UFL/COEL-88/008

A TWO-DIMENSIONAL FINITE-DIFFERENCE MODEL FOR MOVING BOUNDARY HYDRODYNAMIC PROBLEMS

by

Yuming Liu

Thesis

1988

COASTAL & OCEANOGRAPHIC ENGINEERING DEPARTMENT

University of Florida • Gainesville, Florida 32611

UFL/COEL–88/008

A TWO–DIMENSIONAL FINITE–DIFFERENCE MODEL FOR MOVING BOUNDARY HYDRODYNAMIC PROBLEMS

by

Yuming Liu

Thesis

1988

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation and gratitude to my advisor Dr. Y. Peter Sheng, Professor of Coastal and Oceanographic Engineering, for his continuous guidance and encourgement throughout this study. I would also like to extend my thanks and appreciation to my thesis committee members, Dr. James T. Kirby, Associate Professor of Coastal and Oceanographic Engineering, and Dr. D. Max Sheppard, Professor of Coastal and Oceanographic Engineering, for their patience in reviewing this paper.

I would like to thank Dr. Yixin Yan, Dr. Tienshun Wu, Dr. Weichi Wang, Dr. Li-Hwa Lin, and Steven Peene for their help and suggestions in this study.

Finally, I would like to express my deepest appreciation and gratitude to my financial sponsor, K. C. Wong Education Foundation Ltd. of Hong Kong, for offering me the opportunity to study at the University of Florida.

TABLE OF CONTENTS

LIST OF FIGURES

Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Science

A TWO-DIMENSIONAL FINITE-DIFFERENCE MODEL FOR MOVING BOUNDARY HYDRODYNAMIC PROBLEMS

By

YUMING LIU

December 1988

Chairman: Dr. Y. Peter Sheng
Major Department: Coastal and Oceanographic Engineering

To predict the hydrodynamics of lakes, estuaries and shallow seas, a two dimensional numerical model is developed using the method of fractional steps. The governing equations, i.e., the vertically integrated Navier-Stokes equations of fluid motion, are solved through three steps: advection, diffusion and propagation. The characteristics method is used to solve the advection, the alternating direction implicit method is applied to compute the diffusion, and the conjugate gradient iterative method is employed to calculate the propagation. Two ways to simulate the moving boundary problem are studied. The first method is based on the weir formulation. The second method is based on the assumption that a thin water layer exists over the entire dry region at all times. A number of analytical solutions are used to validate the model. The model is also applied to simulate the wind driven circulation in Lake Okeechobee, Florida.

CHAPTER 1
INTRODUCTION

In the past two decades, numerical modeling has been widely applied to the study of the hydrodynamics of lakes, estuaries, coastal regions, etc. Many numerical models (Reid and Bodine, 1968, Leendertse, 1970, Yeh and Chou, 1979, etc.) have been developed from the shallow water equations, i.e. vertically integrated Navier-Stokes equations of fluid motion, using the finite difference technique.

Many numerical schemes have been proposed to solve the shallow water equations in the development of numerical models. They all have advantages and disadvantages. The explicit scheme is computationally simple, but the time step must be sufficiently small such that the Courant number is less than 1 (Smith, 1969) in order to attain numerical stability. The implicit scheme does not have this limitation, but it requires solving the matrix equations. For two- and three-dimensional flows, it is very difficult to overcome the computational difficulties resulting from the sheer size of the matrices. The alternating direction implicit method, or ADI method, avoids solving complex matrix equations, but can obtain accurate solutions only for Courant numbers less than 5 (Weare, 1979). Furthermore the ADI method is not applicable to three-dimensional problems (Yanenko, 1971). The method of fractional steps developed by Yanenko (1971), on the other hand is known to be an effective method for solving complicated multi-dimensional problems in several variables. In this scheme, the computation from one time level to the next is divided into a series of intermediate steps. For each intermediate step, the computational procedure is relatively simple, an exact solution can be obtained in some cases and the time step can be quite large. However, this scheme still has the disadvantage

that its consistency has not completely been justified theoretically. Nevertheless, it has been used to solve the shallow water equations (Benque et al., 1982).

In the development of numerical hydrodynamic models, the treatment of the shoreline boundary is very important. Most existing numerical hydrodynamic models were developed based on the assumption of a fixed boundary with a vertical wall located at the shoreline defined by the mean water depth. However, the shoreline boundary can actually move with time in such problems as wave runup on a beach and coastal flooding into a dry coastal region due to tides or storm surges. Some researchers have tried to simulate this problem numerically. Reid and Bodine (1968) considered the motion of the shoreline according to the water elevation and used the empirical weir formulation to compute the velocity in which water flows into or out of the dry land region. The disadvantage of this method is that the empirical coefficients in the weir formulation are very site-dependent. Yeh and Chou (1979) treated the shoreline boundary as a discrete moving boundary, but the impulsive motion of the boundary could introduce serious numerical problems. Lynch and Gray (1980) simulated the shoreline boundary as a continuous moving boundary using continuous grid deformation. This method, however, can not be applied to problems with complex topography.

In this study, a two-dimensional finite-difference numerical hydrodynamic model is developed from the shallow water equations by using the method of fractional steps. Two ways to simulate the moving boundary problem are studied. For some simplified flow situations, analytical solutions are compared with numerical solutions to verify the consistency of the fractional step method and the accuracy of the numerical model. The numerical model is used to investigate the wind driven circulation in Lake Okeechobee, Florida.

In Chapter 2, a careful derivation of the numerical model from the shallow water equations will be presented. The computational mesh consists of rectangular

cells with variable sizes. The governing equations are solved through three steps: advection, diffusion and propagation. The characteristics method is used in the advection step, the ADI method is applied to the diffusion step, and the conjugate gradient method is applied to the propagation step.

In Chapter 3, linear and nonlinear analytical solutions to the one- dimensional long wave propagation are developed and used to verify the accuracy of the numerical model. Theoretical solutions of tidal responses inside a rectangular basin with Coriolis effect and bottom friction obtained by Rahman (1981) will be used to test the ability of the model to simulate two-dimensional problems.

In Chapter 4, two ways to simulate the moving boundary problem are discussed in detail. One way is similar to that used by Reid and Bodine (1968). Another way is to include the dry land region into the computational domain by assuming a thin water layer on the dry land region at all times. The theoretical solutions for wave propagation on a linearly sloping beach developed by Carrier and Greenspan (1958) are then presented and compared to the numerical solutions to verify the second method for the moving boundary problem.

In Chapter 5, the wind driven circulation in Lake Okeechobee, Florida, including the effects of the Coriolis force and a moving boundary, is investigated numerically.

In Chapter 6, conclusions are drawn and suggestions are made towards further studies on numerical simulation of the moving boundary problem.

CHAPTER 2
DEVELOPMENT OF A TWO-DIMENSIONAL TIDAL CIRCULATION MODEL

In this Chapter, a two-dimensional implicit tidal circulation model is developed using the method of fractional steps. The vertically-integrated momentum and continuity equations, which govern the two-dimensional tide-generated currents, are solved through three steps which include an advection step, a diffusion step and a propagation step. Momentum advection is solved using the method of characteristics. An alternating direction implicit (ADI) method is applied to calculate momentum diffusion. The wave propagation is calculated using the conjugate gradient method.

2.1 Governing Equations

The mathematical equations describing tidal flow in shallow water can be obtained by vertically integrating the three-dimensional Navier-Stokes equations of fluid motion. Generally, it is assumed that the density of water over the depth is constant and the vertical pressure variation is hydrostatic, thus leading to the following continuity and momentum equations in a right-handed Cartisian coordinate system shown in the Fig. 2.1

$$\frac{\partial Z}{\partial t}+\frac{\partial U}{\partial x}+\frac{\partial V}{\partial y}=0 \tag{2.1}$$

$$\frac{\partial U}{\partial t}+\frac{\partial uU}{\partial x}+\frac{\partial vU}{\partial y}+gh\frac{\partial Z}{\partial x}-\Omega V+\frac{\tau_{bx}-\tau_{sx}}{\rho}-[\frac{\partial}{\partial x}(K\frac{\partial U}{\partial x})+\frac{\partial}{\partial y}(K\frac{\partial U}{\partial y})]=0 \tag{2.2}$$

$$\frac{\partial V}{\partial t}+\frac{\partial uV}{\partial x}+\frac{\partial vV}{\partial y}+gh\frac{\partial Z}{\partial y}+\Omega U+\frac{\tau_{by}-\tau_{sy}}{\rho}-[\frac{\partial}{\partial x}(K\frac{\partial V}{\partial x})+\frac{\partial}{\partial y}(K\frac{\partial V}{\partial y})]=0 \tag{2.3}$$

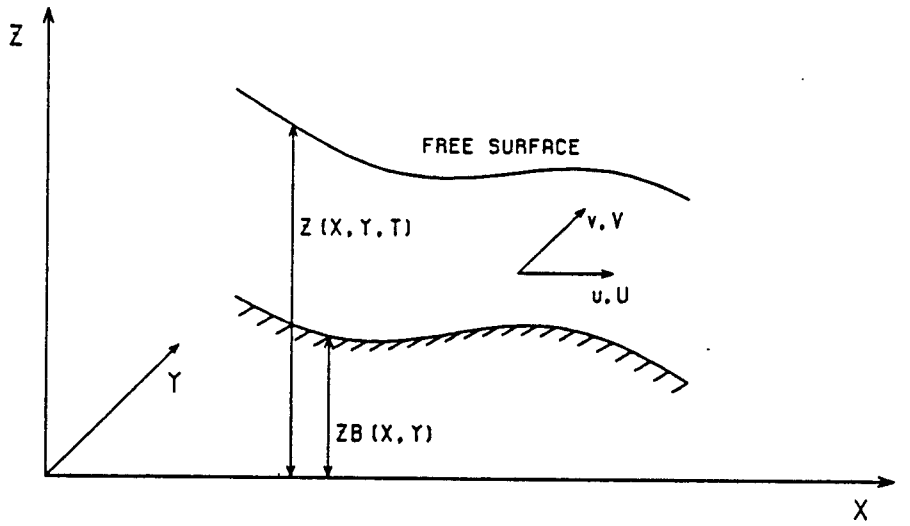


Figure 2.1: Definition sketch for tidal equations

where t is time, x and y are the spatial coordinates, $Z(x,y,t)$ is the free surface elevation, $U(x,y,t)$ and $V(x,y,t)$ are the unit-width discharges in the x- and y-directions, respectively, $u(x,y,t)$ and $v(x,y,t)$ are the vertically-averaged velocities corrresponding to $U(x,y,t)$ and $V(x,y,t)$, $h(x,y,t)$ is the water depth, Ω is the Coriolis acceleration parameter, g is the gravitational acceleration, ρ is the density of water, K is the horizontal turbulent diffusion coefficient, τ_{sx} and τ_{sy} are the wind shear stresses in the x- and y-directions, respectively, and τ_{bx} and τ_{by} are the bottom shear stresses in the x- and y-directions. τ_{bx} and τ_{by} can be expressed as

$$\tau_{bx} = \frac{\rho g U \sqrt{U^2 + V^2}}{C^2 h^2} \tag{2.4}$$

and

$$\tau_{by} = \frac{\rho g V \sqrt{U^2 + V^2}}{C^2 h^2} \tag{2.5}$$

where C is the Chezy coefficient which is

$$C = 8.21 \frac{R^{1/6}}{n} \qquad cm^{1/2}/sec. \tag{2.6}$$

or

$$C = 1.49\frac{R^{1/6}}{n} \qquad ft^{1/2}/sec. \tag{2.7}$$

where R is the hydraulic radius and n is the Manning coefficient.

Obviously we have

$$\left.\begin{array}{c} U = u\,h \\ V = v\,h \\ h = Z - Z_b \end{array}\right\} \tag{2.8}$$

where Z_b is the bottom bed elevation of an estuary which is only a function of x and y. Using these relationships and Eq. (2.1), the nonlinear terms in Eqs. (2.2) and (2.3) can be expressed as

$$\frac{\partial uU}{\partial x} + \frac{\partial vU}{\partial y} = h(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}) - u\frac{\partial Z}{\partial t} \tag{2.9}$$

and

$$\frac{\partial uV}{\partial x} + \frac{\partial vV}{\partial y} = h(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}) - v\frac{\partial Z}{\partial t} \tag{2.10}$$

Substituting them into Eqs. (2.2) and (2.3), we obtain

$$\frac{\partial U}{\partial t} + h(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}) - u\frac{\partial Z}{\partial t} + gh\frac{\partial Z}{\partial x} - \Omega V + \frac{\tau_{bx} - \tau_{sx}}{\rho} - [\frac{\partial(K\frac{\partial U}{\partial x})}{\partial x} + \frac{\partial(K\frac{\partial U}{\partial y})}{\partial y}] = 0 \tag{2.11}$$

$$\frac{\partial V}{\partial t} + h(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}) - v\frac{\partial Z}{\partial t} + gh\frac{\partial Z}{\partial y} + \Omega U + \frac{\tau_{by} - \tau_{sy}}{\rho} - [\frac{\partial(K\frac{\partial V}{\partial x})}{\partial x} + \frac{\partial(K\frac{\partial V}{\partial y})}{\partial y}] = 0 \tag{2.12}$$

Using the method of fractional steps, Eqs. (2.1), (2.11) and (2.12) can be solved through three steps which are called advection step, diffusion step and propagation step (Benque et al., 1982). In order to represent working equations at each step, we use the subscript n to denote the model variables at time $n\Delta t$ and n+1 for the model variables at time $(n+1)\Delta t$.

The working equations for the advection step are as follows:

$$\frac{u^{n+1/3} - u^n}{\Delta t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = 0 \tag{2.13}$$

$$\frac{v^{n+1/3}-v^n}{\Delta t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}=0 \tag{2.14}$$

$$U^{n+1/3}=u^{n+1/3}h^n \tag{2.15}$$

$$V^{n+1/3}=v^{n+1/3}h^n \tag{2.16}$$

where the subscript $n+1/3$ is a symbolic used to denote the result at time $(n+1)\Delta t$ due to the advection step alone.

The working equations for the computation of the diffusion step are

$$\frac{U^{n+2/3}-U^{n+1/3}}{\Delta t}-\Omega V-[\frac{\partial}{\partial x}(K\frac{\partial U}{\partial x})+\frac{\partial}{\partial y}(K\frac{\partial U}{\partial y})]=0 \tag{2.17}$$

$$\frac{V^{n+2/3}-V^{n+1/3}}{\Delta t}+\Omega U-[\frac{\partial}{\partial x}(K\frac{\partial V}{\partial x})+\frac{\partial}{\partial y}(K\frac{\partial V}{\partial y})]=0 \tag{2.18}$$

where the subscript $n+2/3$ denotes the result after the diffusion step.

The working equations for the propagation step are as follows:

$$\frac{Z^{n+1}-Z^n}{\Delta t}+\frac{\partial U}{\partial x}+\frac{\partial V}{\partial y}=0 \tag{2.19}$$

$$\frac{U^{n+1}-U^{n+2/3}}{\Delta t}-u\frac{\partial Z}{\partial t}+gh\frac{\partial Z}{\partial x}+\frac{\tau_{bx}-\tau_{sx}}{\rho}=0 \tag{2.20}$$

$$\frac{V^{n+1}-V^{n+2/3}}{\Delta t}-v\frac{\partial Z}{\partial t}+gh\frac{\partial Z}{\partial y}+\frac{\tau_{by}-\tau_{sy}}{\rho}=0 \tag{2.21}$$

The terms $u\frac{\partial Z}{\partial t}$ and $v\frac{\partial Z}{\partial t}$ come from the nonlinear terms. Compared with other terms in Eqs. (2.20) and (2.21), their values are small and could be neglected in the propagation step.

2.2 Numerical Scheme

Just like the splitting of a single time step into three steps as shown above, three steps could be further split into two directional steps. For example, the advection step can be treated with the following two steps:

x-advection

$$\frac{u^{n+1/6}-u^n}{\Delta t}+u^n\frac{\partial u^{n+1/6}}{\partial x}=0 \tag{2.22}$$

$$\frac{v^{n+1/6}-v^n}{\Delta t}+u^n\frac{\partial v^{n+1/6}}{\partial x}=0 \tag{2.23}$$

y-advection

$$\frac{u^{n+1/3}-u^{n+1/6}}{\Delta t}+v^{n+1/6}\frac{\partial u^{n+1/3}}{\partial y}=0 \tag{2.24}$$

$$\frac{v^{n+1/3}-v^{n+1/6}}{\Delta t}+v^{n+1/6}\frac{\partial v^{n+1/3}}{\partial y}=0 \tag{2.25}$$

where $n+1/6$ represents the state after the x-advection. This scheme is implicit and unconditionally stable.

For the diffusion step, an alternating direction implicit method is used which leads to

x-sweep

$$\frac{U^*-U^{n+1/3}}{\frac{1}{2}\Delta t}-\frac{\partial}{\partial x}(K\frac{\partial U^*}{\partial x})-\frac{\partial}{\partial y}(K\frac{\partial U^{n+1/3}}{\partial y})=\Omega V^{n+1/3} \tag{2.26}$$

$$\frac{V^*-V^{n+1/3}}{\frac{1}{2}\Delta t}-\frac{\partial}{\partial x}(K\frac{\partial V^*}{\partial x})-\frac{\partial}{\partial y}(K\frac{\partial V^{n+1/3}}{\partial y})=0 \tag{2.27}$$

y-sweep

$$\frac{U^{n+2/3}-U^*}{\frac{1}{2}\Delta t}-\frac{\partial}{\partial x}(K\frac{\partial U^*}{\partial x})-\frac{\partial}{\partial y}(K\frac{\partial U^{n+2/3}}{\partial y})=0 \tag{2.28}$$

$$\frac{V^{n+2/3}-V^*}{\frac{1}{2}\Delta t}-\frac{\partial}{\partial x}(K\frac{\partial V^*}{\partial x})-\frac{\partial}{\partial y}(K\frac{\partial V^{n+2/3}}{\partial y})=-\Omega U^* \tag{2.29}$$

where U^* and V^* are the intermediate values of unit-width discharge during the computation of the diffusion step. Yanenko (1971) showed that the ADI scheme was absolutely consistent and unconditionally stable for the two-dimensional heat conduction equations, which become the working equations for the diffusion step if the source terms are added.

The propagation step is the most important of the three steps and its working equations are more complicated than the other two steps. By introducing a

coefficient α $(0 \leq \alpha \leq 1)$ for the spatial derivatives, the working equations for the propagation step can be written as

$$\begin{aligned}\frac{U^{n+1}-U^{n+2/3}}{\Delta t} &+ \alpha(gh\frac{\partial Z}{\partial x})^{n+1}+(1-\alpha)(gh\frac{\partial Z}{\partial x})^{n}-\frac{U^{n+2/3}}{h^n}(\frac{Z^{n+1}-Z^n}{\Delta t}) \\ &+ \alpha(\frac{\tau_{bx}-\tau_{sx}}{\rho})^{n+1}+(1-\alpha)(\frac{\tau_{bx}-\tau_{sx}}{\rho})^{n}=0 \qquad (2.30)\end{aligned}$$

$$\begin{aligned}\frac{V^{n+1}-V^{n+2/3}}{\Delta t} &+ \alpha(gh\frac{\partial Z}{\partial y})^{n+1}+(1-\alpha)(gh\frac{\partial Z}{\partial y})^{n}-\frac{V^{n+2/3}}{h^n}(\frac{Z^{n+1}-Z^n}{\Delta t}) \\ &+ \alpha(\frac{\tau_{by}-\tau_{sy}}{\rho})^{n+1}+(1-\alpha)(\frac{\tau_{by}-\tau_{sy}}{\rho})^{n}=0 \qquad (2.31)\end{aligned}$$

$$\frac{Z^{n+1}-Z^n}{\Delta t}+\alpha(\frac{\partial U}{\partial x})^{n+1}+(1-\alpha)(\frac{\partial U}{\partial x})^{n}+\alpha(\frac{\partial V}{\partial y})^{n+1}+(1-\alpha)(\frac{\partial V}{\partial y})^{n}=0 \qquad (2.32)$$

It is clear that the scheme is fully implicit when α is equal to 1 and fully explicit when α is equal to 0. From Eqs. (2.30) and (2.31), we can obtain the formulae for U^{n+1} and V^{n+1} as follows

$$\begin{aligned}U^{n+1}=U^{n+2/3} &- \Delta t\{\alpha(gh\frac{\partial Z}{\partial x})^{n+1}+(1-\alpha)(gh\frac{\partial Z}{\partial x})^{n}-\frac{U^{n+2/3}}{h^n}(\frac{Z^{n+1}-Z^n}{\Delta t})\} \\ &- \Delta t\{\alpha(\frac{\tau_{bx}-\tau_{sx}}{\rho})^{n+1}+(1-\alpha)(\frac{\tau_{bx}-\tau_{sx}}{\rho})^{n}\} \qquad (2.33)\end{aligned}$$

$$\begin{aligned}V^{n+1}=V^{n+2/3} &- \Delta t\{\alpha(gh\frac{\partial Z}{\partial y})^{n+1}+(1-\alpha)(gh\frac{\partial Z}{\partial y})^{n}-\frac{V^{n+2/3}}{h^n}(\frac{Z^{n+1}-Z^n}{\Delta t})\} \\ &- \Delta t\{\alpha(\frac{\tau_{by}-\tau_{sy}}{\rho})^{n+1}+(1-\alpha)(\frac{\tau_{by}-\tau_{sy}}{\rho})^{n}\} \qquad (2.34)\end{aligned}$$

Treating the bottom friction and surface friction explicitly for clarity, and substituting U^{n+1} and V^{n+1} into equation (2.32) yield

$$\begin{aligned}\frac{\Delta Z}{g(\Delta t)^2} &- \alpha^2\{\frac{\partial}{\partial x}(h^n\frac{\partial \Delta Z}{\partial x}+\Delta Z\frac{\partial Z^n}{\partial x})+\frac{\partial}{\partial y}(h^n\frac{\partial \Delta Z}{\partial y}+\Delta Z\frac{\partial Z^n}{\partial y})\} \\ &+ \frac{\alpha}{g\Delta t}\{\frac{\partial}{\partial x}(\frac{U^{n+2/3}}{h^n}\Delta Z)+\frac{\partial}{\partial y}(\frac{V^{n+2/3}}{h^n}\Delta Z)\}=f_1+f_2 \qquad (2.35)\end{aligned}$$

where

$$f_1=-\frac{(1-\alpha)}{g\Delta t}(\frac{\partial U}{\partial x})^{n}-\frac{\alpha}{g\Delta t}(\frac{\partial U}{\partial x})^{n+2/3}+\alpha\frac{\partial}{\partial x}(h^n\frac{\partial Z^n}{\partial x})+\frac{\alpha}{g}\frac{\partial}{\partial x}(\frac{\tau_{bx}-\tau_{sx}}{\rho})^{n} \qquad (2.36)$$

$$f_2 = -\frac{(1-\alpha)}{g\Delta t}(\frac{\partial V}{\partial y})^n - \frac{\alpha}{g\Delta t}(\frac{\partial V}{\partial y})^{n+2/3} + \alpha\frac{\partial}{\partial y}(h^n\frac{\partial Z^n}{\partial y}) + \frac{\alpha}{g}\frac{\partial}{\partial y}(\frac{\tau_{by}-\tau_{sy}}{\rho})^n \quad (2.37)$$

and

$$\Delta Z = Z^{n+1} - Z^n \quad (2.38)$$

It is very complicated to solve for ΔZ directly from Eq. (2.35) because of the existence of second derivative terms with respect to x and y. Some researchers (Benque et al, 1982) proposed solving this problem by splitting the Eq. (2.35) into the following two equations

$$\frac{\Delta Z_1}{2g(\Delta t)^2} - \alpha^2\frac{\partial}{\partial x}(h^n\frac{\partial \Delta Z_1}{\partial x} + \Delta Z_1\frac{\partial Z^n}{\partial x}) + \frac{\alpha}{g\Delta t}\frac{\partial}{\partial x}(\frac{U^{n+2/3}}{h^n}\Delta Z_1) = f_1 - q \quad (2.39)$$

$$\frac{\Delta Z_2}{2g(\Delta t)^2} - \alpha^2\frac{\partial}{\partial y}(h^n\frac{\partial \Delta Z_2}{\partial y} + \Delta Z_2\frac{\partial Z^n}{\partial y}) + \frac{\alpha}{g\Delta t}\frac{\partial}{\partial y}(\frac{V^{n+2/3}}{h^n}\Delta Z_2) = f_2 + q \quad (2.40)$$

in which $q(x,y)$ is the coordination term. If $\Delta Z_1 = \Delta Z_2$, the solution of Eq. (2.35), ΔZ, will be exactly the same as the solution of Eq. (2.39) or (2.40), ΔZ_1 or ΔZ_2. The details of solving Eqs. (2.39) and (2.40) for ΔZ_1, ΔZ_2 and q are presented in appendix A.

If the bottom friction terms in Eqs. (2.30) and (2.31) are treated implicitly, the first-order series expansion in terms of velocity (U,V) and water depth h can be used to linearize them as

$$(\frac{\tau_{bx}}{\rho})^{n+1} = (\frac{\tau_{bx}}{\rho})^{n+2/3} + \frac{\partial}{\partial h}(\frac{\tau_{bx}}{\rho})^{n+2/3}\Delta h + \frac{\partial}{\partial U}(\frac{\tau_{bx}}{\rho})^{n+2/3}\Delta U \quad (2.41)$$

$$(\frac{\tau_{by}}{\rho})^{n+1} = (\frac{\tau_{by}}{\rho})^{n+2/3} + \frac{\partial}{\partial h}(\frac{\tau_{by}}{\rho})^{n+2/3}\Delta h + \frac{\partial}{\partial V}(\frac{\tau_{by}}{\rho})^{n+2/3}\Delta V \quad (2.42)$$

where

$$\left.\begin{array}{l}\Delta h = h^{n+1} - h^n = \Delta Z \\ \Delta U = U^{n+1} - U^{n+2/3} \\ \Delta V = V^{n+1} - V^{n+2/3}\end{array}\right\} \quad (2.43)$$

Substituting Eqs. (2.4) and (2.5) into the above equations, we get

$$(\frac{\tau_{bx}}{\rho})^{n+1} = (\frac{\tau_{bx}}{\rho})^{n+2/3} - \frac{g}{C^2}(\frac{2U\sqrt{U^2+V^2}}{h^3})^{n+2/3}\Delta Z + \frac{g}{C^2}(\frac{\sqrt{U^2+V^2}}{h^2} + \frac{U^2}{h^2\sqrt{U^2+V^2}})^{n+2/3}(U^{n+1} - U^{n+2/3}) \quad (2.44)$$

$$(\frac{\tau_{by}}{\rho})^{n+1} = (\frac{\tau_{by}}{\rho})^{n+2/3} - \frac{g}{C^2}(\frac{2V\sqrt{U^2+V^2}}{h^3})^{n+2/3}\Delta Z + \frac{g}{C^2}(\frac{\sqrt{U^2+V^2}}{h^2} + \frac{V^2}{h^2\sqrt{U^2+V^2}})^{n+2/3}(V^{n+1} - V^{n+2/3}) \quad (2.45)$$

Plugging them into Eqs. (2.30) and (2.31), expressions for U^{n+1} and V^{n+1} can be extracted as

$$\begin{aligned} U^{n+1} &= M\{U^{n+2/3} + \frac{U^{n+2/3}}{h^n}\Delta Z - \alpha\Delta t(gh\frac{\partial Z}{\partial x})^{n+1} - (1-\alpha)\Delta t(gh\frac{\partial Z}{\partial x})^n \\ &- \Delta t\frac{g}{C^2}(\frac{U\sqrt{U^2+V^2}}{h^2})^{n+2/3}\} + \alpha\Delta t\frac{g}{C^2}(\frac{2U^2+V^2}{h^2\sqrt{U^2+V^2}})^{n+2/3}U^{n+2/3} \\ &+ 2\alpha\Delta t\frac{g}{C^2}(\frac{U\sqrt{U^2+V^2}}{h^3})^{n+2/3}\Delta Z \end{aligned} \quad (2.46)$$

$$\begin{aligned} V^{n+1} &= N\{V^{n+2/3} + \frac{V^{n+2/3}}{h^n}\Delta Z - \alpha\Delta t(gh\frac{\partial Z}{\partial y})^{n+1} - (1-\alpha)\Delta t(gh\frac{\partial Z}{\partial y})^n \\ &- \Delta t\frac{g}{C^2}(\frac{V\sqrt{U^2+V^2}}{h^2})^{n+2/3}\} + \alpha\Delta t\frac{g}{C^2}(\frac{U^2+2V^2}{h^2\sqrt{U^2+V^2}})^{n+2/3}V^{n+2/3} \\ &+ 2\alpha\Delta t\frac{g}{C^2}(\frac{V\sqrt{U^2+V^2}}{h^3})^{n+2/3}\Delta Z \end{aligned} \quad (2.47)$$

where

$$M = 1/[1 + \frac{\alpha g\Delta t}{C^2}(\frac{2U^2+V^2}{h^2\sqrt{U^2+V^2}})^{n+2/3}] \quad (2.48)$$

and

$$N = 1/[1 + \frac{\alpha g\Delta t}{C^2}(\frac{U^2+2V^2}{h^2\sqrt{U^2+V^2}})^{n+2/3}] \quad (2.49)$$

Substituting U^{n+1} and V^{n+1} into Eq. (2.32), an equation for the single unknown $\Delta Z(x,y)$ will be obtained. It can be solved by using a splitting scheme. The two split equations are

$$\frac{\Delta Z_1}{2g(\Delta t)^2} - M\alpha^2\{\frac{\partial(h^n\frac{\partial\Delta Z_1}{\partial x})}{\partial x} + \frac{\partial(\Delta Z_1\frac{\partial Z^n}{\partial x})}{\partial x}\} + \frac{M\alpha}{g\Delta t}\frac{\partial(\frac{U^{n+2/3}}{h^n}\Delta Z_1)}{\partial x} = f_1 - q \quad (2.50)$$

$$\frac{\Delta Z_2}{2g(\Delta t)^2} - N\alpha^2\{\frac{\partial(h^n\frac{\partial\Delta Z_2}{\partial y})}{\partial y} + \frac{\partial(\Delta Z_2\frac{\partial Z^n}{\partial y})}{\partial y}\} + \frac{N\alpha}{g\Delta t}\frac{\partial(\frac{V^{n+2/3}}{h^n}\Delta Z_2)}{\partial y} = f_2 + q \quad (2.51)$$

where

$$f_1 = -\frac{1-\alpha}{g\Delta t}(\frac{\partial U}{\partial x})^n - \frac{M\alpha}{g\Delta t}(\frac{\partial U}{\partial x})^{n+2/3} + M\alpha\frac{\partial(h\frac{\partial Z}{\partial x})^n}{\partial x} + \frac{M\alpha}{C^2}\frac{\partial(\frac{U\sqrt{U^2+V^2}}{h^2})^{n+2/3}}{\partial x} \quad (2.52)$$

$$f_2 = -\frac{1-\alpha}{g\Delta t}(\frac{\partial V}{\partial y})^n - \frac{M\alpha}{g\Delta t}(\frac{\partial V}{\partial y})^{n+2/3} + M\alpha\frac{\partial(h\frac{\partial Z}{\partial y})^n}{\partial y} + \frac{M\alpha}{C^2}\frac{\partial(\frac{V\sqrt{U^2+V^2}}{h^2})^{n+2/3}}{\partial y} \quad (2.53)$$

2.3 Grid System

A mesh of rectangular cells with variable sizes is established for the development of the finite difference approximation. This is shown in Fig. 2.2 in which ΔX_i expresses the size for ith column and ΔY_j represents the size for jth row.

