724)
       DIMENSION CMX0(2,4),CMX1(2,4),CMX2(2,4),CMY0(2,4),CMY1(2,4),
      1       CMY2(2,4),CC(4,16),CC1(4,16)
       COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
       COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,ELZERO,
      1VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN,FF1,FF2,FF3,FF4,FF5,FF6,
      1FF7,B,B1,AM,CMX0,CMX1,CMX2,CMY0,CMY1,CMY2,CC,CC1
       DO 3 JJ=1,6
       GO TO(4,5,6,7,8,9),JJ
 4     CONST(1)=-FF1(2*I,2*J)+FF3(2*I,2*J)
       CONST(2)=-FF1(2*I,2*J+1)+FF3(2*I,2*J+1)
       CONST(3)=-FF1(2*I+1,2*J)+FF3(2*I+1,2*J)
       CONST(4)=-FF1(2*I+1,2*J+1)+FF3(2*I+1,2*J+1)
       GO TO 2
 5     CONST(1)=FF2(2*I,2*J)
       CONST(2)=FF2(2*I,2*J+1)
       CONST(3)=FF2(2*I+1,2*J)
       CONST(4)=FF2(2*I+1,2*J+1)
       GO TO 2
 6     CONST(1)=FF4(2*I,2*J)
       CONST(2)=FF4(2*I,2*J+1)
       CONST(3)=FF4(2*I+1,2*J)
       CONST(4)=FF4(2*I+1,2*J+1)
      GO TO 2
7     CONST(1)=FF5(2*I,2*J)
      CONST(2)=FF5(2*I,2*J+1)
      CONST(3)=FF5(2*I+1,2*J)
      CONST(4)=FF5(2*I+1,2*J+1)
      GO TO 2
8     CONST(1)=FF6(2*I,2*J)
      CONST(2)=FF6(2*I,2*J+1)
      CONST(3)=FF6(2*I+1,2*J)
      CONST(4)=FF6(2*I+1,2*J+1)
      GO TO 2
9     DO 10 M=1,4
10    CONST(M)=2.D0/DTAU
2     CONTINUE
C-----------------------------
      DO 13 L1=1,2
      DO 13 L2=1,4
      GO TO (14,15,16,17,18,18),JJ
14    AKX(L1,L2)=CMX1(L1,L2)
      AKY(L1,L2)=CMY0(L1,L2)
      GO TO 13
15    AKX(L1,L2)=CMX2(L1,L2)
      AKY(L1,L2)=CMY0(L1,L2)
      GO TO 13
16    AKX(L1,L2)=CMX0(L1,L2)
      AKY(L1,L2)=CMY2(L1,L2)
      GO TO 13
17    AKX(L1,L2)=CMX0(L1,L2)
      AKY(L1,L2)=CMY1(L1,L2)
      GO TO 13
18    AKX(L1,L2)=CMX0(L1,L2)
      AKY(L1,L2)=CMY0(L1,L2)
13    CONTINUE
      CALL TENSRP(AKX,AKY,AKE,ICODE,CONST)
      CALL AADD(NX,NY,IA1,IA2,IA3,ICODE,AKE,JJ)
3     CONTINUE
      RETURN
      END
C--------------------------------------------------------
C THIS SUBROUTINE USED TO FIND THE TENSOR PRODUCT       :
C OF TWO MATRICES AND BUILD UP THE 4X16 ELEMENTAL       :
C MATRICES      CC     AND     CC1.                     :
C--------------------------------------------------------
      SUBROUTINE TENSRP(AKX,AKY,AKE,ICODE,CONST)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION AKX(2,4),AKY(2,4),AKE(4,16),CONST(4)
      IM1=0
      DO 1  I1=1,2
      DO 1  IM=1,2
      IM1=IM1+1
      IH=0
      DO 1  J1=1,4
      L=0
      DO 1 K=1,4
      IH=IH+1
      L=L+1
1     AKE(IM1,IH)=AKX(I1,J1)*AKY(IM,L)
C::::::
C MULTIPLY BY A VECTOR CONST
C::::::
      DO 2 I1=1,4
      DO 2 J1=1,16
      AKE(I1,J1)=AKE(I1,J1)*CONST(I1)
2     CONTINUE
      RETURN
      END
C-----------------------------------------------------------
C THIS SUBROUTINE IS USED TO MULTIPLY TWO MATRICES   :
C AND STORE THE RESULT EITHER IN CC  OR  CC1,        :
C  DEPENDING ON THE VALUE OF ICODE.                  :
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
      SUBROUTINE AACD(NX,NY,IA1,IA2,IA3,ICODE,AKE,JJ)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IAP,IRR,ICL,IC
      DIMENSION AKE(4,16)
      DIMENSION C(22,22),WORK(484)
      DIMENSION AK(10),H(10),SEGI(22),SEGJ(22)