The grid system is shown in Fig. 2.3. In this grid, each (U,V) grid point is located at the center of rectangles made by four Z grid points, but each Z grid point is not at the center of rectangles marked by (U,V) points. The cell sizes in the (U,V) grid system and in the Z grid system have the following relationship

$$\Delta X_{u(i)} = \frac{\Delta X_{z(i)} + \Delta X_{z(i+1)}}{2} \quad (2.54)$$

$$\Delta Y_{u(j)} = \frac{\Delta Y_{z(j-1)} + \Delta Y_{z(j)}}{2} \quad (2.55)$$

where $\Delta X_{u(i)}$ and $\Delta Y_{u(j)}$ present the sizes of the cell (i,j) in the (U,V) grid system, $\Delta X_{z(i)}$ and $\Delta Y_{z(j)}$ express the sizes of the cell (i,j) in the Z grid system.

To illustrate the relationship between the numerical values in the Z and (U,V) systems, a property denoted by P is introduced. Its value at the (U,V) grid points is denoted by $P_{u(i,j)}$ and at the Z grid points by $P_{z(i,j)}$. $P_{z(i+1/2,j)}$ is used to denote the value at the middle point between the Z grid points (i,j) and $(i+1,j)$, and $P_{z(i,j+1/2)}$ represents the value at the middle point between the Z grid points (i,j) and $(i,j+1)$. Given these, we have

$$P_{u(i,j)} = [P_{z(i,j)} + P_{z(i,j-1)} + P_{z(i+1,j)} + P_{z(i+1,j-1)}]/4 \quad (2.56)$$

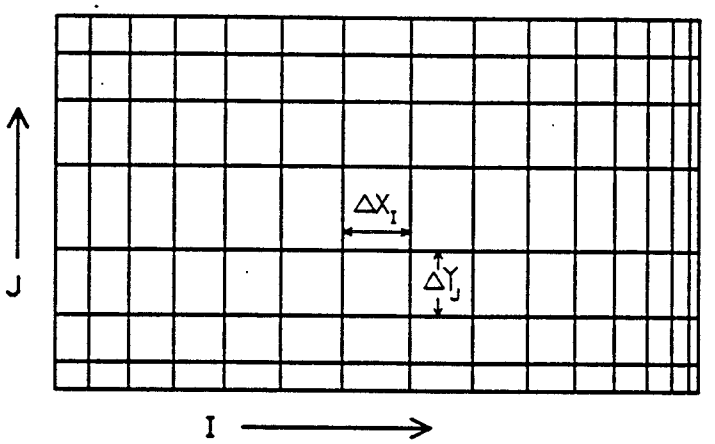


Figure 2.2: Schematic of finite difference mesh with variable rectangular cells

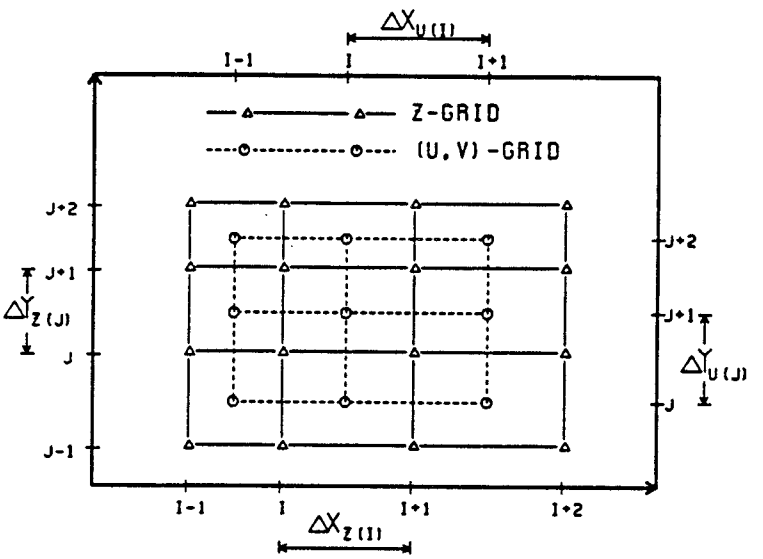


Figure 2.3: Computational grid definition

$$P_{z(i,j)} = \frac{\Delta X_{z(i-1)}\Delta Y_{z(j-1)}}{4\Delta X_{u(i-1)}\Delta Y_{u(j)}} P_{u(i,j+1)} + \frac{\Delta X_{z(i-1)}\Delta Y_{z(j)}}{4\Delta X_{u(i-1)}\Delta Y_{u(j)}} P_{u(i,j)}$$

$$+ \frac{\Delta X_{z(i)}\Delta Y_{z(j-1)}}{4\Delta X_{u(i-1)}\Delta Y_{u(j)}} P_{u(i-1,j+1)} + \frac{\Delta X_{z(i)}\Delta Y_{z(j)}}{4\Delta X_{u(i-1)}\Delta Y_{u(j)}} P_{u(i-1,j)} \quad (2.57)$$

$$P_{z(i+1/2,j)} = \frac{\Delta Y_{z(j-1)}}{2\Delta Y_{u(j)}} P_{u(i,j+1)} + \frac{\Delta Y_{z(j)}}{2\Delta Y_{u(j)}} P_{u(i,j)} \quad (2.58)$$

and

$$P_{z(i,j+1/2)} = \frac{\Delta X_{z(i-1)}}{2\Delta X_{u(i-1)}} P_{u(i,j+1)} + \frac{\Delta X_{z(i)}}{2\Delta X_{u(i-1)}} P_{u(i-1,j+1)} \quad (2.59)$$

$P_{z(i+1/2,j)}$ and $P_{z(i,j+1/2)}$ can also be evaluated by

$$P_{z(i+1/2,j)} = [P_{z(i,j)} + P_{z(i+1,j)}]/2 \quad (2.60)$$

$$P_{z(i,j+1/2)} = [P_{z(i,j)} + P_{z(i,j+1)}]/2 \quad (2.61)$$

For a spatially uniform grid system, Eqs. (2.57), (2.58) and (2.59) become

$$P_{z(i,j)} = [P_{u(i,j+1)} + P_{u(i,j)} + P_{u(i-1,j+1)} + P_{u(i-1,j)}]/4 \quad (2.62)$$

$$P_{z(i+1/2,j)} = [P_{u(i,j+1)} + P_{u(i,j)}]/2 \quad (2.63)$$

$$P_{z(i,j+1/2)} = [P_{u(i,j+1)} + P_{u(i-1,j+1)}]/2 \quad (2.64)$$

2.4 Finite Difference Approximation

Suppose that the state of the model is known at time $n\Delta t$. Then the state of the model at time $(n+1)\Delta t$ can be obtained by successively solving the advection, diffusion and propagation steps. From Eqs. (2.22) to (2.29), it is seen that the water surface elevation Z does not appear in the advection and diffusion steps. Therefore these steps can be executed solely on the (U, V) grid.

The finite difference equations for the advection step are

x-advection

$$\frac{u_{i,j}^{n+1/6} - u_{i,j}^{n}}{\Delta t} + u_{i,j}^{n} \frac{u_{i+1,j}^{n+1/6} - u_{i-1,j}^{n+1/6}}{\Delta X_{u(i-1)} + \Delta X_{u(i)}} = 0 \quad (2.65)$$

$$\frac{v_{i,j}^{n+1/6} - v_{i,j}^{n}}{\Delta t} + u_{i,j}^{n} \frac{v_{i+1,j}^{n+1/6} - v_{i-1,j}^{n+1/6}}{\Delta X_{u(i-1)} + \Delta X_{u(i)}} = 0 \tag{2.66}$$

y-advection

$$\frac{u_{i,j}^{n+1/3} - u_{i,j}^{n+1/6}}{\Delta t} + v_{i,j}^{n+1/6} \frac{u_{i,j+1}^{n+1/3} - u_{i,j-1}^{n+1/3}}{\Delta Y_{u(j-1)} + \Delta Y_{u(j)}} = 0 \tag{2.67}$$

$$\frac{v_{i,j}^{n+1/3} - v_{i,j}^{n+1/6}}{\Delta t} + v_{i,j}^{n+1/6} \frac{v_{i,j+1}^{n+1/3} - v_{i,j-1}^{n+1/3}}{\Delta Y_{u(j-1)} + \Delta Y_{u(j)}} = 0 \tag{2.68}$$

where the subscripts i and j correspond to the (U,V) grid. After x-advection and y-advection, we get the modified velocities $u^{n+1/3}$ and $v^{n+1/3}$. The discharges corresponding to $u^{n+1/3}$ and $v^{n+1/3}$ can be calculated by

$$U_{i,j}^{n+1/3} = u_{i,j}^{n+1/3} h_{i,j}^{n} \tag{2.69}$$

$$V_{i,j}^{n+1/3} = V_{i,j}^{n+1/3} h_{i,j}^{n} \tag{2.70}$$

in which $h_{i,j}^{n}$ is water depth on the (U,V) grid at time $n\Delta t$.

The finite difference equations for the diffusion step are as follows:

x-sweep

$$\begin{aligned} \frac{U_{i,j}^{*} - U_{i,j}^{n+1/3}}{\frac{1}{2}\Delta t} &- \frac{K}{\Delta X_{z(i)}} \left[\frac{U_{i+1,j}^{*} - U_{i,j}^{*}}{\Delta X_{u(i)}} - \frac{U_{i,j}^{*} - U_{i-1,j}^{*}}{\Delta X_{u(i-1)}} \right] \\ &- \frac{K}{\Delta Y_{z(j-1)}} \left[\frac{U_{i,j+1}^{n+1/3} - U_{i,j}^{n+1/3}}{\Delta Y_{u(j)}} - \frac{U_{i,j}^{n+1/3} - U_{i,j-1}^{n+1/3}}{\Delta Y_{u(j-1)}} \right] \\ &= \Omega V_{i,j}^{n+1/3} \end{aligned} \tag{2.71}$$

$$\begin{aligned} \frac{V_{i,j}^{*} - V_{i,j}^{n+1/3}}{\frac{1}{2}\Delta t} &- \frac{K}{\Delta X_{z(i)}} \left[\frac{V_{i+1,j}^{*} - V_{i,j}^{*}}{\Delta X_{u(i)}} - \frac{V_{i,j}^{*} - V_{i-1,j}^{*}}{\Delta X_{u(i-1)}} \right] \\ &- \frac{K}{\Delta Y_{z(j-1)}} \left[\frac{V_{i,j+1}^{n+1/3} - V_{i,j}^{n+1/3}}{\Delta Y_{u(j)}} - \frac{V_{i,j}^{n+1/3} - V_{i,j-1}^{n+1/3}}{\Delta Y_{u(j-1)}} \right] \\ &= 0 \end{aligned} \tag{2.72}$$

y-sweep

$$\frac{U_{i,j}^{n+2/3} - U_{i,j}^{*}}{\frac{1}{2}\Delta t} - \frac{K}{\Delta X_{z(i)}} \left[\frac{U_{i+1,j}^{*} - U_{i,j}^{*}}{\Delta X_{u(i)}} - \frac{U_{i,j}^{*} - U_{i-1,j}^{*}}{\Delta X_{u(i-1)}} \right]$$

$$
\begin{aligned}
& - \frac{K}{\Delta Y_{z(j-1)}}\left[\frac{U^{n+2/3}_{i,j+1} - U^{n+2/3}_{i,j}}{\Delta Y_{u(j)}} - \frac{U^{n+2/3}_{i,j} - U^{n+2/3}_{i,j-1}}{\Delta Y_{u(j-1)}}\right] \\
& = 0 \qquad (2.73)
\end{aligned}
$$

$$
\begin{aligned}
\frac{V^{n+2/3}_{i,j} - V^{*}_{i,j}}{\frac{1}{2}\Delta t} & - \frac{K}{\Delta X_{z(i)}}\left[\frac{V^{*}_{i+1,j} - V^{*}_{i,j}}{\Delta X_{u(i)}} - \frac{V^{*}_{i,j} - V^{*}_{i-1,j}}{\Delta X_{u(i-1)}}\right] \\
& - \frac{K}{\Delta Y_{z(j-1)}}\left[\frac{V^{n+2/3}_{i,j+1} - V^{n+2/3}_{i,j}}{\Delta Y_{u(j)}} - \frac{V^{n+2/3}_{i,j} - V^{n+2/3}_{i,j-1}}{\Delta Y_{u(j-1)}}\right] \\
& = -\Omega\, U^{*}_{i,j} \qquad (2.74)
\end{aligned}
$$

where (i,j) also corresponds to the (U,V) grid.

After the diffusion step, the discharges $U^{n+2/3}$ and $V^{n+2/3}$ are known on the (U,V) grid. The propagation step is performed in terms of these modified discharges and consists of two computations. One is the computation of Z^{n+1} which is executed on the Z grid. The other is the evaluation of U^{n+1} and V^{n+1} which is performed on the (U,V) grid. The finite difference equations for the computation of Z^{n+1} can be expressed as

$$
\begin{aligned}
\frac{\Delta Z_{1(i,j)}}{2g(\Delta t)^2} & - \frac{\alpha^2}{\Delta X_{u(i-1)}}\left[h^{n}_{i+1/2,j}\frac{\Delta Z_{1(i+1,j)} - \Delta Z_{1(i,j)}}{\Delta X_{z(i)}} + \Delta Z_{1(i+1/2,j)}\frac{Z^{n}_{i+1,j} - Z^{n}_{i,j}}{\Delta X_{z(i)}}\right. \\
& \left. - h^{n}_{i-1/2,j}\frac{\Delta Z_{1(i,j)} - \Delta Z_{1(i-1,j)}}{\Delta X_{z(i-1)}} - \Delta Z_{1(i-1/2,j)}\frac{Z^{n}_{i,j} - Z^{n}_{i-1,j}}{\Delta X_{z(i-1)}}\right] \\
& + \frac{\alpha}{g\Delta t\Delta X_{u(i-1)}}\left[\frac{U^{n+2/3}_{i+1/2,j}}{h^{n}_{i+1/2,j}}\Delta Z_{1(i+1/2,j)} - \frac{U^{n+2/3}_{i-1/2,j}}{h^{n}_{i-1/2,j}}\Delta Z_{1(i-1/2,j)}\right] \\
& = f_1 - q \qquad (2.75)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\Delta Z_{2(i,j)}}{2g(\Delta t)^2} & - \frac{\alpha^2}{\Delta Y_{u(j)}}\left[h^{n}_{i,j+1/2}\frac{\Delta Z_{2(i,j+1)} - \Delta Z_{2(i,j)}}{\Delta Y_{z(j)}} + \Delta Z_{2(i,j+1/2)}\frac{Z^{n}_{i,j+1} - Z^{n}_{i,j}}{\Delta Y_{z(j)}}\right. \\
& \left. - h^{n}_{i,j-1/2}\frac{\Delta Z_{2(i,j)} - \Delta Z_{2(i,j-1)}}{\Delta Y_{z(j-1)}} - \Delta Z_{2(i,j-1/2)}\frac{Z^{n}_{i,j} - Z^{n}_{i,j-1}}{\Delta Y_{z(j-1)}}\right] \\
& + \frac{\alpha}{g\Delta t\Delta Y_{u(j)}}\left[\frac{V^{n+2/3}_{i,j+1/2}}{h^{n}_{i,j+1/2}}\Delta Z_{2(i,j+1/2)} - \frac{V^{n+2/3}_{i,j-1/2}}{h^{n}_{i,j-1/2}}\Delta Z_{2(i,j-1/2)}\right] \\
& = f_2 + q \qquad (2.76)
\end{aligned}
$$

in which

$$
f_1 = -\frac{(1-\alpha)}{g\Delta t}\frac{U^{n}_{i+1/2,j} - U^{n}_{i-1/2,j}}{\Delta X_{u(i-1)}} - \frac{\alpha}{g\Delta t}\frac{U^{n+2/3}_{i+1/2,j} - U^{n+2/3}_{i-1/2,j}}{\Delta X_{u(i-1)}}
$$

$$+ \frac{\alpha}{\Delta X_{u(i-1)}}[h^n_{i+1/2,j}\frac{Z^n_{i+1,j}-Z^n_{i,j}}{\Delta X_{z(i)}} - h^n_{i-1/2,j}\frac{Z^n_{i,j}-Z^n_{i-1,j}}{\Delta X_{z(i-1)}}]$$
$$+ \frac{\alpha}{\rho g \Delta X_{u(i-1)}}[(\tau_{bx}-\tau_{sx})^n_{i+1/2,j} - (\tau_{bx}-\tau_{sx})^n_{i-1/2,j}] \quad (2.77)$$

$$f_2 = -\frac{(1-\alpha)}{g\Delta t}\frac{V^n_{i,j+1/2}-V^n_{i,j-1/2}}{\Delta Y_{u(j)}} - \frac{\alpha}{g\Delta t}\frac{V^{n+2/3}_{i,j+1/2}-V^{n+2/3}_{i,j-1/2}}{\Delta Y_{u(j)}}$$
$$+ \frac{\alpha}{\Delta Y_{u(j)}}[h^n_{i,j+1/2}\frac{Z^n_{i,j+1}-Z^n_{i,j}}{\Delta Y_{z(j)}} - h^n_{i,j-1/2}\frac{Z^n_{i,j}-Z^n_{i,j-1}}{\Delta Y_{z(j-1)}}]$$
$$+ \frac{\alpha}{\rho g \Delta Y_{u(j)}}[(\tau_{by}-\tau_{sy})^n_{i,j+1/2} - (\tau_{by}-\tau_{sy})^n_{i,j-1/2}] \quad (2.78)$$

and

$$\left.\begin{aligned} h_{i\pm1/2,j} &= (h_{i\pm1,j}+h_{i,j})/2 \\ h_{i,j\pm1/2} &= (h_{i,j\pm1}+h_{i,j})/2 \\ \Delta Z_{i\pm1/2,j} &= (\Delta Z_{i\pm1,j}+\Delta Z_{i,j})/2 \\ \Delta Z_{i,j\pm1/2} &= (\Delta Z_{i,j\pm1}+\Delta Z_{i,j})/2 \end{aligned}\right\} \quad (2.79)$$

Note that the subscripts i, j, $i \pm 1/2$ and $j \pm 1/2$ refer to the $Z-$grid and the discharges at the $Z-$grid can be calculated by solving Eqs. (2.57), (2.58) and (2.59) in terms of the discharges on the (U,V) grid.

The finite difference forms for U^{n+1} and V^{n+1} are

$$U^{n+1}_{i,j} = U^{n+2/3}_{i,j} - \Delta t\alpha g h^{n+1}_{i,j}\frac{Z^{n+1}_{i+1/2,j}-Z^{n+1}_{i-1/2,j}}{\Delta X_{z(i)}}$$
$$- \Delta t(1-\alpha)gh^n_{i,j}\frac{Z^n_{i+1/2,j}-Z^n_{i-1/2,j}}{\Delta X_{z(i)}} + \frac{U^{n+2/3}_{i,j}}{h^n_{i,j}}(Z^{n+1}_{i,j}-Z^n_{i,j})$$
$$- \frac{\Delta t\alpha}{\rho}(\tau_{bx}-\tau_{sx})^{n+1}_{i,j} - \frac{\Delta t(1-\alpha)}{\rho}(\tau_{bx}-\tau_{sx})^n_{i,j} \quad (2.80)$$

$$V^{n+1}_{i,j} = V^{n+2/3}_{i,j} - \Delta t\alpha g h^{n+1}_{i,j}\frac{Z^{n+1}_{i,j+1/2}-Z^{n+1}_{i,j-1/2}}{\Delta Y_{z(j-1)}}$$
$$- \Delta t(1-\alpha)gh^n_{i,j}\frac{Z^n_{i,j+1/2}-Z^n_{i,j+1/2}}{\Delta Y_{z(j-1)}} + \frac{V^{n+2/3}_{i,j}}{h^n_{i,j}}(Z^{n+1}_{i,j}-Z^n_{i,j})$$
$$- \frac{\Delta t\alpha}{\rho}(\tau_{by}-\tau_{sy})^{n+1}_{i,j} - \frac{\Delta t(1-\alpha)}{\rho}(\tau_{by}-\tau_{sy})^n_{i,j} \quad (2.81)$$

in which the indices i, j, $i \pm 1/2$ and $j \pm 1/2$ refer to the (U,V) grid.

2.5 Boundary and Initial Conditions

Two types of boundary conditions are normally encountered in the numerical simulation of tidal circulation. One is the open boundary which is an artificial termination of the computational system. The other is the shoreline boundary. At the open boundary, the water surface elevation and/or the current velocities are specified. In this model, the velocity gradients in the direction normal to the open boundary is set to zero, and the water surface elevation is specified along the open boundary.

The shoreline boundary can be treated as either a moving boundary or a fixed boundary. The moving boundary problem will be discussed in detail in Chapter 4. Along a fixed boundary, the normal velocity is zero, but the tangential velocity may either be set to zero (no-slip condition) or satisfy the zero-slope in the normal direction (slip condition).

In this model, boundary conditions must be imposed at each of the fractional steps. Since the water surface elevation does not appear in the advection and diffusion steps, only velocity boundary conditions are imposed for these steps. For the propagation step, however, boundary conditions in terms of surface elevation and velocities must be specified along the open and closed (fixed or moving) boundaries.

The specification of the initial conditions requires the knowledge of the free surface position and the flow field at $t = 0$. Usually this is impossible and the model is started with the still water condition that Z =constant and $U = V = 0$ at $t = 0$.

2.6 Consistency and stability

As seen previously, the numerical scheme for each of the three fractional steps is relatively simple. However, we need to show that the combination of these three steps is consistent with the original governing equations. This is hard to prove

theoretically because of the existence of the two-dimensional nonlinear terms and diffusion terms. This is a drawback in applying the method of fractional steps to the problems of fluid dynamics in several variables. It will be seen in the next chapter, however, that this drawback does not appear to affect the accuracy of the numerical model.

Since each fractional step is unconditionally stable, the complete scheme is unconditional stable. However, for numerical accuracy, it is desirable to keep the time step sufficiently small such that the Courant number based on the velocities does not become exceedingly larger than 1.

CHAPTER 3
COMPARISONS BETWEEN THEORETICAL AND NUMERICAL SOLUTIONS

In this Chapter, results of the two-dimensional tidal circulation model will be compared to four different theoretical solutions. The first comparison will primarily investigate the model's capability to simulate the propagation step. The second comparison will validate the model's ability to compute the nonlinear effects. The other two comparisons will focus on the simulation of the Coriolis and friction forces. From these comparisons, we can then assess the model's accuracy and consistency.

3.1 Comparison With a Linear Theoretical Solution

Neglecting the advective terms, diffusion terms, bottom friction and wind surface stress, the one-dimensional shallow water equations are

$$\frac{\partial U}{\partial t} + gh\frac{\partial Z}{\partial x} = 0 \tag{3.1}$$

$$\frac{\partial Z}{\partial t} + \frac{\partial U}{\partial x} = 0 \tag{3.2}$$

where $U = uh$ represents the vertically integrated velocity in the x-direction, u is the vertically averaged velocity, h is the mean water depth and Z is water surface elevation. Assuming a closed boundary at $x = l$ and an open boundary at $x = 0$, the boundary conditions and initial conditions associated with Eqs. (3.1) and (3.2) are:

Boundary conditions

$$U(l,t) = 0 \tag{3.3}$$

$$Z(0,t) = Z_0 + a\sin\omega t \tag{3.4}$$

Initial conditions

$$Z(x,0) = Z_0 \tag{3.5}$$

$$U(x,0) = 0 \tag{3.6}$$

where l is the length of an estuary, Z_0 is the still water level elevation, and a and ω are the amplitude and frequency of the forcing tide at the open boundary, respectively.

Combining Eq. (3.1) with Eq. (3.2), we get

$$\frac{\partial^2 Z}{\partial x^2} = c^2 \frac{\partial^2 Z}{\partial x^2} \tag{3.7}$$

where c is the wave speed. which can be expressed as $c = \sqrt{gh}$. Let the solution of Eq. (3.7) take the form of

$$Z(x,t) = \frac{a \cos k(l-x)}{\cos kl} \sin \omega t + \sum_{n=0}^{\infty} A_n \sin k_n x \sin \omega_n t + Z_0 \tag{3.8}$$

which apparently satisfies the boundary condition (3.4) and initial condition (3.5). Substituting Eq. (3.8) into Eq. (3.7), we can obtain the following dispersion relationship

$$\omega^2 = k^2 c^2 \tag{3.9}$$

$$\omega_n^2 = k_n^2 c^2 \tag{3.10}$$

where k and k_n represent the wave numbers. From Eq. (3.1), it is seen that boundary condition (3.3) leads to

$$\frac{\partial Z}{\partial x} |_{x=l} = 0 \tag{3.11}$$

In order to satisfy this boundary condition, it follows

$$\sum_{n=0}^{\infty} A_n k_n \cos k_n l \sin \omega_n t = 0 \tag{3.12}$$

Therefore

$$\cos k_n l = 0 \tag{3.13}$$

So

$$k_n = \frac{(2n+1)\pi}{2l} \tag{3.14}$$

where n is an integer. By using initial condition (3.6), we can determine the coefficient A_n as

$$\begin{aligned} A_n &= \frac{-2a\omega}{l\omega_n}\int_0^l \frac{\cos k(l-x)}{\cos kl}\sin k_n x\, dx \\ &= \frac{-2a\omega}{lc(k_n^2-k^2)} \end{aligned} \tag{3.15}$$

Consequently, we obtain

$$Z(x,t) = \frac{a\cos k(l-x)}{\cos kl}\sin\omega t + \sum_{n=0}^{\infty}[\frac{-2a\omega}{lc(k_n^2-k^2)}\sin k_n x\sin\omega_n t] + Z_0 \tag{3.16}$$

Substituting the above equation into Eq. (3.1), we get

$$U(x,t) = \frac{ac\sin k(l-x)}{\cos kl}\cos\omega t + \sum_{n=0}^{\infty}[\frac{-2a\omega}{l(k_n^2-k^2)}\cos k_n x\cos\omega_n t] \tag{3.17}$$

To compare the numerical solution with a linear theoretical solution, a rectangular basin shown in Fig. 3.1 with a constant water depth of 10 meters is considered. Assume that a periodic tide with an amplitude of 50 centimeters and a period of 12.4 hrs is forced along the mouth of the basin. The still water level elevation Z_0 is set to be 0. The numerical solutions can be obtained by discretizing the rectangular basin with 15×30 grid points as shown in Fig. 3.2 and the use of a time step of 30 minutes.

It is noted that in the numerical computation, extra diffusion is introduced due to the use of the implicit numerical scheme. Thus the numerical solution should correspond to the first mode of the theoretical solution, i.e., the first terms in the Eqs. (3.16) and (3.17). The comparison of them is shown in Figs. 3.3 and 3.4.

Figure 3.3 shows the water surface elevation near the mouth ($x = 10$km), at the middle point of the estuary ($x = 30$km), and near the closed boundary ($x = 60$km). Figure 3.4 presents the velocity near the mouth ($x = 10$km), at the middle point

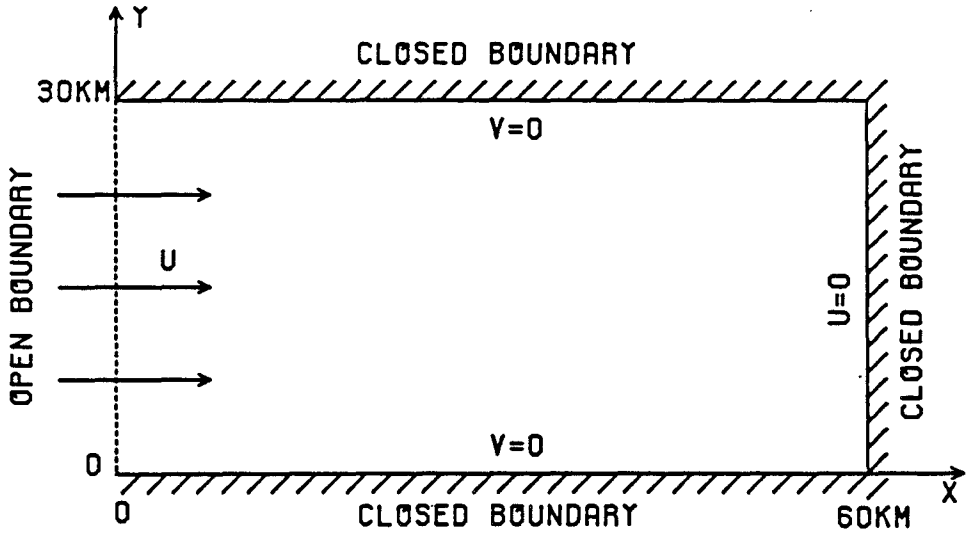


Figure 3.1: Schematic diagram of a rectangular basin

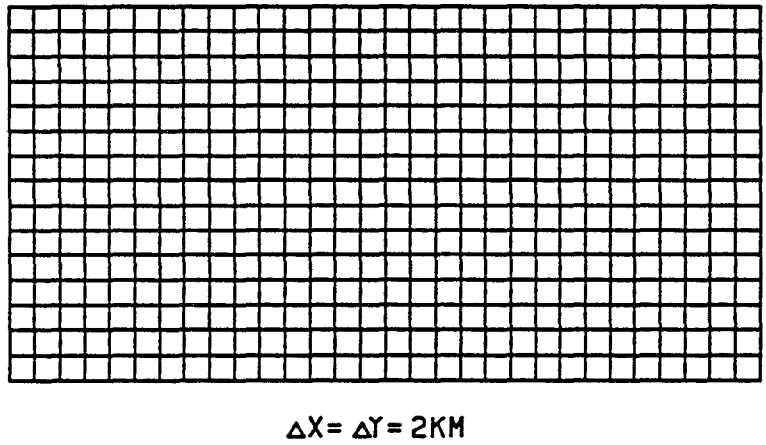


Figure 3.2: Computational grid

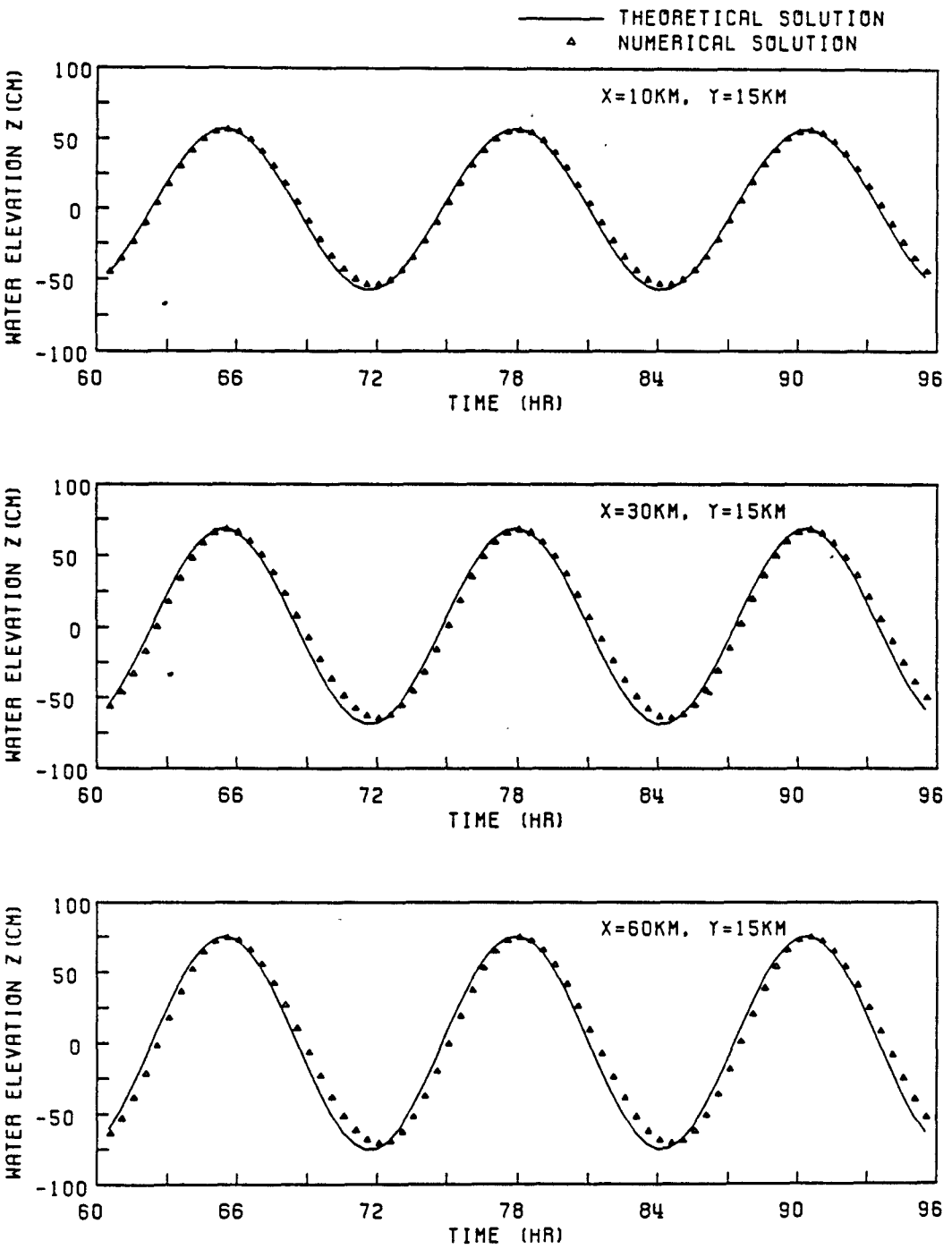


Figure 3.3: Comparison between theoretical and numerical solutions for water surface elevation

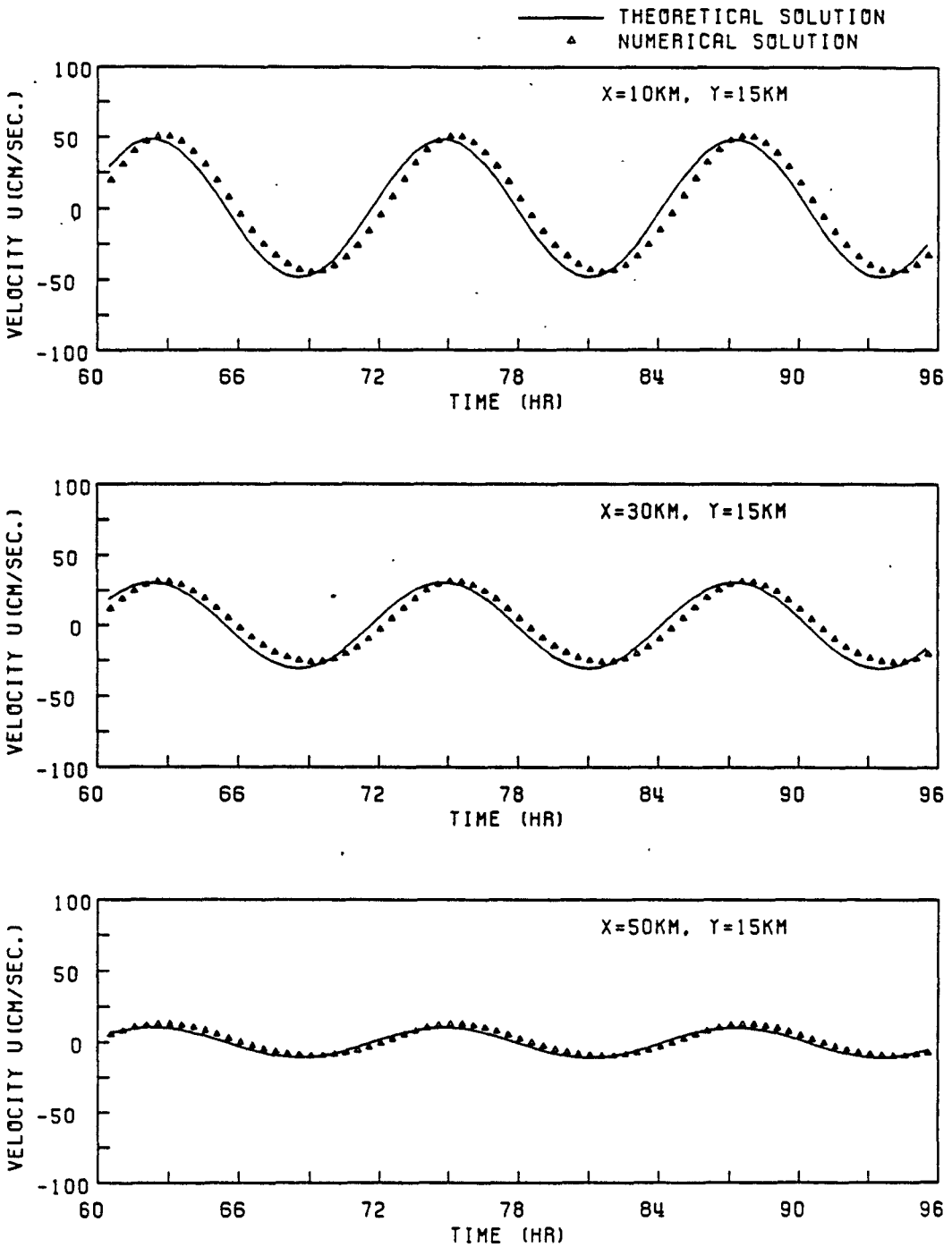


Figure 3.4: Comparison between theoretical and numerical solutions for velocity

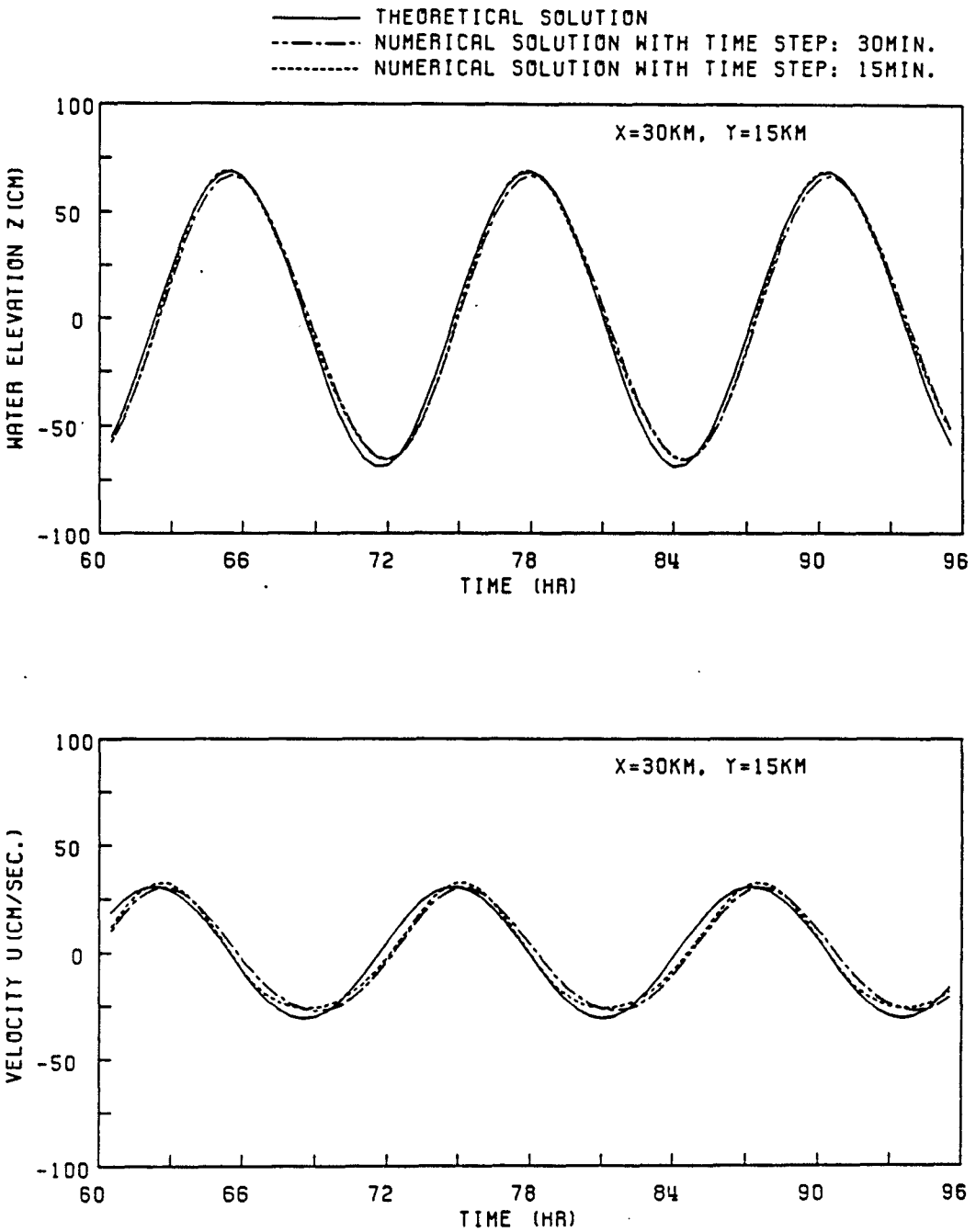


Figure 3.5: Comparison of theoretical solution to numerical results with different time steps

($x = 30$km), and near the closed boundary ($x = 50$km). Both Fig. 3.3 and Fig. 3.4 show that there is a reasonably good agreement between theoretical and numerical solutions, even though a slight phase shift between the theoretical and numerical results exists.

To investigate the origin of the phase shift, different time steps for the numerical computation are used and the numerical results for water surface elevation and velocity are presented in Fig. 3.5. From these results, it is seen that the phase shift between theoretical and numerical solutions tends to diminish as the time step decreases.

3.2 Comparison With a Non-linear Theoretical Solution

When nonlinear terms are included in two-dimensional shallow water equations, it is impossible to obtain an analytical solution. In one-dimensional nonlinear tidal motion, however, we can use harmonic analysis to develop a theoretical solution. Obtaining exact theoretical solutions, however, is still difficult because the high order terms are difficult to solve for. In this study, only the zeroth and first order harmonic solutions are developed.

Without the consideration of Coriolis and bottom friction forces, the governing equations for one-dimensional nonlinear tidal motion can be written as

$$\frac{\partial U}{\partial t} + u\frac{\partial U}{\partial x} + gh\frac{\partial Z}{\partial x} = 0 \tag{3.18}$$

$$\frac{\partial Z}{\partial t} + \frac{\partial U}{\partial x} = 0 \tag{3.19}$$

where h is assumed to be constant. The bounary conditions are

$$Z(0,t) = a\sin\omega t \tag{3.20}$$

$$U(l,t) = 0 \tag{3.21}$$

Based on the idea of harmonic analysis, the theoretical solution of Eqs. (3.18) and

(3.19) can be expressed as

$$Z = Z_1 + Z_2 \tag{3.22}$$

$$U = U_1 + U_2 \tag{3.23}$$

$$u = \frac{U_1 + U_2}{h} \tag{3.24}$$

where Z_1 and U_1 are the theoretical solutions of one-dimensional linear shallow water equations which have been obtained in the preceding section:

$$Z_1 = \frac{a\cos k(l-x)}{\cos kl}\sin\omega t \tag{3.25}$$

$$U_1 = \frac{ac\sin k(l-x)}{\cos kl}\cos\omega t \tag{3.26}$$

Subsituting Z, U and u into Eqs. (3.18) and (3.19) and neglecting terms higher than the first order ones, we can get the governing equations for Z_2 and U_2:

$$\begin{aligned}\frac{\partial U_2}{\partial t} + gh\frac{\partial Z_2}{\partial x} &= \frac{a^2c^2k}{8h\cos^2 kl}\{\sin[2k(l-x)+2\omega t] \\ &+ \sin[2k(l-x)-2\omega t] + 2\sin 2k(l-x)\}\end{aligned} \tag{3.27}$$

$$\frac{\partial Z_2}{\partial t} + \frac{\partial U_2}{\partial x} = 0 \tag{3.28}$$

The boundary conditions for Z_2 and U_2 can be obtained from Eqs. (3.20), (3.21), (3.22) and (3.23) as follows

$$U_2(l,t) = 0 \tag{3.29}$$

$$Z_2(0,t) = 0 \tag{3.30}$$

Subject to these boundary conditions, the solutions of Eqs. (3.27) and (3.28) are

$$\begin{aligned}Z_2 &= \frac{a^2k}{8h\cos^2 kl}[x\sin 2k(l-x) + \frac{l}{\cos 4kl}\sin 2k(l+x) \\ &- \frac{l}{\cos 4kl}\tan 2kl\cos 2k(l-x)]\cos 2\omega t\end{aligned} \tag{3.31}$$

$$\begin{aligned}U_2 &= \frac{a^2\omega}{8h\cos^2 kl}[x\cos 2k(l-x) + \frac{1}{2k}\sin 2k(l-x) \\ &- \frac{l}{\cos 4kl}\cos 2k(l+x) + \frac{l}{\cos 4kl}\tan 2kl\sin 2k(l-x)]\sin 2\omega t\end{aligned} \tag{3.32}$$

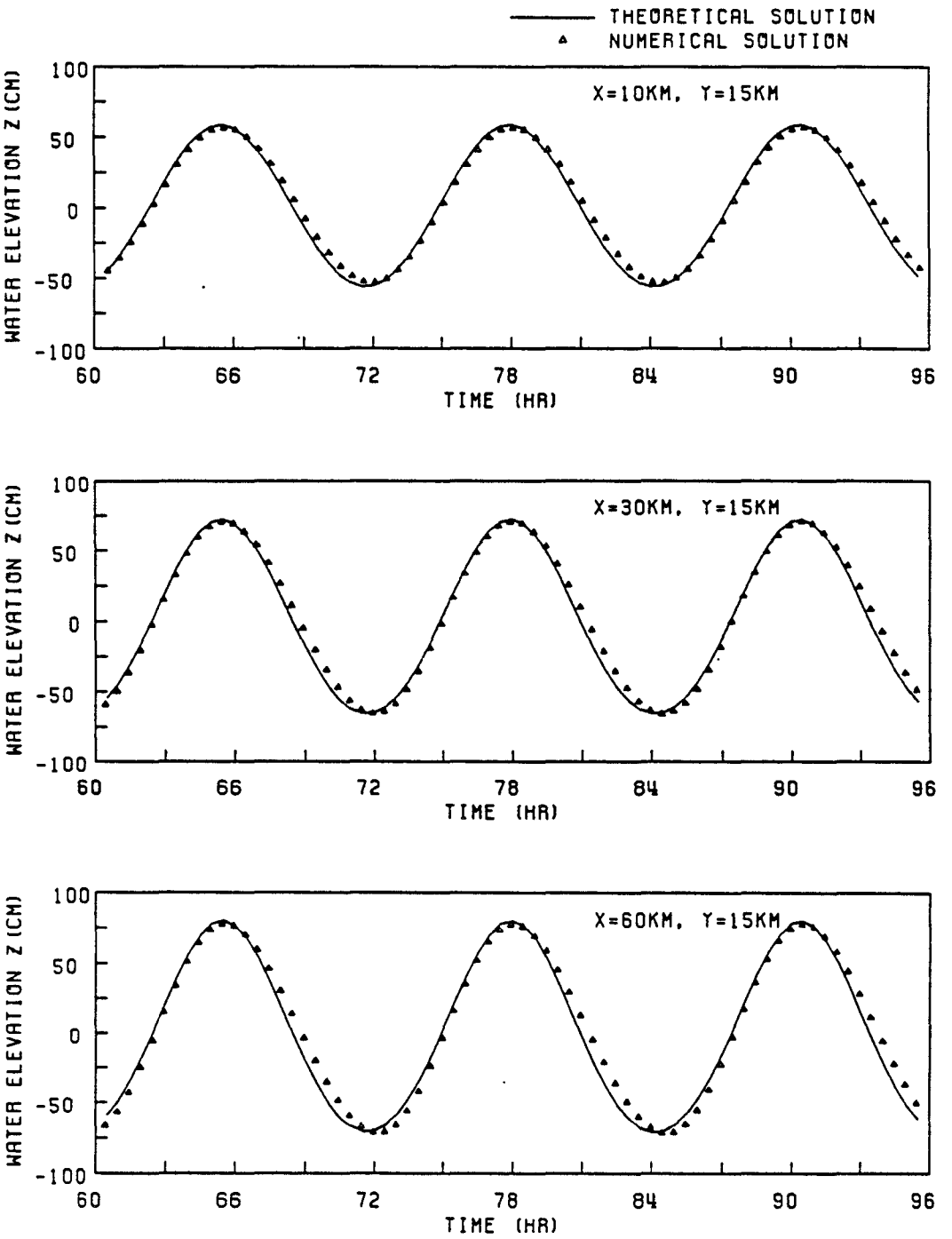


Figure 3.6: Comparison between theoretical and numerical solutions for water surface elevation with nonlinear effects

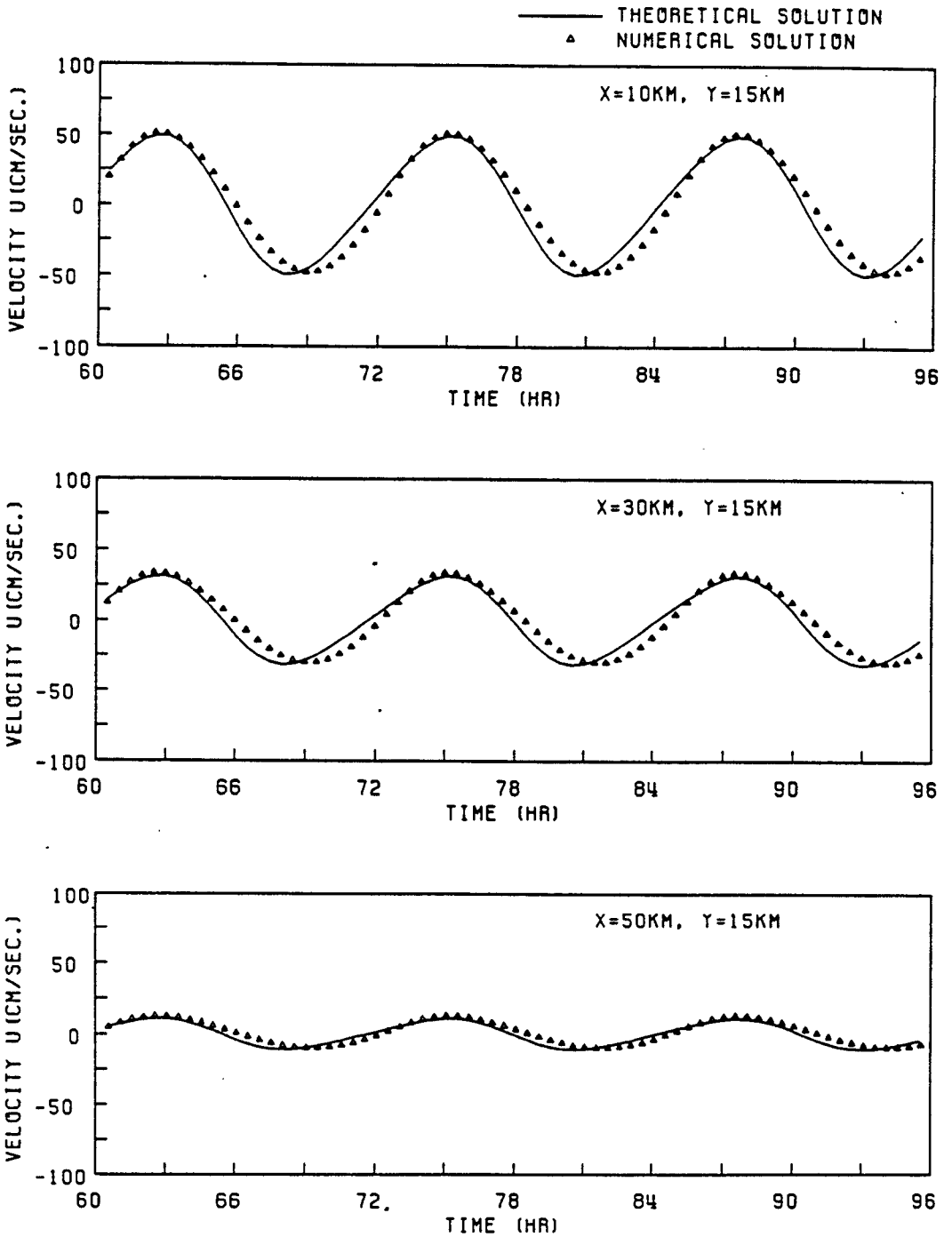


Figure 3.7: Comparison between theoretical and numerical solutions for velocity with nonlinear effects

Details on the derivation of Z_2 and U_2 is presented in appendix B.

The same basin and forcing conditions of the preceding section will now be used to compare theoretical solutions with numerical results obtained with a time step of 30 minutes. The comparisons between the numerical results and the theoretical solutions are shown in Figs. 3.6 and 3.7. Figure 3.6 presents water surface elevations near the mouth, at the middle point, and near the closed boundary of the rectangular basin. Figure 3.7 shows velocities near the mouth, at the middle point, and near the closed boundary. From Figs. 3.6 and 3.7, it can be seen that the numerical results are reasonably similar to the theoretical solutions.

3.3 Comparison to a Theoretical Solution with Coriolis Effect

Neglecting convection, diffusion, and bottom friction, the two-dimensional linear shallow water equations can be written as

$$\frac{\partial U}{\partial t} + c^2 \frac{\partial Z}{\partial x} - \Omega V = 0 \tag{3.33}$$

$$\frac{\partial V}{\partial t} + c^2 \frac{\partial Z}{\partial y} + \Omega U = 0 \tag{3.34}$$

$$\frac{\partial Z}{\partial t} + \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y} = 0 \tag{3.35}$$

where Ω is the Coriolis parameter, and $c = \sqrt{gh}$ in which h is assumed to be constant. Referring to the system shown in Fig. 3.1, the boundary conditions associated with Eqs. (3.33), (3.34) and (3.35) are

$$Z(0, y, t) = Z_m \tag{3.36}$$

$$U(l, y, t) = 0 \tag{3.37}$$

$$V(x, 0, t) = 0 \tag{3.38}$$

$$V(x, b, t) = 0 \tag{3.39}$$

where Z_m is the forced water surface elevation at the mouth of the basin, l and b are the length and width of the basin, respectively.