      DIMENSION VN(22,22),HN(22,22),FHIN(22,22),EMXN(22,22)
      DIMENSION EMZN(22,22),EPSXN(22,22),EPSZN(22,22)
      DIMENSION FF1(22,22),FF2(22,22),FF3(22,22),FF4(22,22)
     1 ,      FF5(22,22),FF6(22,22),FF7(22,22)
      DIMENSION B(484),E1(484),AM(6724)
      DIMENSION CMX0(2,4),CMX1(2,4),CMX2(2,4),CMY0(2,4),CMY1(2,4),
     1        CMY2(2,4),CC(4,16),CC1(4,16)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,ELZERO,
     1VN,HN,FHIN,EMXN,EMZN,EPSXN,EPSZN,FF1,FF2,FF3,FF4,FF5,FF6,
     1FF7,B,E1,AM,CMX0,CMX1,CMX2,CMY0,CMY1,CMY2,CC,CC1
C-------
      DO 1 I1=1,4
      DO 1 J1=1,16
      GO TO (4,3),ICODE
3     CC1(I1,J1)=CC1(I1,J1)+AKE(I1,J1)
      GO TO 1
4      CONTINUE
      IF(JJ .GT. 5) GO TO 6
      CC(I1,J1)=CC(I1,J1)-AKE(I1,J1)
      GO TO 1
6     CC(I1,J1)=CC(I1,J1)+AKE(I1,J1)
1     CONTINUE
      RETURN
      END
C-----------------------------------------------------------
C THIS FUNCTION IS USED TO ASSIGN VALUES AT DOWN-    :
C  STREAM BOUNDARY.                                  :
C-----------------------------------------------------------
      FUNCTION   FV1(A1,A2,TM)
      IMPLICIT REAL*8(A-H,O-Z)
      FV1=0.D0
      RETURN
      END
C-----------------------------------------------------------
C THIS FUNCTION USED TO ASSIGN VALUES   AT BOUNDARY :
C   X-  LEFT  AND RIGHT BANK.                       :
C-----------------------------------------------------------
      FUNCTION FV2(A1,A2,TM)
      IMPLICIT REAL*8(A-H,O-Z)
      FV2=0.D0
      RETURN
      END
C-----                                       ------
```

```
C THIS FUNCTION IS USED TO FIND THE VALUE   :
C OF LAMDA1 AS A FUNCTICN OF X,Y AND TIME   :
C-----                                   -----
      FUNCTION ALAM1(A1,A2,TAU,ELZ,UZR,F6)
      IMPLICIT REAL*8(A-H,O-Z)
      ALAM1=F6
      RETURN
      END
C-----                                     -----
C THIS FUNCTION IS USED TO FIND THE VALUE OF  :
C LAMDA2 AS A FUNCTION OF X,Y AND TIME        :
C-----                                     -----
      FUNCTION ALAM2(A1,A2,TAU,ELZ,LZR,CO,F7)
      IMPLICIT REAL*8(A-H,O-Z)
      ALAM2=F7
      RETURN
      END
      SUBROUTINE GELLM(N,MAX,IERR,ITEMP,RTEMP)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2 LOC,IAP,IRR,ICL,IC,ITEMP
      DIMENSION C(22,22),WORK(484),AK(10),H(10),SEGI(22),SEGJ(22),
     1AW(14,22,22),B(484),B1(484),AM(6724),ITEMP(16418),RTEMP(16000)
      COMMON LOC(6724),IAP(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,LZERO,ELZERO,AW,
     1B,B1,AM
      IY=1
      IU1=IY+N
      IP=1
      IU2=IP+N+1
      JU=IU2+N+1
      CALL GAUSPV(N,MAX,RTEMP(IY),ITEMP(IP),ITEMP(IU2),
     1ITEMP(JU),RTEMP(IU1),IERR)
      RETURN
      END
      SUBROUTINE GAUSPV(N,MAX,Y,IP,IU,JU,U,IERR)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2   JA,IA,IRR,ICL,IC
      INTEGER*2   IP,IU,JU
      DIMENSION C(22,22),WORK(484),AK(10),H(10),SEGI(22),SEGJ(22),
     1AW(14,22,22),B(484),B1(484),AM(6724),Y(1),IP(1),IU(1),
     1JU(1),U(1)
      COMMON JA(6724),IA(488),IRR(484),ICL(484),IC(484)
      COMMON C,WORK,AK,H,SEGI,SEGJ,USTAR,BF,R,UZERO,ELZERO,
     1AW,B,B1,AM
      IF(N .EQ. 0)GO TO 1001