By using the concepts of Kelvin waves and spectrum of Poincare waves, Rahman (1982) proposed the theoretical solutions of Eqs. (3.33), (3.34) and (3.35) as follows:

$$\begin{aligned} Z &= A\,exp[\frac{\Omega k}{\omega}y + ik(l-x) - i\omega t] \\ &+ AR\,exp[\frac{\Omega k}{\omega}(b-y) - ik(l-x) - i\omega t] \\ &+ \sum_{n=1}^{\infty}\{C_n[\cos K_{1n}(b-y) + \frac{\Omega K_{2n}}{\omega K_{1n}}\sin K_{1n}(b-y)] \\ &\quad exp[-iK_{2n}(l-x) - i\omega t]\} \end{aligned} \tag{3.40}$$

$$\begin{aligned} U &= -\frac{Akc^2}{\omega}exp[\frac{\Omega k}{\omega}y + ik(l-x) - i\omega t] \\ &+ \frac{ARkc^2}{\omega}exp[\frac{\Omega k}{\omega}(b-y) - ik(l-x) - i\omega t] \\ &+ \frac{c^2}{\omega}\sum_{n=1}^{\infty}\{C_n[K_{2n}\cos K_{1n}(b-y) + \frac{\Omega\omega}{K_{1n}c^2}\sin K_{1n}(b-y)] \\ &\quad exp[-iK_{2n}(l-x) - i\omega t]\} \end{aligned} \tag{3.41}$$

$$\begin{aligned} V &= -\frac{ic^2\omega}{\omega^2-\Omega^2}\sum_{n=1}^{\infty}\{C_nK_{1n}(1 + \frac{\Omega^2K_{2n}^2}{\omega^2K_{1n}^2})\sin K_{1n}(b-y) \\ &\quad exp[-iK_{2n}(l-x) - i\omega t]\} \end{aligned} \tag{3.42}$$

where A is the Kelvin wave amplitude at $x = l$ and $y = 0$, R is the reflection coefficient of Kelvin waves, k and ω are wave number and frequency of Kelvin waves, respectively, K_{1n} and K_{2n} are wave numbers of the nth Poincare mode with respect to the y- and x-directions, respectively, and C_n is the amplitude of the nth Poincare mode. The dispersion relationship for Kelvin waves is

$$\omega^2 = k^2c^2 \tag{3.43}$$

For Poincare waves, we have

$$K_{1n}^2 + K_{2n}^2 = \frac{\omega^2 - \Omega^2}{c^2} \tag{3.44}$$

From the boundary condition of $V(x,0,t) = 0$, it is easy to obtain

$$K_{1n} = \frac{n\pi}{b} \tag{3.45}$$

The unknown coefficients R and C_n are obtained when the boundary condition $U(l,y,t) = 0$ is satisfied. Thus we have

$$-Ak\,exp(\frac{\Omega k}{\omega}y) + ARk\,exp[\frac{\Omega k}{\omega}(b-y)]$$
$$+\sum_{n=1}^{\infty} C_n[K_{2n}\cos K_{1n}(b-y) + \frac{\Omega\omega}{K_{1n}c^2}\sin K_{1n}(b-y)] = 0 \qquad (3.46)$$

The simple way to solve for the unknown coefficients R and C_n is through the matrix method. Let us truncate the sum in Eq. (3.46) at the Nth term; the number of unknown coefficients is then $N+1$ (R and $C_1, \ldots, C_n$). If Eq. (3.46) is made to hold at $N+1$ points on (l,y), we then have $N+1$ nonhomogeneous linear equations for the same number of unknowns. Solutions are readily obtained by inversing the matrix of the coefficients.

The rectangular basin defined previously is used to compare the tidal responses inside the basin calculated from the theory and the numerical model. The Coriolis parameter Ω is chosen to be 10^{-4}. The period of the forcing tide at the mouth of the basin is 12 hrs and the amplitude of the Kelvin wave is 50 cm. Given this basic information, the theoretical solutions for the tidal response inside the basin can be obtained by choosing the real parts of Eqs. (3.40), (3.41) and (3.42).

For the numerical computation, the time step is chosen to be 30 minutes and the grid system shown in Fig. 3.2 is employed. The amplitude of the forcing tide at the mouth of the basin is

$$\begin{aligned} Z &= Re\{A\,exp(\frac{\Omega k}{\omega}y + ikl) + AR\,exp[\frac{\Omega k}{\omega}(b-y) - ikl] \\ &+ \sum_{n=1}^{\infty} C_n[\cos K_{1n}(b-y) + \frac{\Omega K_{2n}}{\omega K_{1n}}\sin K_{1n}(b-y)]exp(-iK_{2n}l)\} \end{aligned} \qquad (3.47)$$

which is obtained from Eq. (3.40) by setting $t = 0$. The comparison between the numerical results and the theoretical solutions is shown in Figs. 3.8, 3.9 and 3.10.

Figure 3.8 shows the variation of water surface elevation with time near the mouth, at the middle point, and at the closed end of the basin. Figure 3.9 presents

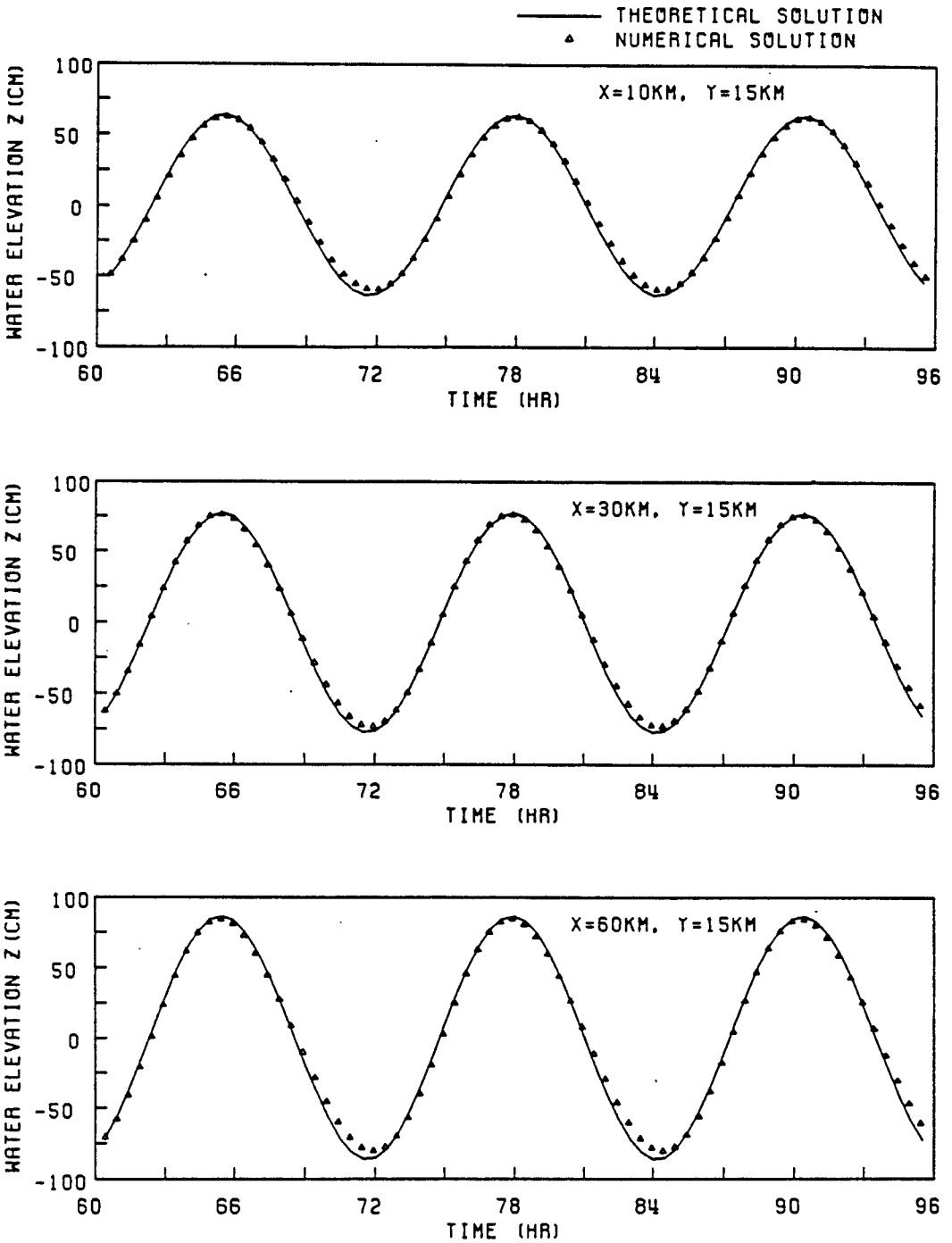


Figure 3.8: Comparison between theoretical and numerical solutions for water surface elevation with coriolis effect

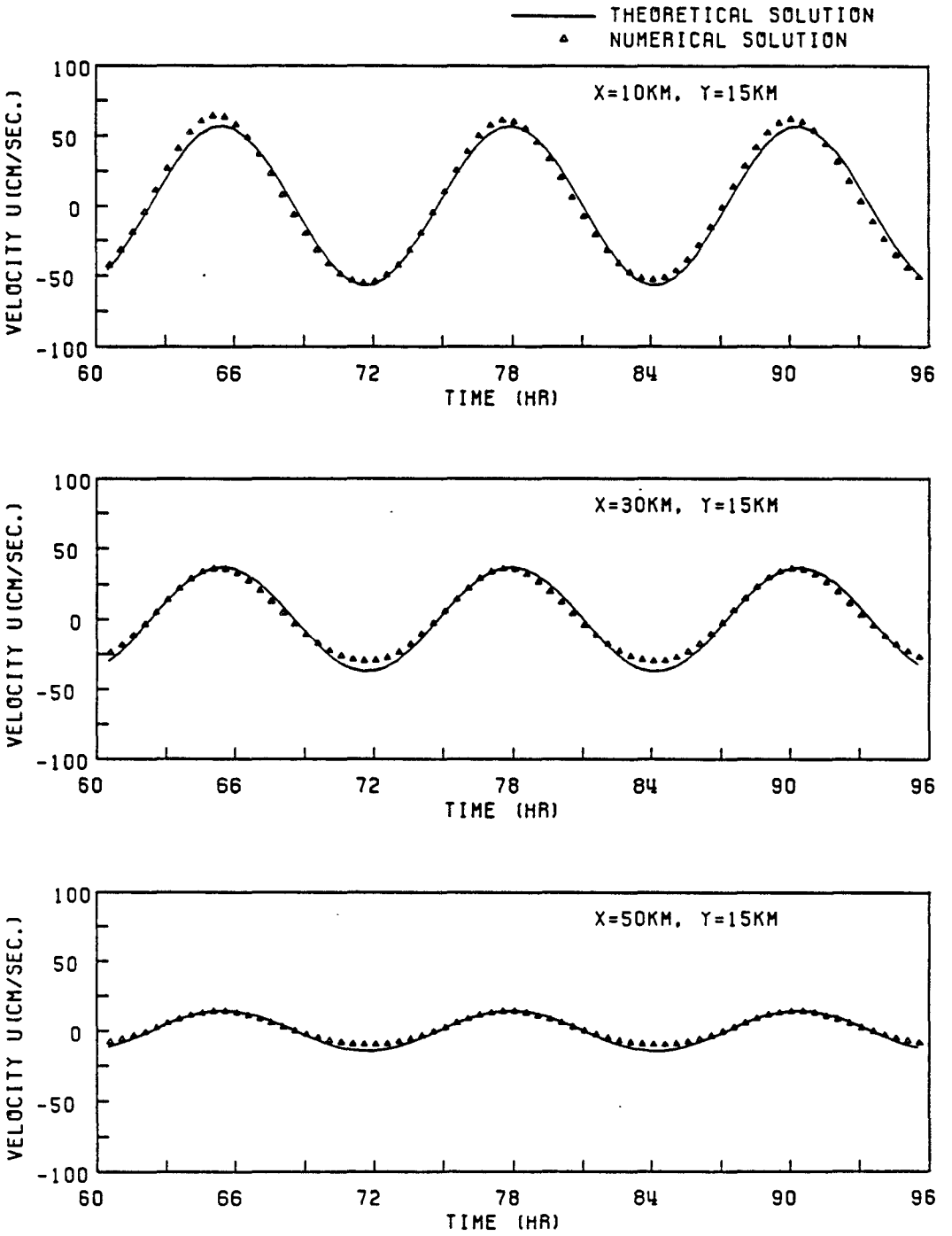


Figure 3.9: Comparison between theoretical and numerical solutions for velocity U with coriolis effect

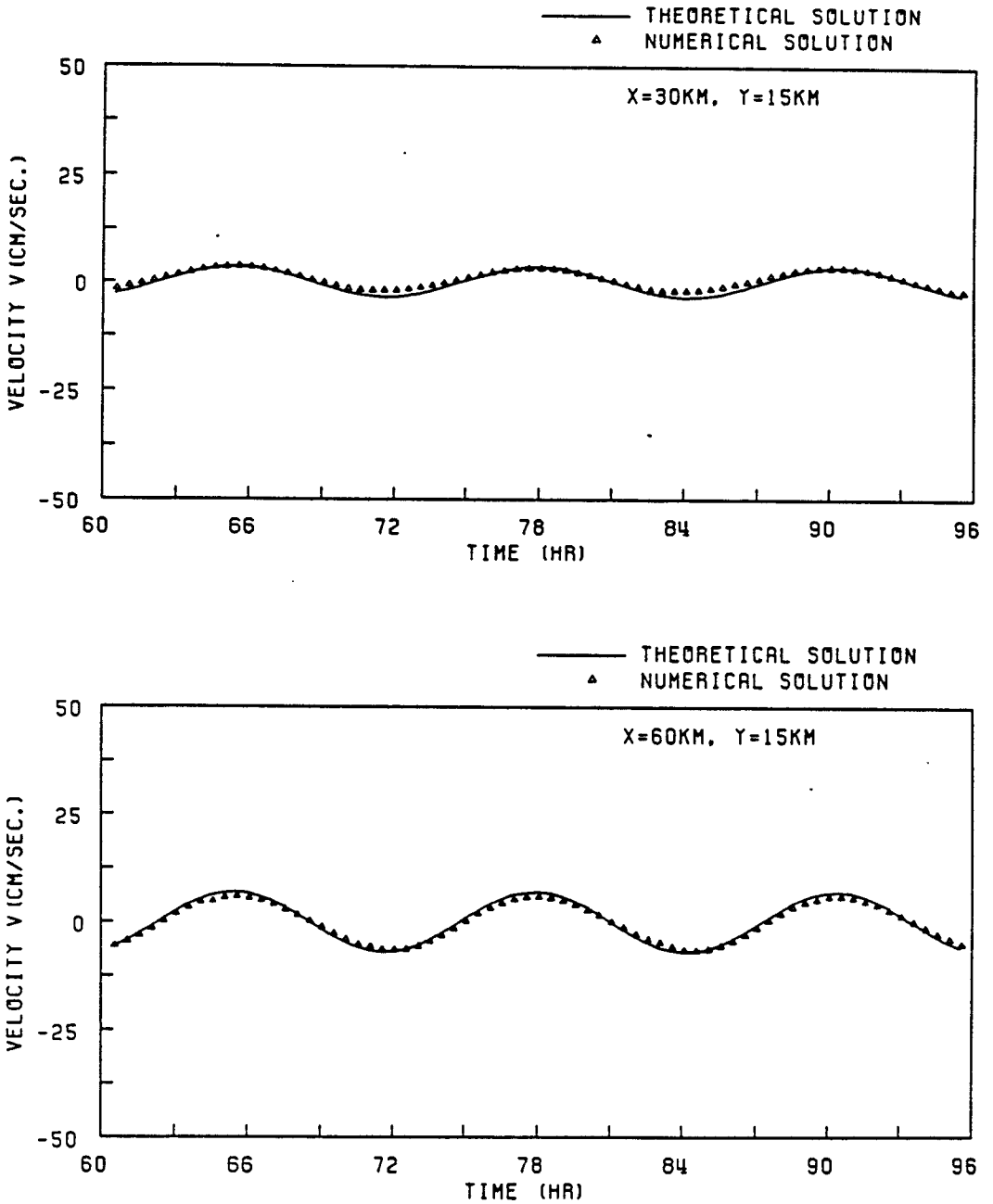


Figure 3.10: Comparison between theoretical and numerical solutions for velocity V with coriolis effect

the flow velocities u in the x-direction at three points in the middle axis of the basin. Figure 3.10 shows the flow velocities v in the y-direction at the middle point and at the closed boundary. Both the theoretical and the numerical solutions at $x = 10km$ are very small, thus are not shown in the figures. These figures illustrate that the model results agree quite well with the theoretical solutions.

3.4 Comparison to a Theoretical Solution with Friction Effect

With the bottom frictions, the two-dimensional shallow water equations can be written as

$$\frac{\partial U}{\partial t} + gh\frac{\partial Z}{\partial x} + \frac{\tau_{bx}}{\rho} = 0 \tag{3.48}$$

$$\frac{\partial V}{\partial t} + gh\frac{\partial Z}{\partial y} + \frac{\tau_{by}}{\rho} = 0 \tag{3.49}$$

$$\frac{\partial Z}{\partial t} + \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y} = 0 \tag{3.50}$$

where ρ is the density of water. Assume that the water depth h is constant and the bottom frictions can be calculated by linear friction formulae

$$\tau_{bx} = \rho FU \tag{3.51}$$

$$\tau_{by} = \rho FV \tag{3.52}$$

where

$$F = \frac{g\sqrt{U_{max}^2 + V_{max}^2}}{C^2 h^2} \tag{3.53}$$

Given these, Eqs. (3.48), (3.49) and (3.50) can be simplified as

$$\frac{\partial U}{\partial t} + c^2\frac{\partial Z}{\partial x} + FU = 0 \tag{3.54}$$

$$\frac{\partial V}{\partial t} + c^2\frac{\partial Z}{\partial y} + FV = 0 \tag{3.55}$$

$$\frac{\partial Z}{\partial t} + \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y} = 0 \tag{3.56}$$

Subject to the boundary conditions defined by Eqs. (3.36), (3.37), (3.38) and (3.39), Rahman (1982) proposed the theoretical solutions of Eqs. (3.54), (3.55) and (3.56) as follows

$$\begin{aligned} Z &= Re\{A_0(\cos kx + \tan kl \sin kx)exp(-i\omega t) \\ &+ \sum_{n=1}^{\infty}[A_n \cos K_{1n}(b-y) \\ &(\cos K_{2n}x + \tan K_{2n}l \sin K_{2n}x)]exp(-i\omega t)\} \end{aligned} \tag{3.57}$$

$$\begin{aligned} U &= Re\{\frac{-c^2}{F-i\omega}A_0k(-\sin kx + \tan kl \cos kx)exp(-i\omega t) \\ &+ \frac{-c^2}{F-i\omega}\sum_{n=1}^{\infty}[A_nK_{2n} \cos K_{1n}(b-y) \\ &(-\sin K_{2n}x + \tan K_{2n}l \cos K_{2n}x)]exp(-i\omega t)\} \end{aligned} \tag{3.58}$$

$$\begin{aligned} V &= Re\{\frac{-c^2}{F-i\omega}\sum_{n=1}^{\infty}[A_nK_{1n} \sin K_{1n}(b-y) \\ &(\cos K_{2n}x + \tan K_{2n}l \sin K_{2n}x)]exp(-i\omega t)\} \end{aligned} \tag{3.59}$$

where b and l are the width and length of the basin, respectively, and

$$K_{1n} = \frac{n\pi}{b} \tag{3.60}$$

$$k^2 = \frac{\omega^2}{c^2}(1 + i\frac{F}{\omega}) \tag{3.61}$$

$$K_{2n}^2 = \frac{\omega^2}{c^2}(1 + i\frac{F}{\omega}) - (\frac{n\pi}{b})^2 \tag{3.62}$$

The coefficients $A_0, \ldots, A_n$ can be obtained using the condition

$$Z(0, y, t) = Z_m\, exp(-i\omega t) \tag{3.63}$$

at $x = 0$. They are given as

$$A_0 = \int_0^b Z_m dy \tag{3.64}$$

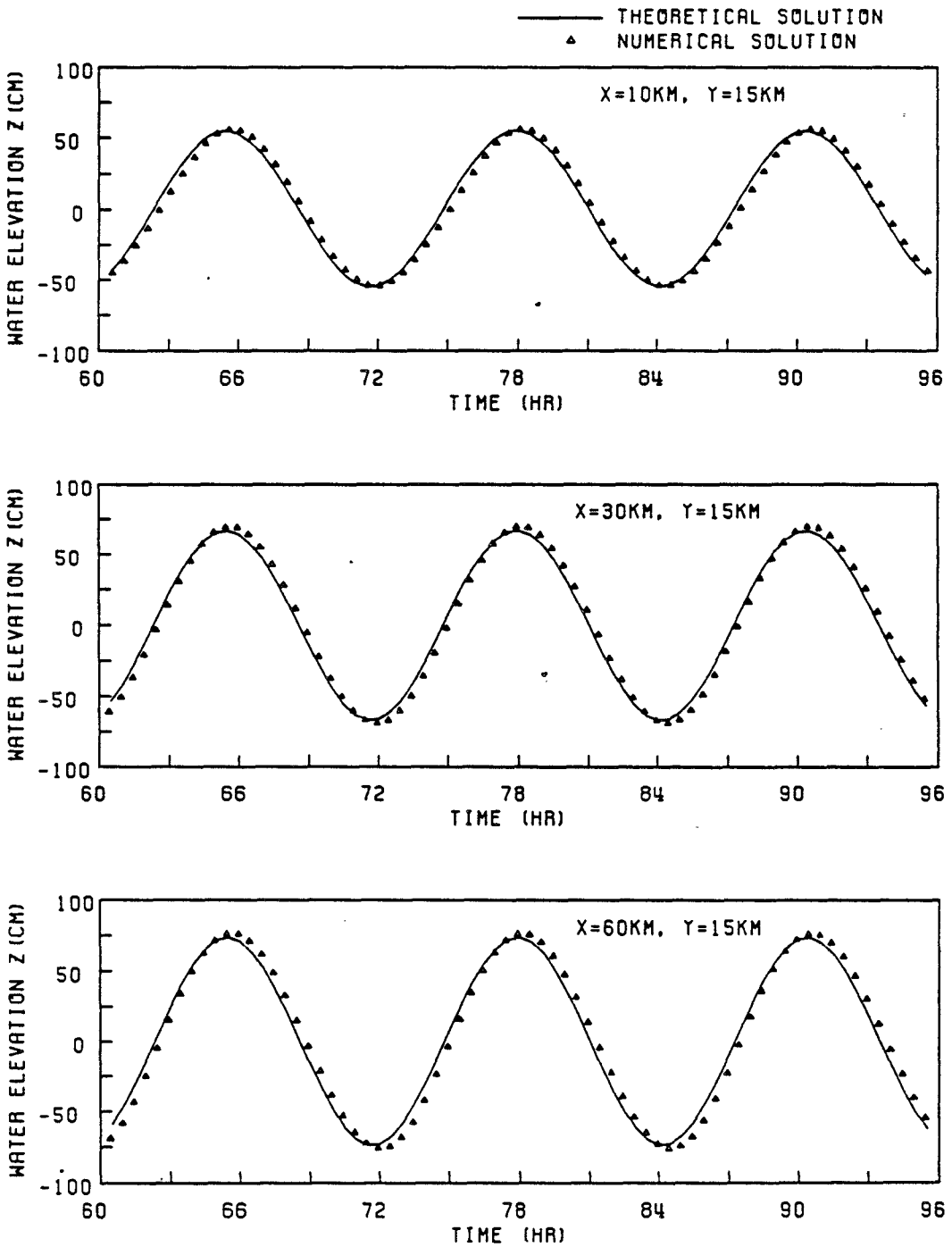


Figure 3.11: Comparison between theoretical and numerical solutions for water surface elevation with friction effect

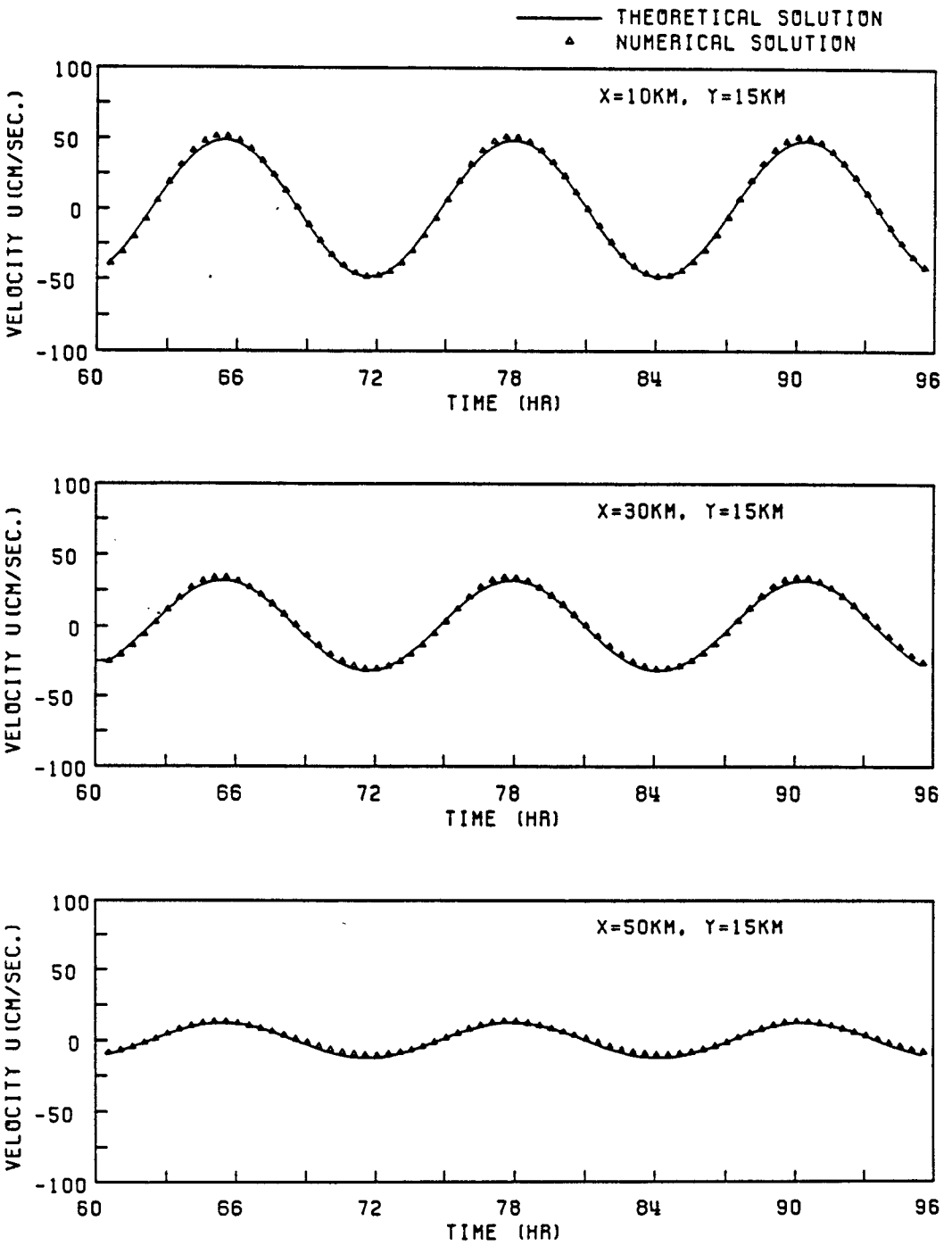


Figure 3.12: Comparison between theoretical and numerical solutions for velocity U with friction effect

$$A_n = \int_0^b Z_m \cos K_{1n}(b-y)dy \tag{3.65}$$

To compare the theory with the numerical results, the same basin defined previously is used. The amplitude of the forcing tide along the mouth of the basin is assumed to be constant (50 cm), which leads to no flow in the y-direction, i.e. $V = 0$, and zero value for the coefficients A_1, ..., A_n. The period of the forcing tide is 12 hrs. The Manning coefficient is 0.02 and the maximum velocity is 50 cm/sec.. Thus the theoretical solutions for the tidal responses inside the basin can be easily obtained from Eqs. (3.57) and (3.58). With a time step of 30 minutes, the numerical results can be obtained by the use of the grid system shown in Fig. 3.2. The comparisons between the theory and the model results are shown in Figs. 3.11 and 3.12.

Figure 3.11 shows the variation of the water surface elevations with time at three points inside the basin. And Fig. 3.12 presents the comparison for the velocities. From both figures, it is seen that the numerical results are very close to the theoretical ones.

CHAPTER 4
NUMERICAL SIMULATION OF FLOW OVER TIDAL FLATS

In the computation of flow over tidal flats, the difficulty is to simulate properly the shoreline boundary which moves with time. The work associated with it includes the determination of the instantaneous location of the shoreline and the implementation of boundary conditions. In this chapter, two ways to treat this problem will be discussed in detail, also the wave propagation on a sloping beach will be theoretically and numerically studied to investigate the ability of these two methods to simulate the moving boundary problems.

4.1 Properties of a Moving boundary

The moving boundary has the basic property that it can move with time. Its movement is mainly controlled by the topography of the coastal region, the tidal amplitude, the water level associated with a storm surge, etc. Based on the Lagrangian description of fluid motion, the movement of a boundary can be mathematically expressed as

$$\vec{X}(t) = \vec{X}_0 + \int_0^t \vec{v}_b \, dt \tag{4.1}$$

where $\vec{X}(t)$ is the location of the boundary at the time t, $\vec{X}_0$ is the initial location, and $\vec{v}_b$ is the velocity of the boundary motion. The basic boundary conditions on the moving boundary are

$$h = 0 \tag{4.2}$$

$$\vec{v} = \vec{v}_b \tag{4.3}$$

where h is total water depth and $\vec{v}$ is the velocity of a fluid particle on the moving boundary.

Although the mathematical expressions for the movement of the boundary are simple, it is difficult to couple numerically the boundary movement to the main numerical model. The first problem which arises is the difficulty in simulating the continuous boundary motion with conventional finite difference techniques. Without a generalized finite difference formulation for irregular grids, it is impossible to consider the continuous boundary motion.

The second problem is that it is difficult to compute the velocity of the moving boundary, which is governed by the topography and the water surface gradient. Usually the topography is given, but the water surface gradient on the moving boundary is unknown and is related to the velocity of the boundary motion implicitly.

The third problem arises from the computation of the bottom friction in the vicinity of a moving boundary. The most common bottom friction formulation in vertically-integrated shallow water problems is the Manning-Chezy formulation. This formulation provides finite bottom friction throughout the interior of the computational domain. However, it breaks down near the moving boundary since the computed bottom friction approaches infinity as the water depth tends to zero. Using the conventional finite difference model, the computed strong bottom friction causes problems both in the flooding and drying. During the flooding, the strong bottom friction leads to very sharp surface slope which may lead to very large velocity and cause numerical instability. During the drying, the strong bottom friction leads to very slow water motion locally.

4.2 Past Study

Several numerical models for simulating the moving boundary problem have been proposed in the past. Most of them are developed using the finite difference technique and finite element technique with fixed grids or deforming grids. The technique that makes use of the fixed grids usually treats the moving boundary by turning cells on or off at the boundary based on the mass conservation. Although

this technique is simpler to implement than the technique that makes use of deforming grids, it possesses other problems. The impulsive filling of a cell with fluid often leads to numerical problems unless treated very carefully.

Reid and Bodine (1968) investigated the transient storm surges in Galveston Bay, Texas. Omitting nonlinear advection and Coriolis terms, they developed a finite difference numerical model with the inclusion of the tidal flat. A uniform Cartesian mesh and staggered grid system were used. The elevation of the sea bed or land was represented by a constant value at each grid point within a square grid. Hence, the actual topography was approximated in a stair-step fashion. The movement of the boundary was controlled by the water elevation. If the water elevation in a flooded square was less than the base elevation of an adjacent dry square, then a zero normal flow boundary condition was applied along their common boundary. However, if the water elevation in a flooded square was greater than that of an adjacent dry square, then the water was permitted to flow into the dry square. The flow rate between two squares was determined using an empirical equation for flow over a broad-crested barrier. The overtopping of a barrier could be treated also. However, the model could treat only barriers aligned along the grid mesh division. Flow across the barrier was permitted when the water height on one side exceeded the barrier height. If the water height exceeded the barrier height on both sides, then the flow rate was determined using an empirical equation for flow over a submerged weir. The empirical coefficients in the model were determined by iteration, comparing the model with tidal data and data from hurricane Carla (September 9-12, 1963). The gross features of the inundation were predicted reasonably well. However, it should be noted that the empirical coefficients used could be very site-dependent.

Yeh and Yeh (1976) developed a nonlinear nondispersive moving boundary model for simulating storm surge using an ADI technique. The shoreline in this

numerical model moved as the flow inundated low lying land. However, details of the treatment of the boundary were not given. It appears that the shoreline advanced or retreated in discrete increments of grid cells. They reported good agreement with field data.

Yeh and Chou (1979) developed a nonlinear nondispersive finite difference surge model using an explicit technique with reference to a fixed grid system. The boundary between dry land and the water was simulated as a discrete moving boundary, i.e. the boundary moves in discrete jumps. It advances or retreats according to the rise or recession of the surge level. During the rising surge, a new grid was added to the computations if the surge elevation of any of its neighbors was above the base elevation of that grid point. During the receding surge, a grid point was taken out of the computations if its total water depth was decreased to a preset value. If any of its neighboring points still has a surge elevation above its bottom, this grid point was kept in the computations. They compared the model to the field results and also with a similar numerical model, which used a fictitious vertical wall instead of a sloping shoreline. They reported that their model showed much better agreement with field data than the fixed boundary model. The fixed boundary model predicted up to 30 percent higher surge levels than their moving boundary model. The discrepancy was greater for higher surge values. They explained the discrepancy as being due to the storage effect of the inland region where water can accumulate but which is not part of the computational domain of the fixed boundary model.

Hirt and Nichols (1981) proposed a method of treating the free boundary which was similar to the marker particle method. They called this method the volume of fluid (VOF) technique. According to this technique, they defined a step function, F, whose value is unity at any point occupied by fluid and zero otherwise. The average value of F in a cell would then represent the fractional volume of the cell occupied by fluid. In particular, a unity value of F would correspond to a cell full

of fluid, while a zero value would indicate that the cell contains no fluid. Cells with F values between zero and one must then contain a free surface. The location of the free boundary in a boundary cell is determined in terms of the F value and the normal direction of the boundary. The normal direction to the boundary is thought to be the direction in which the value of F changes most rapidly. The F field is governed by the equation which states that F moves with the fluid at any time. This equation is solved by the donator- acceptor iteration. With this method to simulate the free boundary, they developed a finite difference free surface model with a variable rectangular mesh using an iteration scheme. The model was applied to the broken dam, undular bore and breaking bore problems, etc. They reported good agreements with the experimental results.

Some French researchers (Benque et al., 1982) developed a finite difference numerical model with the inclusion of the tidal flat using the method of fractional steps. They solved shallow water equations through three steps, which are advection, diffusion and propagation steps. A different numerical scheme was applied at each computational step. The treatment of the boundary motion was considered at the propagation step. The dry land region was assumed to be covered by a thin water layer. The flow in this region was governed by the bottom friction. During the computation of the propagation, the shallow water equations were first applied to the whole computational domain including the region of the thin water layer. Then the solution in the vicinity of the moving boundary needed to be adjusted to satisfy the resistance flow. The Manning-Chezy friction formulation was used to compute the bottom friction. They reported that the moving boundary model developed in this way violated the continuity equation slightly. Good agreement of the numerical results with the measured data were presented based on the model application to the Bay of Saint Brieuc and the River Canche Estuary, France.

Lynch and Gray (1980) outlined a general technique whereby a moving boundary

can be treated by finite element Eulerian models. The finite element basis functions are chosen to be functions of time so that the element boundaries track the moving shoreline. They showed how this motion generates extra terms which, if treated properly, reduce the problem to one that can be treated by standard finite element procedures. They showed how to apply the method to treat the propagation and runup of long waves. They looked at two simple problems involving the runup of waves on plane beaches. They showed that estimating the runup by extrapolating the wave height at a vertical wall could introduce significant errors. They also pointed out that treating deforming elements is computationally more expensive than fixed ones, and recognized that potential problem can arise if the mesh becomes geometrically too distorted.

Gopalakrishnan and Tung (1983) described a finite-element nonlinear long wave runup model valid only for one horizontal dimension. The model contained terms that accounted for vertical accelerations. The moving shoreline was handled by allowing the shoreline element to deform so that the beach node always tracked the shoreline. If the shoreline element became too stretched, it split into two elements. The element containing the shoreline node continued to deform but the other new element created by splitting stayed fixed. They showed some plots that detailed the runup process, but they did not present any results about the rundown process. The technique outlined by the authors seems applicable to tsunami runup, but they did not present a thorough or convincing argument to show that their model could be used reliably for such studies. It should be noted that the technique in their work cannot be extended easily to include two horizontal dimensions since the element-splitting procedure would be very complex.

4.3 Numerical Treatment of a Moving Boundary

In this section, two ways to simulate the moving boundary problem will be studied in detail. The first method was proposed by Reid and Bodine, and the

second method was proposed by Benque et al. (1982).

4.3.1 Method to Treat a Moving Boundary with the Weir Formulation

Reid and Bodine (1968) treated the continuous moving boundary as a discrete moving boundary. In their scheme, a discrete Cartesian grid system is used and the actual topography in the vicinity of a moving boundary is approximated by two-dimensional stair-steps. Thus the elevation of the sea bed or land can be regarded as uniform over each grid square.

The boundary motion is controlled by the water level. During the flooding, a grid point is added to the computational system if its water depth is greater than a preset value h_1 which is the minimum water depth for the effective application of the Navier-Stokes equations. The value of h_1 also depends on the topography and the grid system and is usually taken to be 10 to 20 cm. During the drying, a grid point is taken out of the computation if its water depth is decreased to a preset value, h_2, which is usually slightly different from h_1.

In the computation of flooding, the condition of no normal mass flux is applied at the moving boundary when the water elevation is less than that of the adjacent dry land, i.e., the normal component of flow, Q_n, at the juncture of a flooded cell and a dry cell is taken as zero, while the tangential component of flow may satisfy the no-slip or free-slip condition. However, if the water elevation is greater than that of the dry land, flow will be allowed to flood into the dry cell until the water depth in this cell reaches h_1. The mass flux per unit width of the dry cell, Q_n, can be calculated by the weir formulation (Reid and Bodine, 1968) as:

$$Q_n = C_0 D \sqrt{g \mid Z_d - Z_c \mid} \tag{4.4}$$

where Z_d and Z_c are the water surface elevation in the donator and acceptor cells, respectively; C_0 is an empirical dimensionless coefficient which was suggested to be

less than 0.5 by Reid and Bodine (1968); and

$$D = Z_d - Z_b \tag{4.5}$$

where Z_b is the bottom elevation of the acceptor cell. During the period of Δt, the increment of water level in the acceptor cell, ΔZ_c, is

$$\Delta Z_c = \frac{Q_n \Delta t B}{A_c} \tag{4.6}$$

where A_c is the area of the acceptor cell and B is the width of the acceptor cell. The decrement of water level at the donator cell, ΔZ_d, is

$$\Delta Z_d = \frac{Q_n \Delta t B}{A_d} \tag{4.7}$$

where A_d is the area of the donator cell. During the flooding computation of each time step, each donator cell must be examined to see if too much water is flooded from the donator cell to the acceptor cell. If the water level in the donator cell is less than that in the acceptor cell, the mass flux computed by Eq. (4.4) must be adjusted to make the water level in the donator cell the same as that in the acceptor cell.

When the water depth becomes small during the drying, an artificial water depth has to be considered so that the Manning-Chezy friction formulation can still be used to compute the bottom friction. The value of this artificial water depth is determined empirically. Usually we can set it to be 20 cm.

The development of a two-dimensional moving boundary model in this manner is very cumbersome since the location of the new boundary must be determined at each time step. But if the grids in the vicinity of the moving boundary and the time step are kept small, we can obtain reasonable numerical solutions for the moving boundary.

4.3.2 Method to Treat a Moving Boundary With a Thin Water Layer

Based on an assumption that the water flow in the region of shallow water is dominated by the bottom friction, Benque et al. (1982) proposed a way to

treat the moving boundary by controlling the water flow discharges through the adjustment of the water depth. In this scheme, a grid system is first established in the computational domain which includes the entire tidal flat region. To ensure numerical accuracy, grid sizes in the tidal flat region should be smaller than those in the main computational domain.

A thin water layer is assumed to exist at all times over the dry land region so that all grids in the computational domain are always wet. Thus, no grid needs to be taken out of the computational domain during the computation and we do not need to consider the motion of the boundary. The shoreline boundary can thus be treated simply as a fixed boundary, so the Navier-Stokes equations of fluid motion can be applied to the entire computational domain so long as the bottom friction is adequately resolved for small water depths. From the Manning-Chezy friction formulation, we see that the water depth plays an important role in the computation of the bottom friction, so it is possible to "control" the bottom friction by adjusting the water depth.

As seen in Chapter 2, the free surface elevation Z does not appear during the advection and diffusion steps. Since the motion of the boundary is mainly controlled by the water elevation, it is only necessary to consider the propagation step. From the governing equations, Eqs. (2.75) and (2.76), of the propagation step, it is seen that the water depth, h, and the increment of water surface elevation during one time step, ΔZ, at the points of $i \pm 1/2$ and $j \pm 1/2$ must be calculated. They are governed by the water depth and the surface elevation at their upstream and downstream grid points, and can be calculated by the following formulae:

$$\left.\begin{aligned} h_{i+1/2} &= \gamma_i\, h_i + (1-\gamma_i) h_{i+1} \\ h_{i-1/2} &= \gamma_{i-1}\, h_{i-1} + (1-\gamma_{i-1}) h_i \\ \Delta Z_{i+1/2} &= \gamma_i\, \Delta Z_i + (1-\gamma_i) \Delta Z_{i+1} \\ \Delta Z_{i-1/2} &= \gamma_{i-1}\, \Delta Z_{i-1} + (1-\gamma_{i-1}) \Delta Z_i \end{aligned}\right\} \tag{4.8}$$

and

$$\left.\begin{aligned} h_{j+1/2} &= \gamma_j\, h_j + (1-\gamma_j) h_{j+1} \\ h_{j-1/2} &= \gamma_{j-1}\, h_{j-1} + (1-\gamma_{j-1}) h_j \\ \Delta Z_{j+1/2} &= \gamma_j\, \Delta Z_j + (1-\gamma_j) \Delta Z_{j+1} \\ \Delta Z_{j-1/2} &= \gamma_{j-1}\, \Delta Z_{j-1} + (1-\gamma_{j-1}) \Delta Z_j \end{aligned}\right\} \tag{4.9}$$

where γ is a weighting coefficient. Normally the shape of the water surface between two successive grid points can be assumed to be a straight line, so that γ can be set to 0.5. However, in the regions where the water flow is dominated by the bottom resistance, γ cannot be equal to 0.5 since the shape of the water surface is greatly curved. Therefore we need to look for a formulation to calculate it.

Neglecting the effect of the interior force, we can obtain the governing equations for the flow dominated by the bottom resistance as follows

$$gh\frac{\partial Z}{\partial x} + \frac{\tau_{bx}}{\rho} = 0 \tag{4.10}$$

$$gh\frac{\partial Z}{\partial y} + \frac{\tau_{by}}{\rho} = 0 \tag{4.11}$$

Using the Manning-Chezy friction formulation,

$$\tau_{bx} = \frac{\rho g U\sqrt{U^2+V^2}}{C^2\, h^2} \tag{4.12}$$

$$\tau_{by} = \frac{\rho g V\sqrt{U^2+V^2}}{C^2\, h^2} \tag{4.13}$$

we can rewrite Eqs. (4.10) and (4.11) as

$$\frac{\partial Z}{\partial x} + \frac{U\sqrt{U^2+V^2}}{C^2 h^3} = 0 \tag{4.14}$$

$$\frac{\partial Z}{\partial y} + \frac{V\sqrt{U^2+V^2}}{C^2 h^3} = 0 \tag{4.15}$$

For flow controlled by the bottom resistance, it can be assumed that the discharge should always increase when the downstream level decreases, i.e,

$$\frac{\partial U}{\partial Z_{ds}} \leq 0 \tag{4.16}$$

$$\frac{\partial V}{\partial Z_{ds}} \leq 0 \tag{4.17}$$

Where the subscript ds refers to the downstream. Rewriting Eqs. (4.14) and (4.15) in the discrete form, we have

$$\frac{Z_{ds} - Z_{us}}{\Delta x_i} + \frac{U\sqrt{U^2+V^2}}{C^2 h_{i+1/2}^3} = 0 \tag{4.18}$$

$$\frac{Z_{ds} - Z_{us}}{\Delta y_j} + \frac{V\sqrt{U^2+V^2}}{C^2 h_{j+1/2}^3} = 0 \tag{4.19}$$

where the subscript us refers to the upstream. In Eq. (4.18), the upstream and downstream correspond to flow in the x-direction, but in Eq. (4.19) they refer to flow in the y-direction. Introducing a coefficient β, the water depth h in Eqs. (4.18) and (4.19) can be expressed as

$$h_{i+1/2} = \beta\, h_{us} + (1-\beta) h_{ds} \tag{4.20}$$

$$h_{j+1/2} = \beta\, h_{us} + (1-\beta) h_{ds} \tag{4.21}$$

where

$$h_{us} = Z_{us} - ZB_{us} \tag{4.22}$$

$$h_{ds} = Z_{ds} - ZB_{ds} \tag{4.23}$$

in which ZB_{us} and ZB_{ds} are the bottom elevations upstream and downstream of the point $i+1/2$ or $j+1/2$, respectively. Differentiating Eq. (4.18) with respect to Z_{ds} and bearing in mind that U is a function of Z_{us} and Z_{ds}, and V is constant, we obtain

$$\frac{\partial U}{\partial Z_{ds}}\left(\sqrt{U^2+V^2} + \frac{U^2}{\sqrt{U^2+V^2}}\right) = \frac{C^2 h^2}{\Delta x_i}\left[-h + 3(Z_{us} - Z_{ds})\frac{\partial h}{\partial Z_{ds}}\right] \tag{4.24}$$

Applying the condition $\frac{\partial U}{\partial Z_{ds}} \leq 0$, we get

$$-h + 3(Z_{us} - Z_{ds})\frac{\partial h}{\partial Z_{ds}} \leq 0 \tag{4.25}$$

From Eq. (4.20), we know

$$\frac{\partial h}{\partial Z_{ds}} = 1 - \beta \tag{4.26}$$

Substituting $\frac{\partial h}{\partial Z_{ds}}$ and h into inequality (4.25) yields

$$-[\beta h_{us} + (1-\beta)h_{ds}] + 3(1-\beta)(Z_{us} - Z_{ds}) \leq 0 \tag{4.27}$$

So we obtain

$$\beta \geq \frac{3(Z_{us} - Z_{ds}) - h_{ds}}{3(Z_{us} - Z_{ds}) + h_{us} - h_{ds}} \tag{4.28}$$

Differentiating Eq. (4.19) with respect to Z_{ds}, treating U as a constant and following the same procedure as above, we can get the same inequality for β as in (4.28).

The weighting coefficient γ in Eq. (4.8) for the propagation step must be replaced by either β or $1-\beta$ in the shallow water region. Since the direction of flow during flooding is different from that during drying, the concepts of upstream and downstream are changed. Thus during flooding, $\gamma = \beta$ and during drying $\gamma = 1-\beta$. If β computed from Eq. (4.28) is less than 0, it can be set equal to 0. If it is greater than 1, it is set equal to 1. In general, β is 0.5 in the computational domain except in the transitional region between the thin water layer and the deep water region.

For the flow controlled by the bottom friction, we can obtain the maximum discharge, U_{max} and V_{max}, from Eqs. (4.14) and (4.15)

$$U_{max} = sign(u)\sqrt{C^2h^3 \mid \frac{\partial Z}{\partial x} \mid} \tag{4.29}$$

$$V_{max} = sign(v)\sqrt{C^2h^3 \mid \frac{\partial Z}{\partial y} \mid} \tag{4.30}$$

in which

$$sign(u) = -\frac{\partial Z}{\partial x} / \mid \frac{\partial Z}{\partial x} \mid \tag{4.31}$$

$$sign(v) = -\frac{\partial Z}{\partial y} / \mid \frac{\partial Z}{\partial y} \mid \tag{4.32}$$

After the propagation step, the new state of the model is known. Corresponding to the water surface gradient at this state, we can calculate the maximum discharge from the above two equations. Also we have discharges U and V at this state which are computed from the momentum and continuity equations. If U or V is greater

than U_{max} or V_{max}, U or V must be replaced by U_{max} or V_{max} before the computation proceeds to the next time step.

It should be noted that using this way to develop a moving boundary model leads to relatively simple computer program since we do not need to simulate the motion of the boundary. But the continuity equation is slightly violated by always maintaining a thin water layer on the dry land region and replacing the discharges U and V by U_{max} and V_{max}. The thickness of the water layer on the dry land region is determined by the requirement that the computational cell cannot become dry during one time step.

4.4 Theoretical Solution of Wave Propagation on a Sloping Beach

In this section, the theoretical solution for the wave propagation on a linearly sloping beach obtained by Carrier and Greenspan (1958) will be presented briefly. Referring to the system shown in Fig. 4.1, the one- dimensional nonlinear shallow water equations can be written as

$$\frac{\partial \eta^*}{\partial t^*} + \frac{\partial}{\partial x^*}[(\eta^* + h^*)u^*] = 0 \tag{4.33}$$

$$\frac{\partial u^*}{\partial t^*} + u^*\frac{\partial u^*}{\partial x^*} + g\frac{\partial \eta^*}{\partial x^*} = 0 \tag{4.34}$$

where the asterisks denote dimensional quantities, η is the displacement of water surface above the mean water level, h is the still water depth which varies linearly with x, u is the velocity in the x direction. Let L be the characteristic horizontal length scale of the wave motion. Then we can define a time scale, T, and velocity scale, u_0, as follows

$$T = \sqrt{L/\phi g} \tag{4.35}$$

$$u_0 = \sqrt{\phi g L} \tag{4.36}$$

where ϕ is the beach angle. Let us choose the following nondimensionalization

$$x = \frac{x^*}{L} \qquad t = \frac{t^*}{T} \qquad \eta = \frac{\eta^*}{\phi L}$$

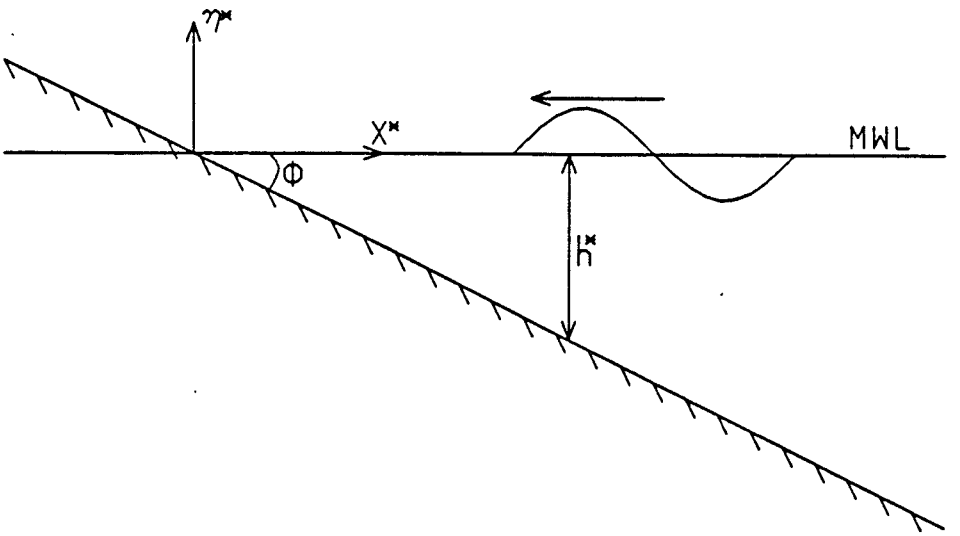


Figure 4.1: Definition sketch for wave propagation on a sloping beach

$$h = \frac{h^*}{\phi L} = x \qquad u = \frac{u^*}{u_0} \tag{4.37}$$

and define

$$c^2 = \frac{h^* + \eta^*}{\phi L} = h + \eta = x + \eta \tag{4.38}$$

With these definitions, Eqs. (4.33) and (4.34) become

$$\eta_t + [(\eta + x)u]_x = 0 \tag{4.39}$$

$$u_t + u\,u_x + \eta_x = 0 \tag{4.40}$$

In terms of u and c, Eqs. (4.39) and (4.40) become

$$2c_t + 2u\,c_x + c\,u_x = 0 \tag{4.41}$$

$$u_t + u\,u_x + 2c\,c_x = 1 \tag{4.42}$$

Through the elegant series of transformations, Carrier and Greenspan (1958) were able to transform this problem, with two coupled nonlinear equations, into a new problem with only one linear equation. A brief derivation will be presented in the following.

If Eqs. (4.41) and (4.42) are added and subtracted, we obtain

$$\frac{d}{dt}(u \pm 2c - t) = 0 \qquad along \qquad \frac{dx}{dt} = u \pm c \tag{4.43}$$

Let us define the characteristic variables ς and ξ by

$$\varsigma = u + 2c - t \tag{4.44}$$

$$\xi = -u + 2c + t \tag{4.45}$$

Hence, Eq. (4.43) becomes

$$\varsigma = const \qquad along \qquad \frac{dx}{dt} = u + c \tag{4.46}$$

$$\xi = const \qquad along \qquad \frac{dx}{dt} = u - c \tag{4.47}$$

Now let us consider x and t to be functions of ς and ξ. Then for $\varsigma = const$ or $\xi = const$ we get

$$\frac{dx}{dt} = \frac{\partial x}{\partial \xi} \Big/ \frac{\partial t}{\partial \xi} \qquad if \qquad \varsigma = const \tag{4.48}$$

$$\frac{dx}{dt} = \frac{\partial x}{\partial \varsigma} \Big/ \frac{\partial t}{\partial \varsigma} \qquad if \qquad \xi = const \tag{4.49}$$

From above two equations, we get

$$x_\xi = t_\xi (u + c) \tag{4.50}$$

$$x_\varsigma = t_\varsigma (u - c) \tag{4.51}$$

From Eqs. (4.44) and (4.45), we can obtain

$$u + c = (3\varsigma - \xi)/4 + t \tag{4.52}$$

$$u - c = (\varsigma - 3\xi)/4 + t \tag{4.53}$$

Substituting $u + c$ and $u - c$ into Eqs. (4.50) and (4.51) yields the transform relationship between (x, t) and (ς, ξ) as follows:

$$x_\xi = t_\xi(3\varsigma - \xi)/4 + (t^2/2)_\xi \tag{4.54}$$

$$x_\varsigma = t_\varsigma(\varsigma - 3\xi)/4 + (t^2/2)_\varsigma \tag{4.55}$$

Eliminating x from Eqs. (4.54) and (4.55), we get

$$2(\varsigma + \xi)t_{\varsigma\xi} + 3(t_\varsigma + t_\xi) = 0 \tag{4.56}$$

This is a linear partial differential equation for $t(\varsigma, \xi)$. It is convenient to introduce new variables σ and λ by

$$\lambda = \xi - \varsigma = 2(t - u) \tag{4.57}$$

$$\sigma = \xi + \varsigma = 4c \tag{4.58}$$

Then Eq. (4.56) becomes

$$t_{\lambda\lambda} = t_{\sigma\sigma} + \frac{3}{\sigma}t_\sigma \tag{4.59}$$

Since from Eq. (4.57), $t = \lambda/2 + u$, u must also satisfy Eq. (4.59)

$$u_{\lambda\lambda} = u_{\sigma\sigma} + \frac{3}{\sigma}u_\sigma \tag{4.60}$$

If we introduce a "potential" $\varphi(\sigma, \lambda)$ so that

$$u = \frac{\varphi_\sigma}{\sigma} \tag{4.61}$$

then Eq. (4.60) reduces to

$$\varphi_{\lambda\lambda} = \varphi_{\sigma\sigma} + \frac{1}{\sigma}\varphi_\sigma \tag{4.62}$$

This is a single linear partial differential equation. The boundary condition at the shoreline for Eq. (4.62) is

$$\sigma = 0 \tag{4.63}$$

which corresponds to the condition $c = 0$, i.e., the total water depth must be identically zero at the shoreline for all time.

In terms of the variables σ, λ and the potential $\varphi(\sigma, \lambda)$, Carrier and Greenspan (1958) proposed the following expressions for t, x, η, u and c

$$t = \frac{\lambda}{2} + u = \frac{\lambda}{2} + \frac{\varphi_\sigma}{\sigma} \tag{4.64}$$

$$x = \frac{1}{2}u^2 + c^2 + \frac{1}{4}\varphi_\lambda = \frac{1}{2}(\frac{\varphi_\sigma}{\sigma})^2 + \frac{1}{4}\varphi_\lambda + \frac{\sigma^2}{16} \tag{4.65}$$

$$\eta = c^2 - x = \frac{\sigma^2}{16} - x = -\frac{1}{4}\varphi_\lambda - \frac{\sigma^2}{16} \tag{4.66}$$

$$u = \frac{\varphi_\sigma}{\sigma} \tag{4.67}$$

$$c = \frac{\sigma}{4} \tag{4.68}$$

Although Eq. (4.62) is certainly much simpler to solve than the two original coupled nonlinear equations (4.39) and (4.40), it is difficult to obtain η or u as explicit functions of x and t. If $\varphi(\sigma,\lambda)$ is given, then Eqs. (4.64)−(4.68) will give t, x, η and u all parametrically in terms of the variables σ and λ. In general, it is very difficult to eliminate σ and λ to obtain direct functional relationships for η and u in terms of x and t.

4.5 Comparison of Theoretical Solution with Numerical Solution

A simple solution of Eq. (4.62) pointed out by Carrier and Greenspan (1958) is

$$\varphi(\sigma,\lambda) = -8A_0 J_0(\frac{\sigma}{2}) \sin\frac{\lambda}{2} \tag{4.69}$$

where A_0 is an arbitrary amplitude parameter and J_0 is the Bessel function of the first kind of order zero. This potential corresponds to a standing wave solution resulting from the perfect reflection from the shore of a wave of unit frequency. With $\varphi(\sigma,\lambda)$ given by Eq. (4.69), Eqs. (4.64) to (4.68) will implicitly give the solution of this standing wave. To evaluate $\eta(x,t)$ and $u(x,t)$ for arbitrary x and t, Eqs. (4.64) to (4.68) must be solved numerically. For specific values of x and t, σ and λ can be obtained from Eqs. (4.64) and (4.65) by using Newton's Method, so that $\eta(x,t)$ and $u(x,t)$ can easily be obtained from Eqs. (4.66) and (4.67), respectively.