      ONE=1.D0
      ZERO=0.D0
      DO 10 J=1,N
10    B1(J)=ZERC
      IU(1)=1
      JUPTR=0
      DO 170 K=1,N
      IP(N+1)=N+1
      KKV=IRR(K)
      JMIN=IA(KKV)
      JMAX=IA(KKV+1) - 1
      IF(JMIN .GT. JMAX)GO TO 1002
      J=JMAX
20    JAJ=JA(J)
      JVV=IC(JAJ)
      B1(JVV)=AM(J)
```

```
       IIK=N+1
 30    IK=IIK
       IIK=IP(IK)
       IF(IIK-JVV)30,1003,40
 40    IP(JVV)=IIK
       IP(IK)=JVV
       J=J-1
       IF(J .GE. JMIN)GO TO 20
       IVI=N+1
       YK=WORK(KKV)
 50    IVI=IP(IVI)
       IF(IVI .GE. K)GO TO 110
       ALKI=-B1(IVI)
       B1(IVI)=ZERO
       YK=YK+ALKI*Y(IVI)
       IIK=IVI
       JMIN=IU(IVI)
       JMAX=IU(IVI+1)-1
       IF(JMIN .GT. JMAX)GO TO 50
       DO 100 J=JMIN,JMAX
       JUJ=JU(J)
       JVV=IC(JUJ)
       IF(B1(JVV).NE.ZERO)GO TO 90
       IF(JVV-IIK)60,90,70
 60    IIK=IVI
 70    IK=IIK
       IIK=IP(IK)
       IF(IIK-JVV)70,90,80
 80    IP(JVV)=IIK
       IP(IK)=JVV
       IIK=JVV
 90    B1(JVV)=B1(JVV)+ALKI*U(J)
 100   CONTINUE
       GO TO 50
 110   IF(IVI.GT.N)GO TO 1004
       XPVMAX=DABS(B1(IVI))
       MAXC=IVI
       NZCNT=0
       IPV=IVI
 120   IVR=IPV
       IPV=IP(IPV)
       IF(IPV.GT.N)GO TO 130
       NZCNT=NZCNT+1
       XPV=DABS(B1(IPV))
       IF(XPV .LE. XPVMAX)GO TO 120
       XPVMAX=XPV
       MAXC=IPV
       MAXCL=IVR
       GO TO 120
 130   IF(XPVMAX.EQ.ZERO)GO TO 1004
       IF(IVI .EQ. K)GO TO 140
       IF(IVI .EQ.MAXC)GO TO 140
       IP(MAXCL)=IP(MAXC)
       GO TO 150
 140   IVI=IP(IVI)
 150   CONTINUE
       DK=ONE/B1(MAXC)
       B1(MAXC)=B1(K)
       I=ICL(K)
       ICL(K)=ICL(MAXC)
       ICL(MAXC)=I
```

```
      ICK=ICL(K)
      IC(ICK)=K
      IC(I)=MAXC
      B1(K)=ZERO
      Y(K)=YK*DK
      IU(K+1)=IU(K)+NZCNT
      IF(IU(K+1).GT.MAX+1)GO TO 1005
      IF(IVI .GT. N)GO TC 170
      J=IVI
160   JUPTR=JUPTR+1
      JU(JUPTR)=ICL(J)
      U(JUPTR)=B1(J)*DK
      B1(J)=ZERO
      J=IP(J)
      IF(J .LE. N) GO TO 160
170   CONTINUE
      K=N
      DO 200 I=1,N
      YK=Y(K)
      JMIN=IU(K)
      JMAX=IU(K+1)-1
      IF(JMIN .GT. JMAX)GO TO 190
      DO 180 J=JMIN,JMAX
      JUJ=JU(J)
      JUJ=IC(JUJ)
      YK=YK - U(J)*Y(JUJ)
180   CONTINUE
190   Y(K)=YK
      ICK=ICL(K)
      B1(ICK)=YK
      K=K-1
200   CONTINUE
      IERR=IU(N+1)-IU(1)
      RETURN
1001  IERR=0
      RETURN
1002  IERR=-K
      RETURN
1003  IERR=-(N+K)
      RETURN
1004  IERR=-(2*N+K)
      RETURN
1005  IERR=-(3*N+K)
      RETURN
      END
      SUBROUTINE PREORD(N)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER*2  LOC,IA,IRR,ICL,IC
      COMMON LOC(6724),IA(488),IRR(484),ICL(484),IC(484)
      DO 1 I=1,N
      IRR(I)=I
      ICL(I)=I
1     IC(I)=I
      DO 5 I=1,N
5     ICL(I)=0
      DO 10 K=1,N
      KDEG=IA(K+1)-IA(K)
      IF(KDEG .EQ. 0)KDEG=KDEG+1
      IC(K)=ICL(KDEG)
      ICL(KDEG)=K
10    CONTINUE
```

```
       I=0
       DO 30 J=1,N
       IF(ICL(J) .EQ. 0)GO TO 30
       K=ICL(J)
20     I=I+1
       IRR(I)=K
       K=IC(K)
       IF(K .GT. 0)GO TO 20
30     CONTINUE
       DO 40 I=1,N
       ICL(I)=I
       IC(I)=I
40     CONTINUE
       RETURN
       END
/*
//GO.SYSIN DD *
```