To test the ability of the finite-difference model to simulate the moving boundary problem, a numerical simulation of a long wave will be performed in a rectangular basin with linearly varying water depth. All quantities used in the finite-difference

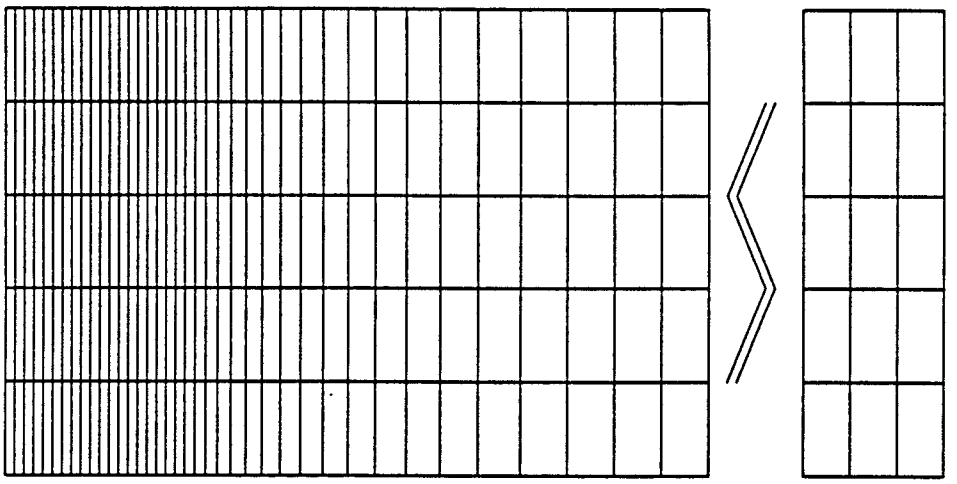


Figure 4.2: Computational grid

model are dimensional since the model is developed based on the dimensional governing equations, but the final solution obtained by the numerical model will be converted to dimensionless form in order to be compared with the theoretical solution. In the rest of this section, all dimensional quantities will be presented with units and dimensionless quantities will be presented without units.

The length of the rectangular basin is 60 km as measured from the mean water level and the width of the basin is 10 km. The slope of the bottom, ϕ, is 1 : 2500. The still water depth at the offshore boundary is 24 meters. The period of the long wave is set equal to 1 hour. Thus the frequency, ω^*, is $0.00174 sec.^{-1}$. We may now define the characteristic horizontal length scale by

$$L = \frac{\phi g}{(\omega^*)^2} = 129500 \ \ cm \tag{4.70}$$

Therefore we have the velocity scale

$$u_0 = \sqrt{\phi g L} = 224 \ \ cm/sec. \tag{4.71}$$

and the time scale

$$T = \sqrt{L/\phi g} = sec. \tag{4.72}$$

The computational domain is shown in the Fig. 4.2. Here we see that the grid density in the x-direction in the vicinity of the moving boundary is higher than in the offshore region. From the grid line of 50 km to the offshore boundary, the grid space increment is held at 1000 meters. Starting at this grid line, the grid space increment decreases 10 percent successively for about 6 km until the grid space increment of 200 meters is obtained. In the region of the moving boundary, the grid space increment is fixed at 200 meters. The grid density in the y- direction is unchanged, and the space increment is 2000 meters.

The moving boundary is simulated using the way of assuming a thin water layer to cover the dry region all the time. The thickness of this water layer is set to be 5 cm. To decrease the violation of mass conservation resulting from this layer, the friction with the Manning coefficient of 0.03 is considered in the region of thin water layer. However, there is no friction considered in the main computational domain since we did not consider the effect of friction in the derivation of the theoretical solution.

A time step of $\Delta t = 30$ seconds is chosen. At $t = 0$, the fluid is quiescent. For $t > 0$, the wave amplitude at the offshore boundary, η^*, can be given by

$$\eta^*(t) = \eta(t)\,\phi L \qquad cm \tag{4.73}$$

where $\eta(t)$ is the dimensionless value of the wave amplitude which can be obtained from the theoretical solution. After about three periods, we can obtain the state numerical solution of the standing wave which will be compared with the theoretical solution.

Figures 4.3 and 4.4 shows the comparisons between numerical and theoretical wave profiles over the entire length of the basin at 7 time instant during half of a period. Fig. 4.5 presents the amplification of these comparisons near the moving boundary region. From Figs. 4.3, 4.4 and 4.5, it is seen that the agreement between the numerical solution and the theoretical solution is good.

Figure 4.6 presents the comparisons of water surface displacement near the moving boundary and the offshore boundary. Figure 4.7 presents the comparisons of velocities at the same points as in Fig. 4.6. Both Figs. 4.6 and 4.7 show that the agreement between the numerical solution and the theoretical solution is good.

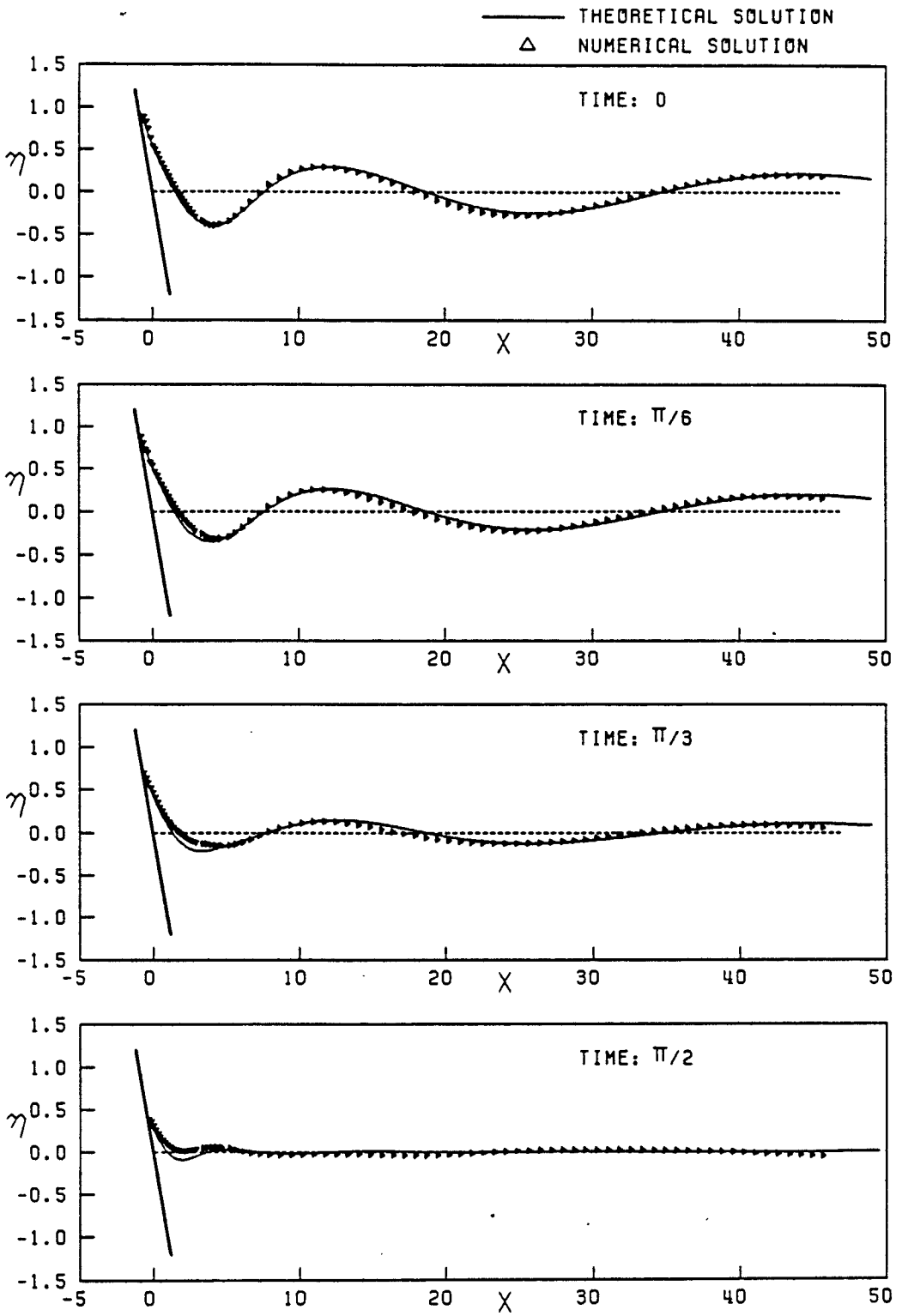


Figure 4.3: Comparison between wave profiles as predicted by theory and the numerical model ($\eta = \eta^* \omega^2/\phi^2 g$, $x = x^* \omega^2/\phi g$)

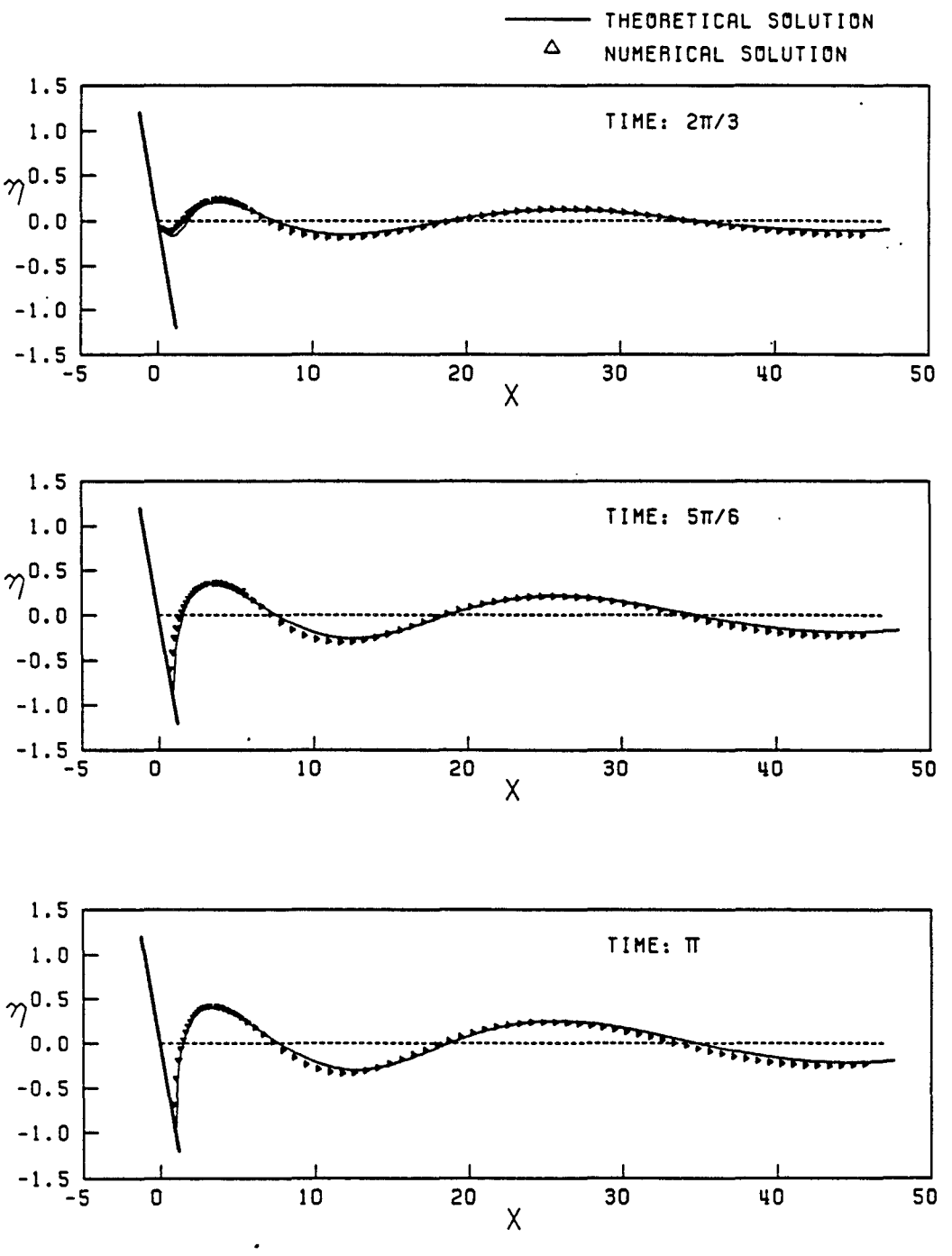


Figure 4.4: Comparison between wave profiles as predicted by theory and the numerical model ($\eta = \eta^* \omega^2/\phi^2 g$, $x = x^* \omega^2/\phi g$)

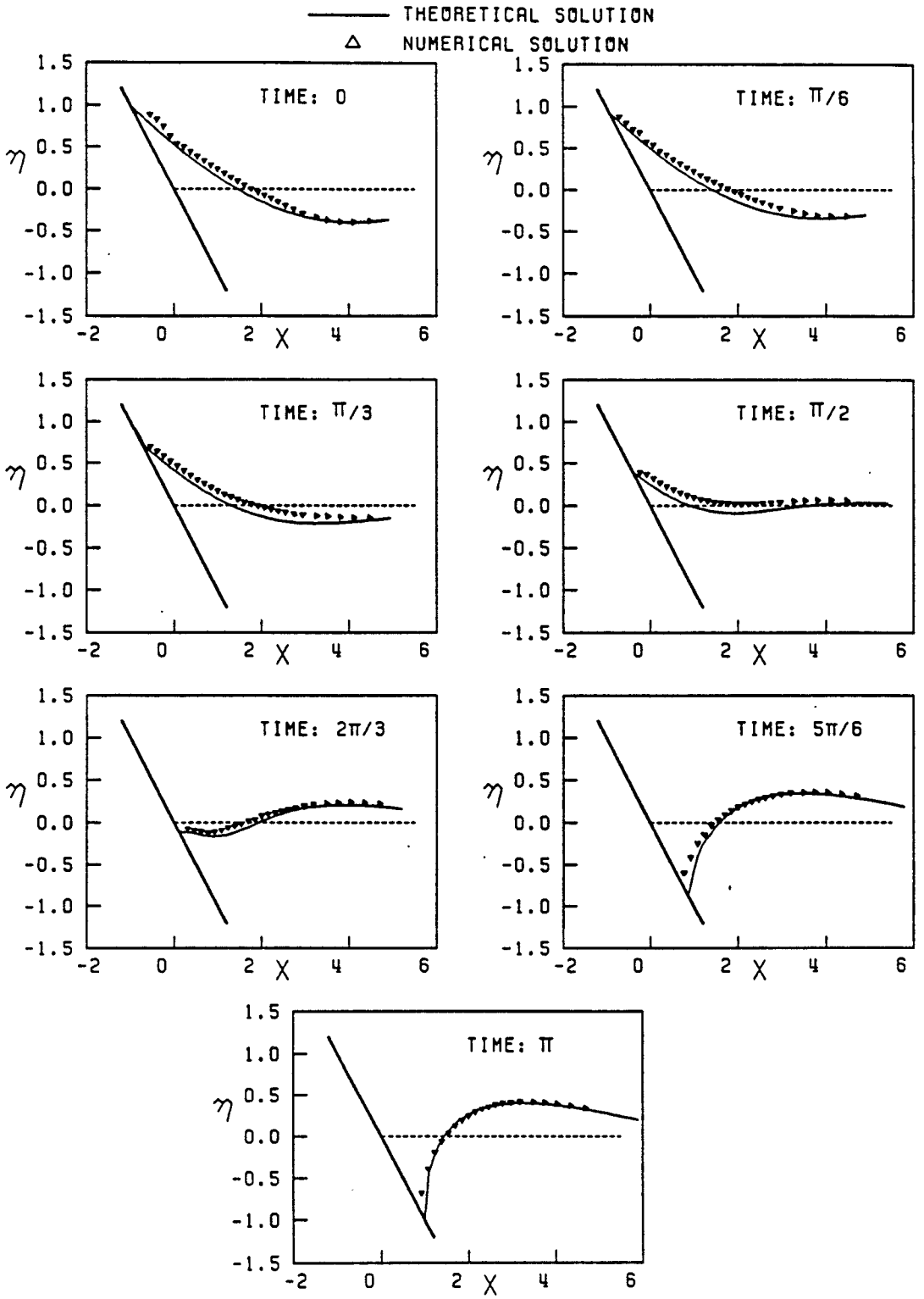


Figure 4.5: Comparison between wave profiles near moving boundary as predicted by theory and the numerical model ($\eta = \eta^* \omega^2/\phi^2 g$, $x = x^* \omega^2/\phi g$)

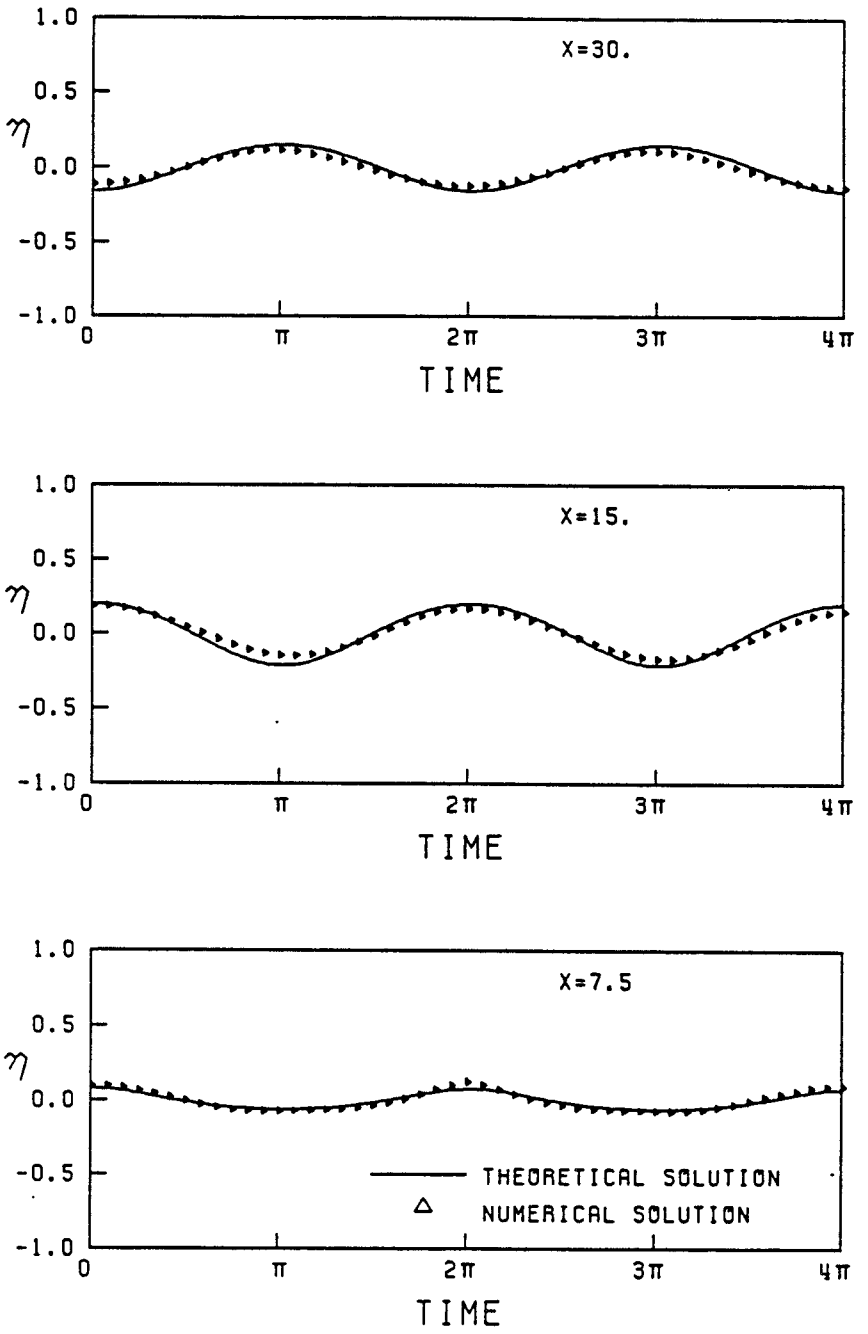


Figure 4.6: Comparison between theoretical and numerical solutions of water elevation ($\eta = \eta^* \omega^2 / \phi^2 g, t = \omega t^*$)

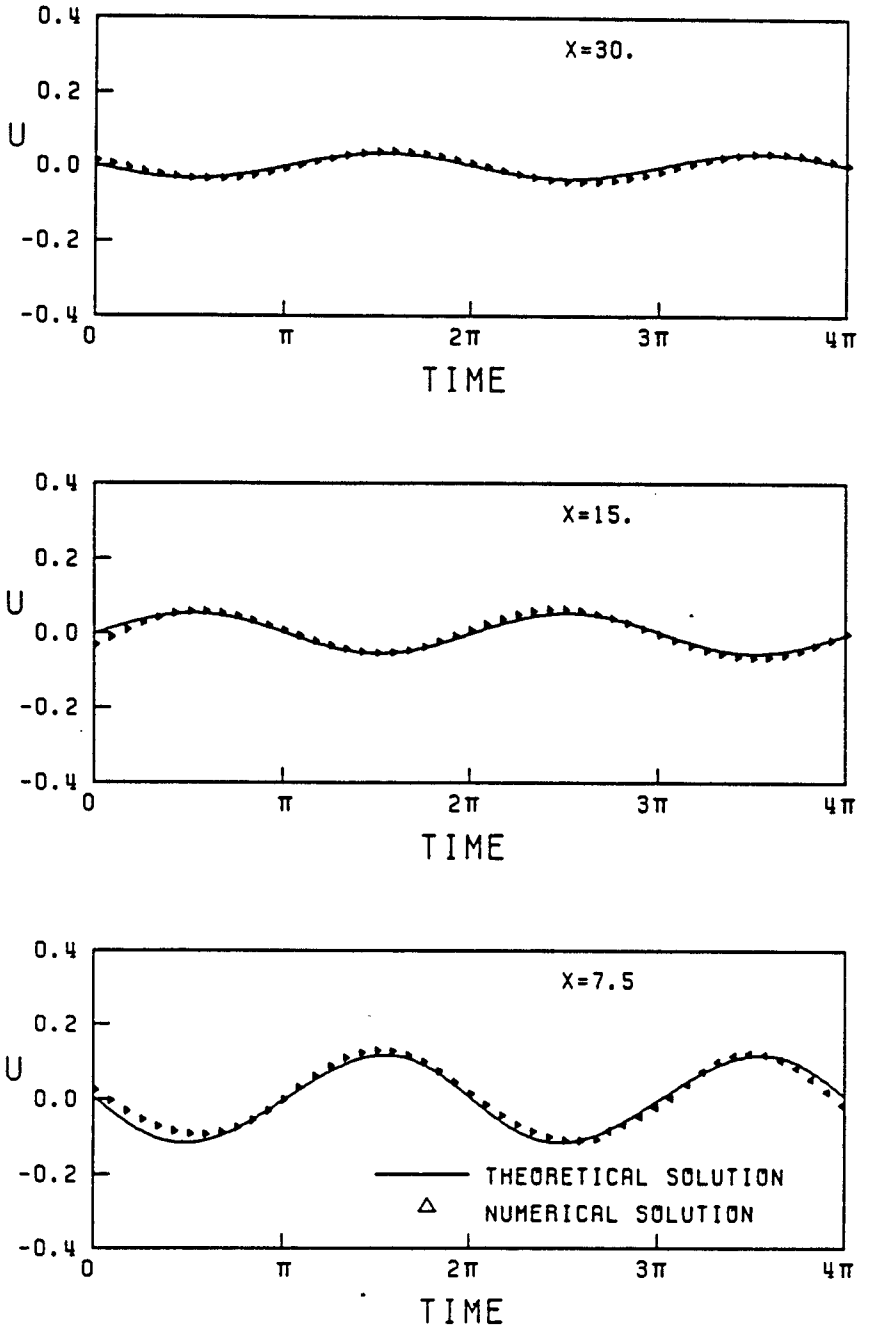


Figure 4.7: Comparison between theoretical and numerical solutions of velocity ($u = u^*\omega/\phi g, t = \omega t^*$)

CHAPTER 5
APPLICATION TO LAKE OKEECHOBEE

In this chapter, the moving boundary numerical model developed in the previous chapter will be applied to simulate wind driven circulation in Lake Okeechobee, Florida.

As shown in Fig. 5.1, the western part of Lake Okeechobee is the grass region where the water depth is about 30 to 100 centimeters. During a seiche induced by a hurricane, this region may be inundated or dried due to the temporal variation of water level. The water depth outside the grass area is about 4 meters on the average.

A 31 × 50 grid is constructed as shown in Fig. 5.2. The north to south grid spacing is uniform, but the east to west grid spacing is variable with smaller spacing in the grass area. The north to south grid spacing is 1120 meters. The maximum east to west grid spacing is 2240 meters and the minimum is 1120 meters.

Water depth values at the grid points are obtained by adding 1.2 meters to the numbers given in the 1987 contour map of Lake Okeechobee which is for low water conditions.

The wind shear stress acting on the lake surface can be calculated by the following formulae

$$\tau_x = \rho_a C_{da} \sqrt{W_x^2 + W_y^2}\, W_x \tag{5.1}$$

$$\tau_y = \rho_a C_{da} \sqrt{W_x^2 + W_y^2}\, W_y \tag{5.2}$$

where τ_x and τ_y denote respectively the wind shear stresses in the x- and y-directions, ρ_a is the density of air, W_x and W_y are the wind speeds in the x- and y-directions, respectively, and C_{da} is the wind shear stress coefficient. The value of C_{da} can be

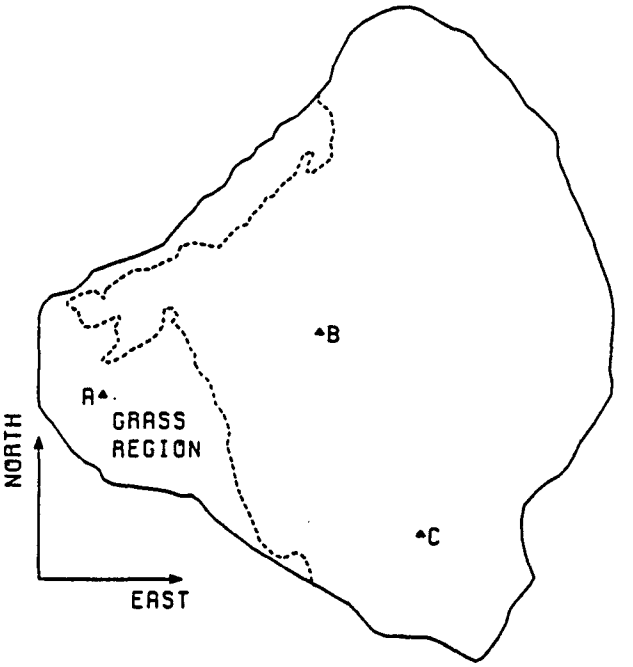


Figure 5.1: The sketch of Lake Okeechobee

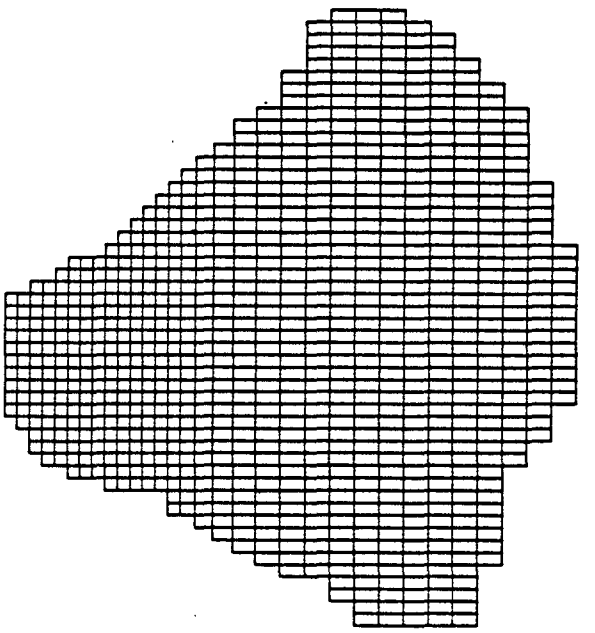


Figure 5.2: Computational grid

obtained from the following formulation proposed by Garrett (1977)

$$C_{da} = 0.001 \times (0.75 + 0.00067 \times \sqrt{W_x^2 + W_y^2}) \tag{5.3}$$

where the unit of W_x and W_y is $cm/sec.$. The maximum value of C_{da} is 0.003. The bottom friction can be calculated from the Manning-Chezy friction formulation.

As discussed previously, the computer programming becomes very complicated when using the weir formulation to simulate the two-dimensional moving boundary problem. Thus the method proposed by Reid and Bodine (1968) is not applied to study the wind driven circulation of Lake Okeechobee. The method to simulate the moving boundary problem with a thin water layer on the dry land area is employed here. To illustrate the significance of the moving boundary to the actual problem, the simulations will be conducted with and without the moving boundary. For the moving boundary case, the grass area is included in the computational domain and the thickness of the thin water layer on the dry area is assumed to be 10 cm. For the fixed boundary case, the grass area is taken out of the computational domain and a vertical wall is assumed to be located at the edge of the grass region. Thus the boundary between the grass area and open water can be considered as a closed boundary.

A numerical time step of 3 minutes is used here. The Coriolis parameter, Ω, in Lake Okeechobee is $0.73 \times 10^{-4}\, sec.^{-1}$ (Schmalz,1986).

A spatially uniform wind from the east to the west is applied over the lake surface for 9 hours. The wind speed is 20 m/sec.. After the wind stops, a seiche in the lake will be produced. When the wind is applied over the lake, the Manning coefficient is chosen to be 0.02 (Schmalz, 1986). But after the wind stops, the Manning coefficient is taken as 0.005 in order that the seiche can last for a longer time for studying the shoreline motion.

The steady state of wind-driven circulation in the lake is reached after the wind is applied for 7 hours. Figures 5.3 and 5.4 show the circulations for the moving

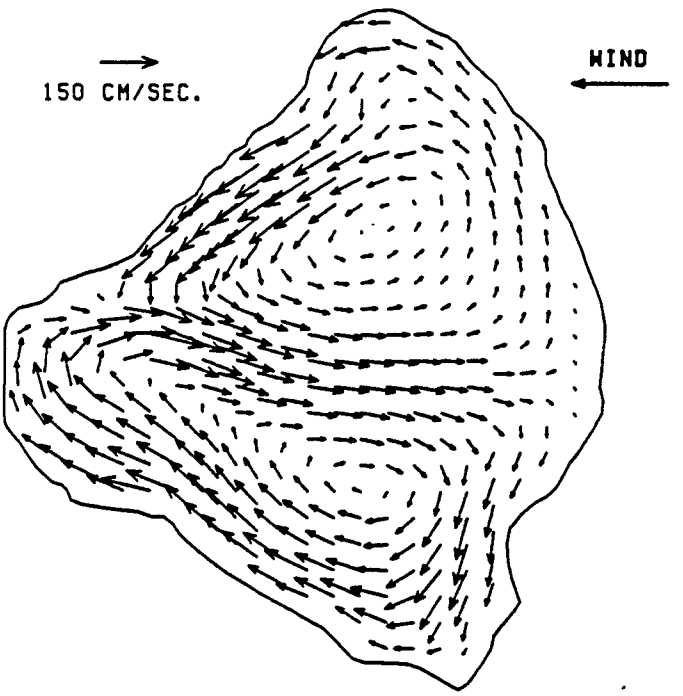


Figure 5.3: Wind driven circulation with moving boundary

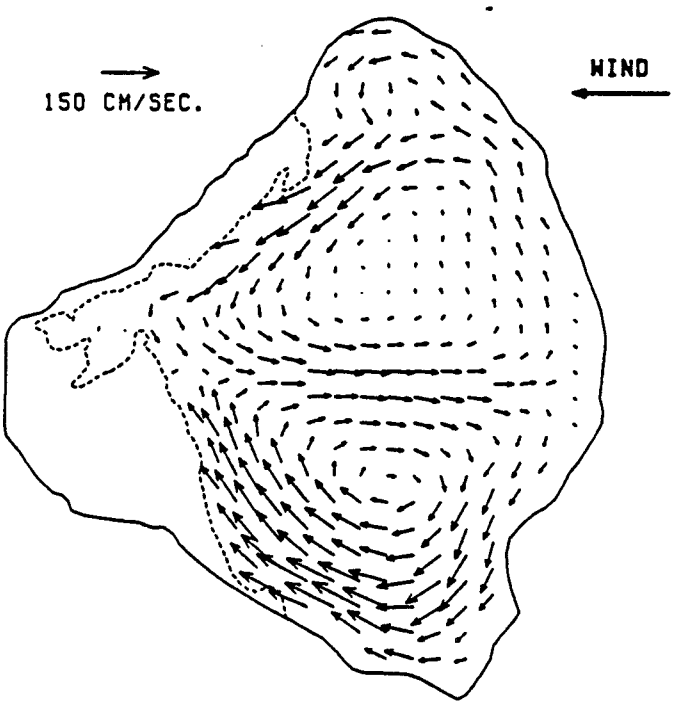


Figure 5.4: Wind driven circulation with fixed boundary

boundary case and the fixed boundary case respectively after the wind has been applied for 8 hours. It is seen that the circulation for the two cases are quite similar, although the velocities in the moving boundary case appear to be larger than those in the fixed boundary case. This can be explained by the fact that the area of the lake in the moving boundary case is greater than that in the fixed boundary case.

Figure 5.5 shows the variation of the wind speed with time. Figure 5.6 shows the motion of the western shoreline during the first cycle of seiche oscillation in the lake after the wind has stopped.

As mentioned previously, mass conservation is violated by assuming that a thin water layer exists on the dry area at all times. Figure 5.7 shows the extent of the mass conservation violation during the entire computational period. It is found that an extra mass of 0.2 percent of the total water mass in the lake is induced due to the consideration of the shoreline motion. Figure 5.8 presents the variation of the total dry area in the lake with the time.

Figure 5.9 shows the comparisons of the transient variations of the water surface elevation at points A, B and C between the moving boundary case and the fixed boundary case. The locations of points A, B and C in the lake are shown in the Fig. 5.1. Figures 5.10 and 5.11 present the comparisons for the velocities U and V, respectively, between the moving boundary case and the fixed boundary case. From Figs. 5.9, 5.10 and 5.11, it is seen that there is a phase lag between the results for the moving boundary case and the fixed boundary case.

Figures 5.12 and 5.13 present the transient variation of the bottom friction in the x- and y-directions at point A, respectively.

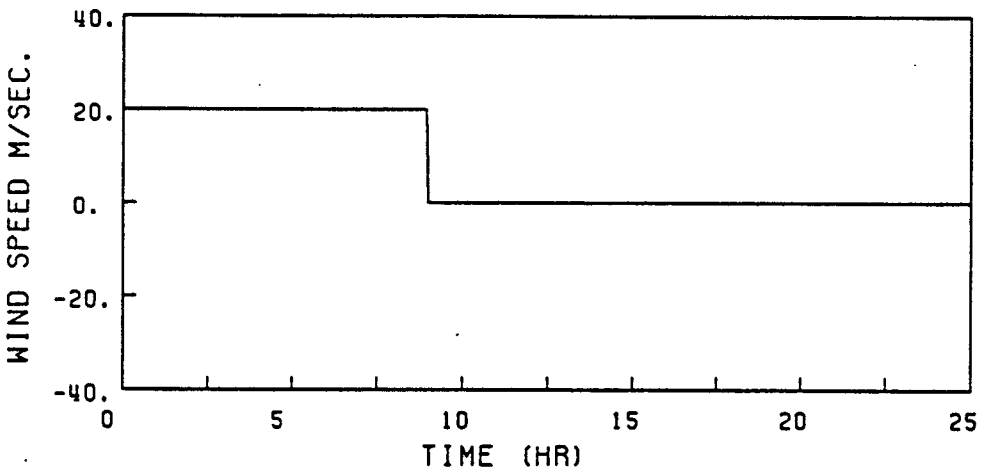


Figure 5.5: Variation of wind speed with time

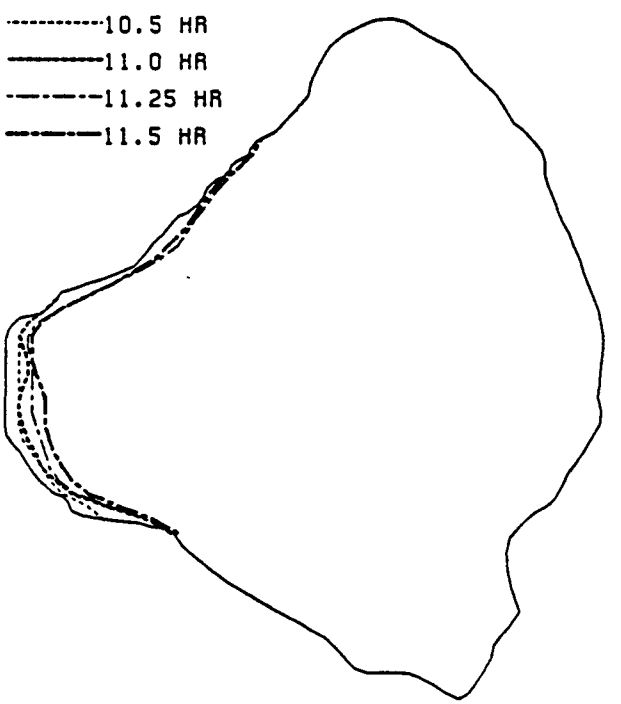


Figure 5.6: Locations of the moving boundary at four different time

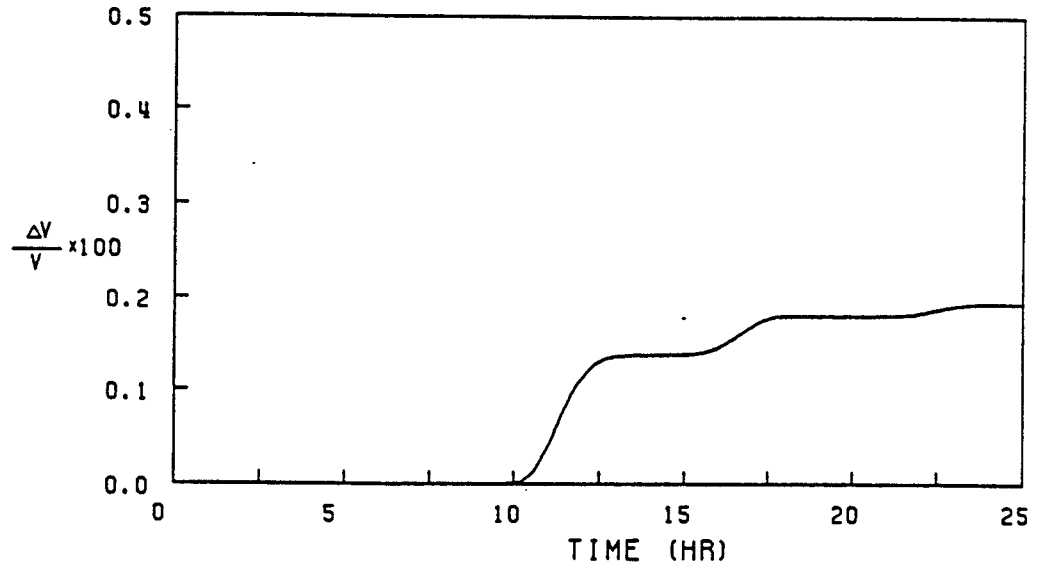


Figure 5.7: Extra mass introduced by considering the moving boundary

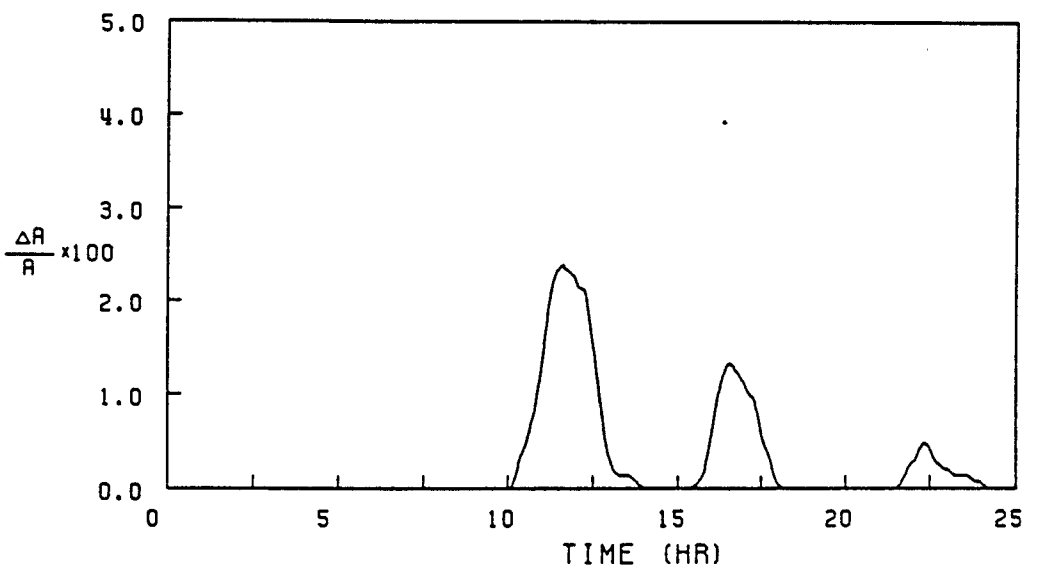


Figure 5.8: Transient variation of dry area

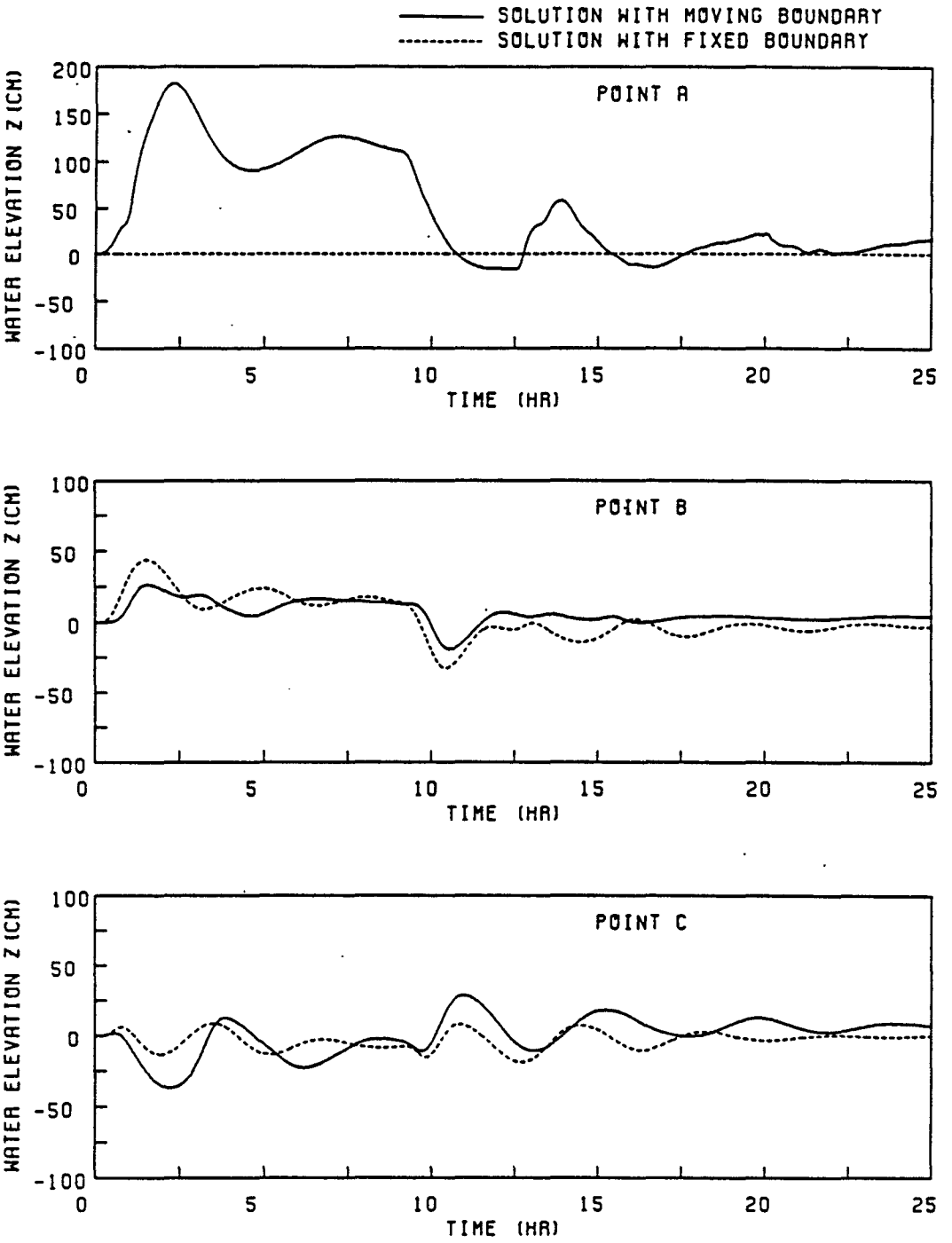


Figure 5.9: Comparisons of water surface elevation with moving boundary and fixed boundary

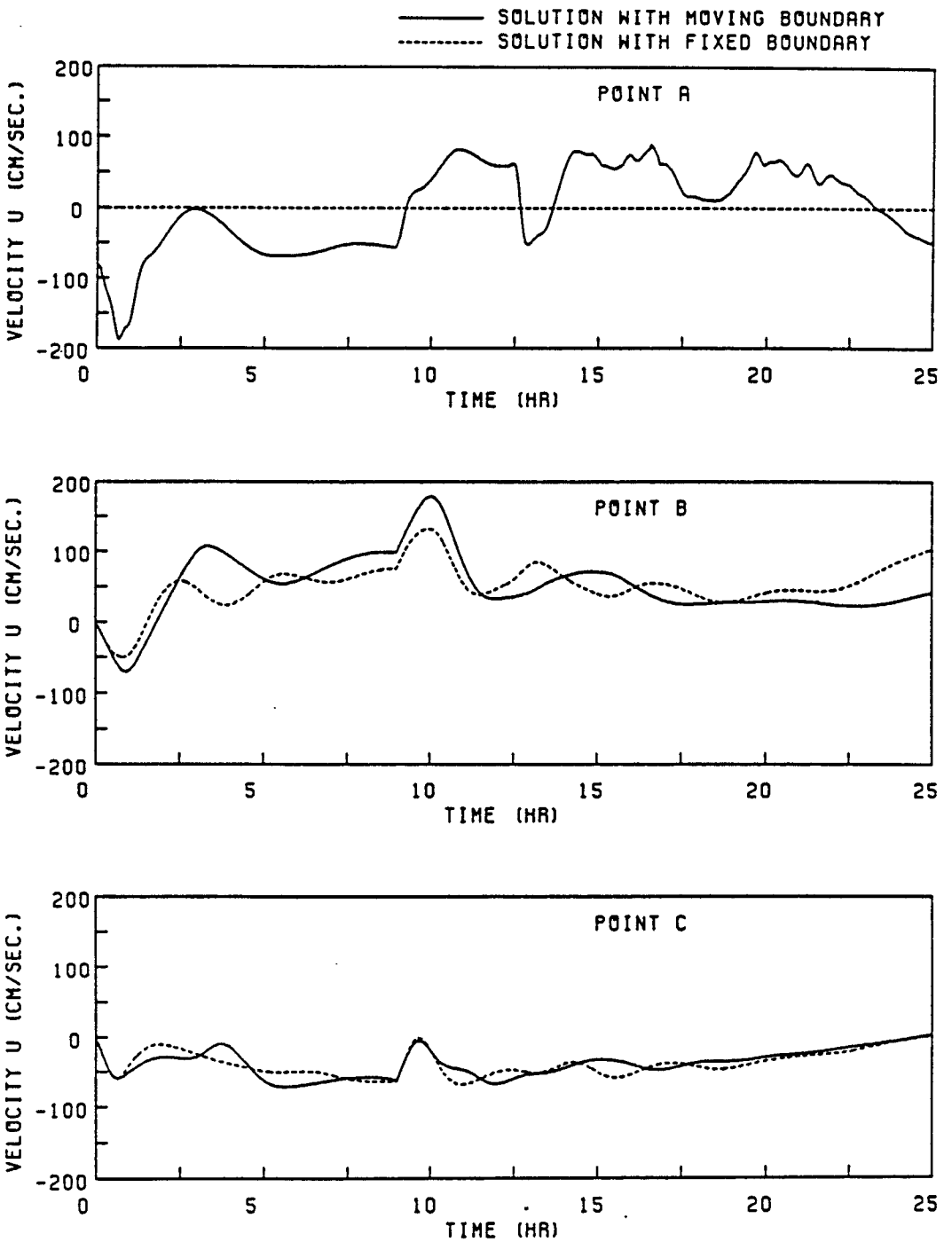


Figure 5.10: Comparisons of velocity U with moving boundary and fixed boundary

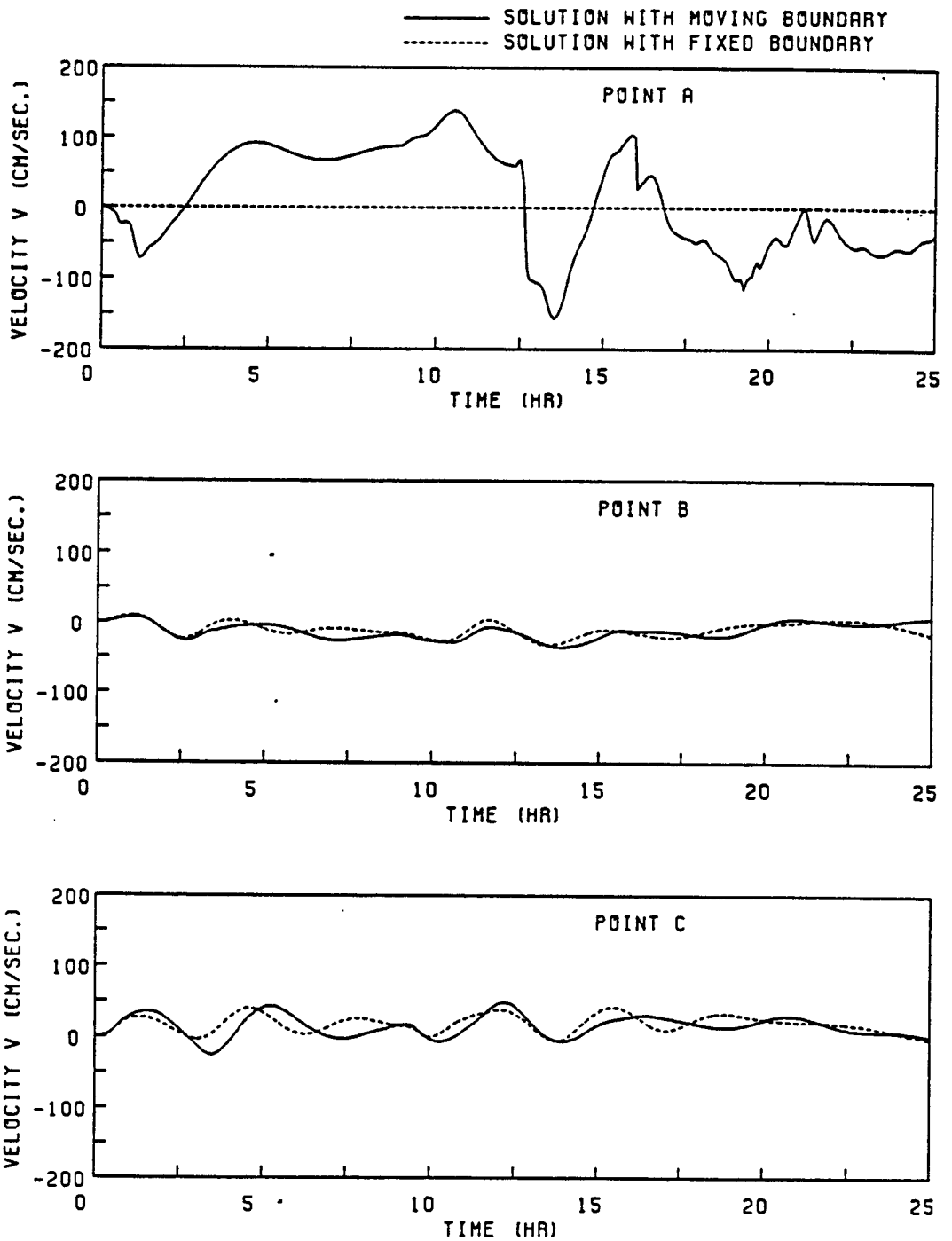


Figure 5.11: Comparisons of velocity V with moving boundary and fixed boundary

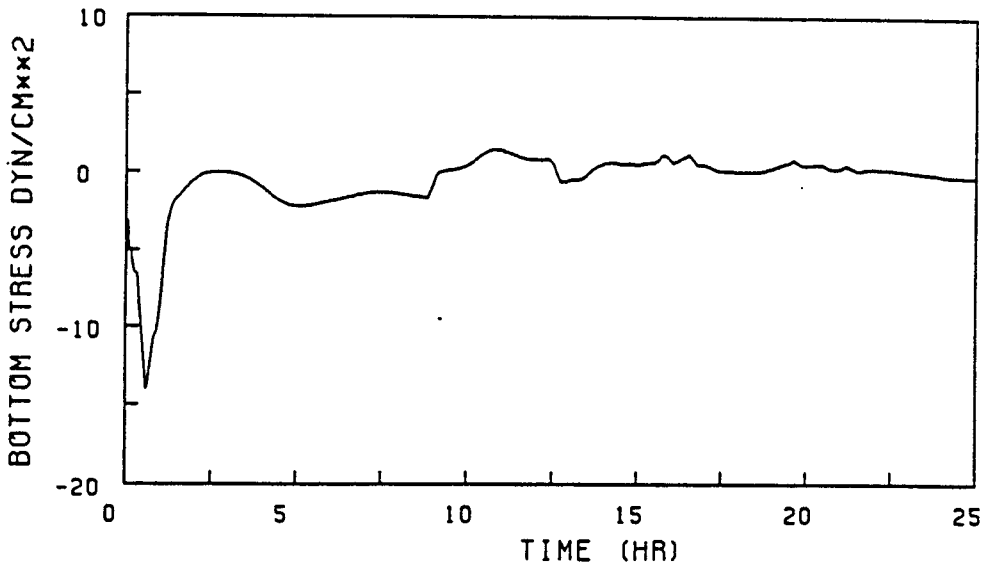


Figure 5.12: Transient variation of bottom friction in the x-direction at point A

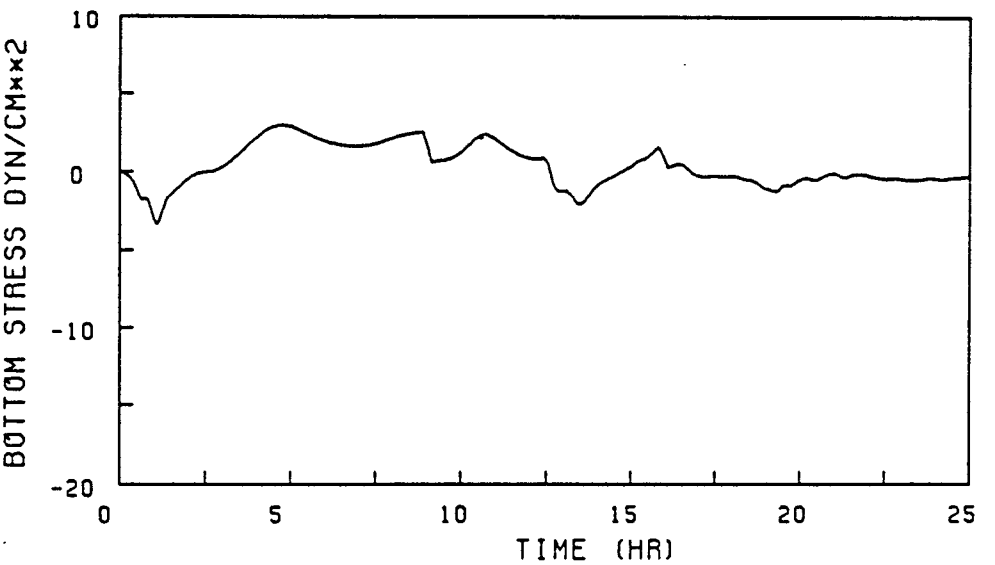


Figure 5.13: Transient variation of bottom friction in the y-direction at point A

CHAPTER 6
CONCLUSIONS

6.1 Conclusions

The objective of this study is to develop a two-dimensional moving boundary numerical model which can predict the hydrodynamics in lakes, estuaries and shallow seas with the effect of a moving boundary . The study consists of derivation, verification, and application of the model.

The finite difference technique is used to develop the model in terms of a non-uniform rectangular grid. The governing equations, which are vertically-integrated Navier-Stokes equations of water motion, are solved using the method of fractional steps. The transition from one stage of the computation to the next is divided into three steps which are advection, diffusion, and propagation. Different numerical schemes are employed for each computational step. The method of characteristics is used for the advection, the ADI method is applied to the diffusion, and the conjugate gradient iterative method is used for the propagation. The numerical schemes used in the model are implicit so that there is no limitation for the choice of the numerical time step.

Two methods for simulating the moving boundary problem are discussed in this study. The first, which was proposed by Reid and Bodine (1968), is examined briefly. It is found that it is difficult to determine the values of empirical coefficients in the weir formulation since they are very site-dependent. It is also found that the computer program for simulating the motion of the boundary is very complicated for two-dimensional problems. The second, which was proposed by Benque et al. (1982), is studied in detail. The advantage of this method is that the computer

program for simulating two-dimensional moving boundary problems is very simple. The disadvantage is that the mass conservation is violated slightly.

For the verification of the model, four analytical solutions are used to compare with the numerical solutions. From these comparisons, it can be concluded that the consistency and the accuracy of the model are acceptable. It is also found that the method of fractional steps is a powerful means of solving the complicated problems in several variables although its consistency has not been completely proved. In order to validate the model's ability to simulate the motion of the boundary, wave propagation on a linearly sloping beach is studied theoretically and numerically. It is found that the moving boundary model developed using this method can simulate moving boundary problems reasonably well although the mass conservation is violated slightly.

The model is applied to the wind-driven circulation in Lake Okeechobee, Florida. Comparison is made between the results obtained from the moving boundary model and the fixed boundary model. From this comparison, it is seen that the boundary motion can not be neglected when studying wind-driven circulation in Lake Okeechobee. It is found that the violation of mass conservation in the moving boundary model can be negligible.

6.2 Future Study

In the moving boundary region, the water depth is usually very small and the water motion is controlled by the bottom friction. In this case, the Manning-Chezy formulation cannot give a correct estimation of the bottom friction since it breaks down when the water depth approaches zero. Therefore, basic hydrodynamic research is needed in this area. For example, Yih (1963) investigated the stability of liquid flow down an inclined plane and Melkonian (1987) studied nonlinear waves in thin films.

When using a uniform or non-uniform rectangular grid to approximate the com-

plex geometry of water bodies, a large number of grid points is generally required. As a result, the computational cost is increased greatly. To avoid this, it is useful to modify this model to work in a curvilinear grid.

In addition, we can use the method of fractional steps to develop a three-dimensional numerical model.

APPENDIX A
APPLICATION OF THE CONJUGATE GRADIENT METHOD TO THE PROPAGATION STEP

In this appendix, the algorithm of the conjugate gradient method for the solution of simultaneous linear algebraic equations will be presented briefly and the procedure of the application of this method to the propagation step will be described in detail.

A.1 Conjugate Direction Method

In order to understand the algorithm of the conjugate gradient method, the conjugate direction method needs to be reviewed briefly since the conjugate gradient method is a special case of the conjugate direction method. It is assumed that there is a solution vector h existed to a system of simultaneouse linear algebraic equations,

$$A\,x = k \tag{A.1}$$

in which A is an $N \times N$ matrice of coefficients, x is an $N \times 1$ vector of unknowns, and k is an $N \times 1$ vector of constants. In the following description, the symbol (p, q) presents the inner product of the vectors p and q, or the value of $p^T q$.

Let us suppose that a set of N "A-conjugate" or "A-orthogonal" vectors $\{p_i\}$, $i = 0, 1, 2, \cdots, N-1$, is available to us. This means that the inner product $(Ap_i, p_j) = 0$ where $i \neq j$. If A is positive definite, then $(Ap_i, p_i) > 0$. In this case, since the vectors $\{p_i\}$ are necessarily linearly independent and span the N-dimensional space, the solution vector h can be written as

$$h = c_0 p_0 + c_1 p_1 + c_2 p_2 + \cdots + c_{N-1} p_{N-1} \tag{A.2}$$

where $\{c_i\}$ are coefficients. If we can determine $\{c_i\}$, the solution h can be quickly calculated. Since

$$(k, p_i) = (Ah, p_i) \tag{A.3}$$

$$\begin{aligned}(Ah, p_i) = c_0(Ap_0, p_i) &+ c_1(Ap_1, p_1) + \cdots + c_i(Ap_i, p - i) + \cdots \\ &+ c_{N-1}(Ap_{N-1}, p_i) = c_i(Ap_i, p_i)\end{aligned} \tag{A.4}$$

we get

$$c_i = \frac{(k, p_i)}{(Ap_i, p_i)} \tag{A.5}$$

Consequently

$$h = \frac{(k, p_0)}{(Ap_0, p_0)}p_0 + \frac{(k, p_1)}{(Ap_1, p_1)}p_1 + \cdots + \frac{(k, p_{N-1})}{(Ap_{N-1}, p_{N-1})}p_{N-1} \tag{A.6}$$

This computation of h defined by equation (A.6) also can be described as the following iterative scheme

$$x_1 = \beta_0 p_0 \tag{A.7}$$

$$x_{i+1} = x_i + \beta_i p_i \tag{A.8}$$

where

$$\beta_i = \frac{(k, p_i)}{(Ap_i, p_i)} \tag{A.9}$$

and $p_0, p_1, \cdots, p_{N-1}$ is the given set of N $A-$conjugate vectors. The iteration can be terminated when $x_M = h$ where $M \leq N$.

A.2 Conjugate Gradient Method

In conjugate gradient method, a particular set of $A-$conjugate vectors, $\{p_i\}$, is developed and a solution to equation $(A.1)$ formed in terms of these. Introducing residual vectors,

$$r_{i+1} = k - A\,x_{i+1} \tag{A.10}$$

Beckman (1958) used Gram-Schmidt Orthogonalization procedure to obtain the $A-$conjugate direction vectors $\{p_i\}$. They are expressed as

$$p_{i+1} = r_{i+1} + \beta_i p_i \tag{A.11}$$

in which

$$\beta_i = \frac{| r_{i+1} |^2}{| r_i |^2} \tag{A.12}$$

where $| r_{i+1} |$ and $| r_i |$ represent the length of the vectors r_{i+1} and r_i. The fundamental conjugate gradient iterative procedure leading to a solution of equation $(A.1)$ can be defined by following formulae:

$$p_0 = r_0 = k - A x_0 \tag{A.13}$$

$$\alpha_i = \frac{(p_i, r_i)}{(p_i, Ap_i)} \tag{A.14}$$

$$x_{i+1} = x_i + \alpha_i p_i \tag{A.15}$$

$$r_{i+1} = k - A x_{i+1} \tag{A.16}$$

$$\beta_i = \frac{| r_{i+1} |^2}{| r_i |^2} \tag{A.17}$$

$$p_{i+1} = r_{i+1} + \beta_i p_i \tag{A.18}$$

After M iterations, with $M \leq N$, x_M will be equal to the solution h if all conputations are done with no loss of accuracy.

A.3 Application to the Propagation Step

The key to the prapagation step presented in Chapter 2 is to solve for $\Delta Z_1(x,y)$, $\Delta Z_2(x,y)$ and $q(x,y)$ which satisfy

$$\frac{\Delta Z_1}{2g(\Delta t)^2} - \alpha^2\{\frac{\partial(h^n \frac{\partial \Delta Z_1}{\partial x})}{\partial x} + \frac{\partial(\Delta Z_1 \frac{\partial Z^n}{\partial x})}{\partial x}\} + \frac{\alpha}{g\Delta t}\frac{\partial(\frac{U^{n+2/3}}{h^n}\Delta Z_1)}{\partial x} = f_1 - q \tag{A.19}$$

$$\frac{\Delta Z_2}{2g(\Delta t)^2} - \alpha^2\{\frac{\partial(h^n\frac{\partial\Delta Z_2}{\partial 2})}{\partial y} + \frac{\partial(\Delta Z_2\frac{\partial Z^n}{\partial y})}{\partial y}\} + \frac{\alpha}{g\Delta t}\frac{\partial(\frac{V^{n+2/3}}{h^n}\Delta Z_2)}{\partial y} = f_2 + q \quad \text{(A.20)}$$

$$\Delta Z_1(x,y) = \Delta Z_2(x,y) \quad \text{(A.21)}$$

and subject to the open boundary and fixed boundary or moving boundary conditions. In order to present the way to solve above equations clearly, we need to write the above equations in the matrix form as

$$[A_x][\Delta Z_1] = [f_1] - [B_x][q] \quad \text{(A.22)}$$

$$[A_y][\Delta Z_2] = [f_2] - [B_y][q] \quad \text{(A.23)}$$

$$[B_y][\Delta Z_1] + [B_x][\Delta Z_2] = 0 \quad \text{(A.24)}$$

in which $[A_x]$ and $[A_y]$ are symetrical coefficient matrix of Eqs. (A.19) and (A.20), respectively, $[\Delta Z_1]$ and $[\Delta Z_2]$ are vectors of unknowns, $[f_1]$ and $[f_2]$ are vectors of known terms at the right sides of Eqs. (A.19) and (A.20), respectively, $[q]$ is the vector of coordinate terms, and $[B_x]$ is identity matrice.

It should be noted that the arrangement of elements in $[\Delta Z_1]$ is different from that in $[\Delta Z_2]$. Equation (A.21) is for the same grid point. Therefore, a matrice needs to be defined to make the same arrangement of elements in $[\Delta Z_1]$ and $[\Delta Z_2]$. $[B_y]$ is defined to do it. Actually for large grid system, it is difficult to obtain the explicit form of $[B_y]$. Fortunately it will be seen in later description that we do not really need to know $[B_y]$.

Eqs. (A.22), (A.23) and (A.24) can be rewritten as

$$\begin{bmatrix} A_x & 0 \\ 0 & A_y \end{bmatrix}\begin{bmatrix} \Delta Z_1 \\ \Delta Z_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} - \begin{bmatrix} B_x \\ B_y \end{bmatrix}[q] \quad \text{(A.25)}$$

$$\begin{bmatrix} B_x \\ B_y \end{bmatrix}^T\begin{bmatrix} \Delta Z_1 \\ \Delta Z_2 \end{bmatrix} = 0 \quad \text{(A.26)}$$

Denoting $\begin{bmatrix} A_x & 0 \\ 0 & A_y \end{bmatrix} = A$, $\begin{bmatrix} \Delta Z_1 \\ \Delta Z_2 \end{bmatrix} = \Delta Z$, $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = f$, $\begin{bmatrix} B_x \\ B_y \end{bmatrix} = B$, and $[q] = q$, Eqs. (A.25) and (A.26) become

$$A\,\Delta Z = f - B\,q \tag{A.27}$$

$$B^T\,\Delta Z = 0 \tag{A.28}$$

where B^T is transposed matrice of B. From Eq. (A.27), we can get

$$\Delta Z = A^{-1}\,(f - B\,q) \tag{A.29}$$

Substitution of ΔZ into Eq. (A.28) yields

$$(B^T\,A^{-1}\,B)\,q = B^T\,A^{-1}\,f \tag{A.30}$$

where $(B^T A^{-1} B)$ is a symetric matrice.

The conjugate gradient method is used to solve Eq. (A.30) for q. The iteration on q consists of looking for q^{m+1} by given q^m where m means the mth iteration. The residual vector is

$$\begin{aligned} r^{m+1} &= B^T\,A^{-1}\,f - (B^T\,A^{-1}\,B)\,q^{m+1} \\ &= B^T\,(A^{-1}\,f - A^{-1}\,B\,q^{m+1}) \\ &= B^T\,\Delta Z^{m+1} \\ &= \Delta Z_1^{m+1} - \Delta Z_2^{m+1} \end{aligned} \tag{A.31}$$

Coefficient β_m can be calculated by

$$\begin{aligned} \beta_m &= \frac{|\,r^{m+1}\,|^2}{|\,r^m\,|^2} \\ &= \sum_{i,j}^{I,J}[\Delta Z_{1(i,j)}^{m+1} - \Delta Z_{2(i,j)}^{m+1}]^2 / \sum_{i,j}^{I,J}[\Delta Z_{1(i,j)}^{m} - \Delta Z_{2(i,j)}^{m}]^2 \end{aligned} \tag{A.32}$$

The direction vector is

$$p^{m+1} = r^{m+1} + \beta_m\,p^m \tag{A.33}$$

The coefficient α_m is

$$\alpha_m = \frac{(p^m, r^m)}{(p^m, B^T A^{-1} B\, p^m)}$$

$$= \frac{[p^m]^T [B^T \Delta Z^m]}{[p^m]^T [B^T A^{-1} B\, p^m]} \tag{A.34}$$

Defining $A^{-1}Bp^m = H^m$, therefore we have

$$A\, H^m = B\, p^m \tag{A.35}$$

Comparing Eq. (A.35) with Eq. (A.27), it is seen that H^m can be obtained by using the double-sweep method to solve

$$\frac{H_1}{2g(\Delta t)^2} - \alpha^2\{\frac{\partial(h^n \frac{\partial H_1}{\partial x})}{\partial x} + \frac{\partial(H_1 \frac{\partial Z^n}{\partial x})}{\partial x}\} + \frac{\alpha}{g\Delta t}\frac{\partial(\frac{U^{n+2/3}}{h^n} H_1)}{\partial x} = B_x\, p^m \tag{A.36}$$

$$\frac{H_2}{2g(\Delta t)^2} - \alpha^2\{\frac{\partial(h^n \frac{\partial H_2}{\partial 2})}{\partial y} + \frac{\partial(H_2 \frac{\partial Z^n}{\partial y})}{\partial y}\} + \frac{\alpha}{g\Delta t}\frac{\partial(\frac{V^{n+2/3}}{h^n} H_2)}{\partial y} = B_y\, p^m \tag{A.37}$$

where $\begin{bmatrix} H_1 \\ H_2 \end{bmatrix} = H$. Consequently

$$\alpha_m = \frac{[p^m]^T [B^T \Delta Z^m]}{[p^m]^T [B^T H^m]}$$

$$= \sum_{i,j}^{I,J} [\Delta Z^m_{1(i,j)} - \Delta Z^m_{2(i,j)}]\, p^m_{(i,j)} / \sum_{i,j}^{I,J} [H^m_{1(i,j)} - H^m_{2(i,j)}]\, p^m_{(i,j)} \tag{A.38}$$

The iteration procedure for q can be sumarized as:

(1) Let the final value of q at the previouse time step be the initial approximation to the solution of q at the new time step.

(2) Apply the double-sweep method to solve Eqs. (A.19) and (A.20) for ΔZ_1^0 and ΔZ_2^0.

(3) Calculate the residual vector by $r^0 = \Delta Z_1^0 - \Delta Z_2^0$.

(4) Let $p^0 = r^0$

(5) Use the double-sweep method to solve Eqs. (A.36) and (A.37) for H_1^0 and H_2^0.

(6) Compute the coefficient α_0 using Eq. (A.38).

(7) Advance q by $q^1 = q^0 + \alpha_0 p^0$

(8) Determine successively

$$r^{m+1} = \Delta Z_1^{m+1} - \Delta Z_2^{m+1} \tag{A.39}$$

$$\beta_m = \sum_{i,j}^{I,J} [\Delta Z_{1(i,j)}^{m+1} - \Delta Z_{2(i,j)}^{m+1}]^2 / \sum_{i,j}^{I,J} [\Delta Z_{1(i,j)}^{m} - \Delta Z_{2(i,j)}^{m}]^2 \tag{A.40}$$

$$p^{m+1} = r^{m+1} + \beta_m p^m \tag{A.41}$$

$$\alpha_{m+1} = \sum_{i,j}^{I,J} [\Delta Z_{1(i,j)}^{m+1} - \Delta Z_{2(i,j)}^{m+1}] p_{(i,j)}^{m+1} / \sum_{i,j}^{I,J} [H_{1(i,j)}^{m+1} - H_{2(i,j)}^{m+1}] p_{(i,j)}^{m+1} \tag{A.42}$$

$$q^{m+2} = q^{m+1} + \alpha^{m+1} p^{m+1} \tag{A.43}$$

(9) Repeat Step 8 with $m+1$ replacing m and continue until $m = N-1$ or until the residual vector becomes sufficiently small, whichever condition may be satisfied first.

(10) Let $\Delta Z = (\Delta Z_1 + \Delta Z_2)/2$

APPENDIX B
DERIVATION OF Z_2 AND U_2

The governing equations are

$$\frac{\partial Z_2}{\partial t} + \frac{\partial U_2}{\partial x} = 0 \tag{B.1}$$

$$\begin{aligned} \frac{\partial U_2}{\partial t} + gh\frac{\partial Z_2}{\partial x} &= \frac{a^2c^2k}{8h\cos^2 kl}\{\sin[2k(l-x)+2\omega t] \\ &+ \sin[2k(l-x)-2\omega t] + 2\sin 2k(l-x)\} \end{aligned} \tag{B.2}$$

and the boundary conditions are

$$U_2(l,t) = 0 \tag{B.3}$$

$$Z_2(0,t) = 0 \tag{B.4}$$

Eliminating U_2 from Eqs. (B.1) and (B.2), we obtain

$$\begin{aligned} \frac{\partial^2 Z_2}{\partial t^2} - c^2\frac{\partial^2 Z_2}{x} &= \frac{a^2c^2k^2}{4h\cos^2 kl}\{\cos[2k(l-x)+2\omega t] \\ &+ \cos[2k(l-x)-2\omega t] + 2k\cos 2k(l-x)\} \end{aligned} \tag{B.5}$$

Since we want to obtain the solutions which vary with the time, the third term in the right hand side of Eq. (B.5) can be neglected. Let the general solution of Eq. (B.5) take the form of

$$\begin{aligned} Z_2 &= F\{\sin[2k(l-x)+2\omega t] + \sin[2k(l-x)-2\omega t]\} \\ &+ G\{\cos[2k(l-x)+2\omega t] + \cos[2k(l-x)-2\omega t]\} \end{aligned} \tag{B.6}$$

where F and G are fountion of x only. Substitution of Z_2 into Eq. (B.5) yields

$$\begin{aligned} & [-c^2F'' - c^2 4kG'][\sin[2k(l-x)+2\omega t] + \sin[2k(l-x)-2\omega t]] \\ + \; & [c^2 4kF' - c^2 G''][\cos[2k(l-x)+2\omega t] + \cos[2k(l-x)-2\omega t]] \\ = \; & \frac{a^2c^2k^2}{4h\cos^2 kl}[\cos[2k(l-x)+2\omega t] + \cos[2k(l-x)-2\omega t]] \end{aligned} \tag{B.7}$$

Obviousely we get

$$-c^2F'' - c^2 4kG' = 0 \tag{B.8}$$

$$c^2 4kF' - c^2 G'' = \frac{a^2c^2k^2}{4h\cos^2 kl} \tag{B.9}$$

Integrating Eq. (B.9) with respect to x and set the integrating coefficient as zero, we have

$$4kF - G' = \frac{a^2k^2}{4h\cos^2 kl}x \tag{B.10}$$

Eliminating G' from Eqs. (B.8) and (B.10), we obtain

$$\frac{F''}{4k} + 4kF = \frac{a^2k^2}{4h\cos^2 kl}x \tag{B.11}$$

A general solution for F of Eq. (B.11) is

$$F_g = C_1 \cos 4kx \tag{B.12}$$

where C_1 is a constant to be determined from the boundary conditions. A particular solution of Eq. (B.11) is

$$F_p = \frac{a^2k}{16h\cos^2 kl}x \tag{B.13}$$

Therefore

$$\begin{aligned} F &= F_g + F_p \\ &= C_1 \cos 4kx + \frac{a^2k}{16h\cos^2 kl}x \end{aligned} \tag{B.14}$$

Substituting F into Eq. (B.8), we can obtain

$$G = C_1 \sin 4kx + C_2 \tag{B.15}$$

where C_2 is a integration constant. Plugging F and G into Eq. (B.6) and simplifying it, we have

$$\begin{aligned} Z_2 &= [C_1 \cos 4kx + \frac{a^2 k}{8h\cos^2 kl} x] \sin 2k(l-x) \cos 2\omega t \\ &+ [C_1 \sin 4kx + C_2] \cos 2k(l-x) \cos 2\omega t \end{aligned} \tag{B.16}$$

From the boundary condition (B.3), we obtain

$$[C_1 \cos 4kl + \frac{a^2 kl}{8h\cos^2 kl}](-2k) + C_1 4k \cos 4kl = 0 \tag{B.17}$$

Thus

$$C_1 = \frac{a^2 kl}{8h\cos^2 kl \cos 4kl} \tag{B.18}$$

Similarly from boundary condition (B.4), we can get

$$C_2 = -\frac{a^2 kl}{8h\cos^2 kl \cos 4kl} \tan 2kl \tag{B.19}$$

Substituting C_1 and C_2 into Eq. (B.16) and simplifying it, we consequently obtain

$$\begin{aligned} Z_2 &= \frac{a^2 k}{8h\cos^2 kl}[x \sin 2k(l-x) + \frac{l}{\cos 4kl} \sin 2k(l+x) \\ &- \frac{l}{\cos 4kl} \tan 2kl \cos 2k(l-x)] \cos 2\omega t \end{aligned} \tag{B.20}$$

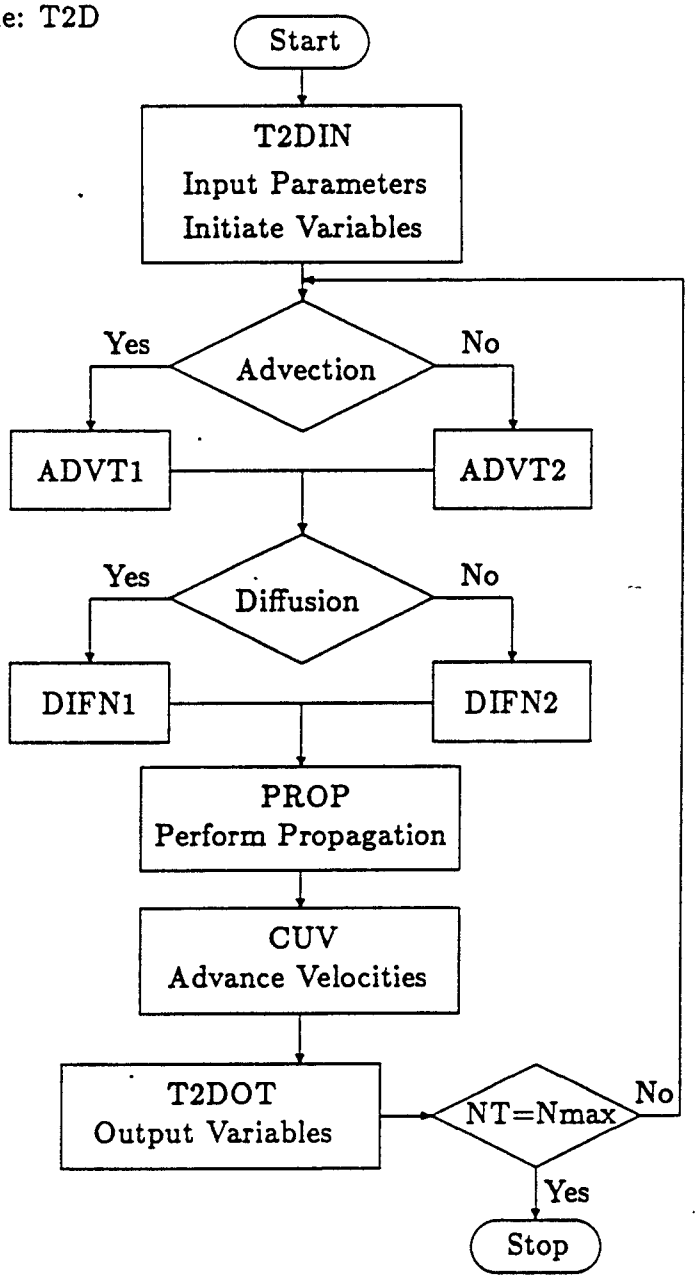
Substituting Z_2 into Eq. (B.1) and integrating with respect to x, we have

$$\begin{aligned} U_2 &= \frac{a^2 \omega}{8h\cos^2 kl}[x \cos 2k(l-x) + \frac{1}{2k} \sin 2k(l-x) \\ &- \frac{l}{\cos 4kl} \cos 2k(l+x) + \frac{l}{\cos 4kl} \tan 2kl \sin 2k(l-x)] \sin 2\omega t \end{aligned} \tag{B.21}$$

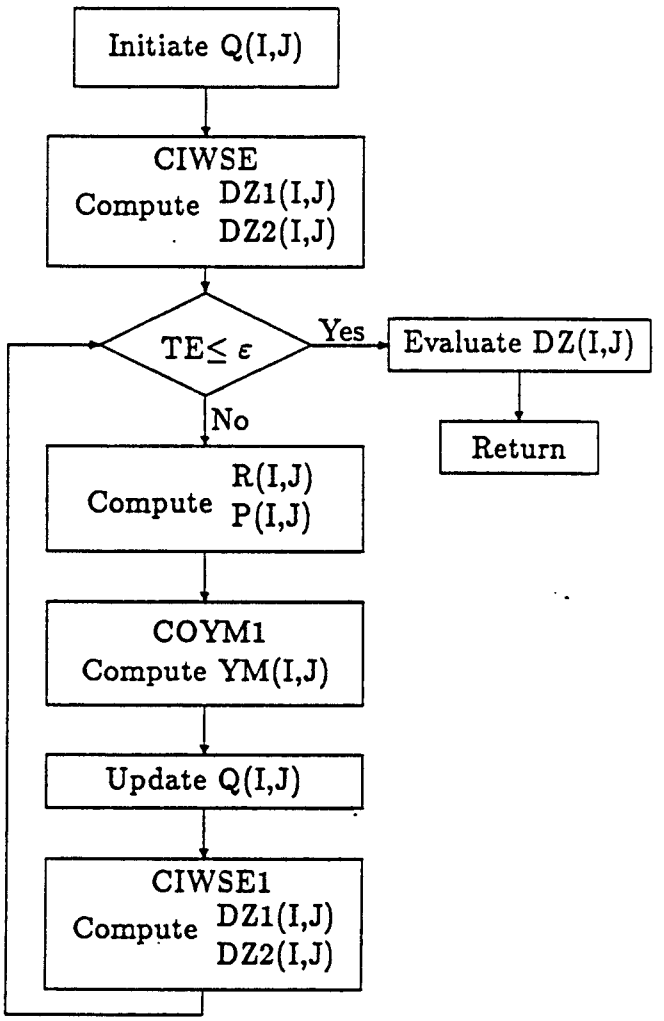
APPENDIX C
PROGRAM LISTING

C.1 Flow Chart

C.1.1 Main Routine: T2D



C.1.2 Subroutine: PROP



C.2 Program listing

C.2.1 Description of Parameters

KI0: Maximum number of grid points in the x-direction;

KJ0: Maximum number of grid points in the y-direction;

KX1(J), KX2(J), KY1(I), KY2(I): Grid numbers for the boundary's location;

CN: Manning coefficient;

F: Coriolis parameter;

G: Gravitational acceleration;

NT: Number of time steps;

HMIN: Minimum water depth;

DELT: Time increment;

EPSLON: Accuracy of the conjugate gradient iteration;

NCG: Number of the conjugate gradient iterations;

ALFA: Coefficient of implicitization;

DC: Horizontal diffusion coefficient;

ATA: Amplitude of the focing tide;

WOMAGA: Frequency of the forcing tide;

AX(I,J), BX(I,J), CX(I,J), DDX(I,J): Coefficients of the matrix equations for the propagation in the x-direction;

AY(I,J), BY(I,J), CY(I,J), DDY(I,J): Coefficients of the matrix equations for the propagation in the y-direction;

DELX(I): Space increment of (U, V) grid in the x-direction;

DELY(J): Space increment of (U, V) grid in the y-direction;

ZDELX(I): Space increment of Z grid in the x-direction;

ZDELY(J): Space increment of Z grid in the y-direction;

Z(I,J): Water surface elevation;

ZB(I,J): Bed elevation;

H(I,J): Water depth at Z grid;

HUV(I,J): Water depth at (U, V) grid;

ZP(I,J): Water surface elevation at the previous time;

HP(I,J): Water depth of Z grid at the previouse time;

HUVP(I,J): Water depth of (U, V) grid at the previous time;

U(I,J): Unit-width discharge in the x-direction;

V(I,J): Unit-width discharge in the y-direction;

PU(I,J): Velocity corresponding to U(I,J);

PV(I,J): Velocity corresponding to V(I,J);

UD(I,J): Unit-width discharge in the x-direction after the diffusion;

VD(I,J): Unit-width discharge in the y-direction after the diffusion;

UAD(I,J): Unit-width discharge in the x-direction after the advection;

VAD(I,J): Unit-width discharge in the y-direction after the advection;

W(I,J): Wind speed;

DZ(I,J): Increment of water surface elevation during one time step;

DZ1(I,J): Increment of water surface elevation due to the propagation in the x-direction;

DZ2(I,J): Increment of water surface elevation due to the propagation in the y-direction;

GAMA(I,J): Weighting coefficient;

P(I,J): Direction vector;

R(I,J): Residual vector;

Q(I,J): Lagrange coordinates;

HX(I,J): Water depth at the middle point of (I+1,J) and (I,J) on Z grid;

HY(I,J): Water depth at the middle point of (I,J) and (I,J+1) on Z grid;

C.2.2 Program listing

```
C*************** MAIN PROGRAM: T2D *************
C** THIS PROGRAM IS USED TO SIMULATE THE TWO-DIMENSIONAL
C   WATER MOTION IN ESTUARIES, LAKES AND SHALLOW SEAS.
C   IT IS DEVELOPED USING THE METHOD OF FRACTIONAL STEPS
C
      INCLUDE 'DIM.FOR'
C
C--- INPUT PARAMETERS AND INITIATE VARIABLES
C
      CALL T2DIN
C
C--- ADVACE VARIABLES
C
      DO 1000 NT=1,600
C
      IF(NT.GT.200) THEN
       DO 10 I=1,KIO
```

```
          DO 10 J=1,KJO
           W(I,J)=0.
10        CONTINUE
         ENDIF
C
C--- DETERMINE THE POSITION OF THE MOVING BOUNDARY
C
         CALL FMOVB
C
C--- EVALUATE WATER DEPTH AT 'UV' GRID ,
C    HYDRAULIC RADIUS HX, AND HY
C
         CALL UVDEP
C
C--- PERFORM CALCULATION OF THE ADVETION
C
         CALL ADVT1        ! FOR CASE OF ADVECTION
C        CALL ADVT2        ! FOR CASE OF NO ADVECTION
C
C--- CALCULATE THE DIFFUSION
C
         CALL DIFN1        ! FOR CASE OF DIFFUSION
C        CALL DIFN2        ! FOR CASE OF NO DIFFUSION
C
C--- COMPUTE THE PROPAGATION
C
         CALL PROP(NCG,EPSLON)
C
C--- ADVANCE THE UNIT-WIDTH DISCHARGES
C
         CALL CUV
C
C--- OUTPUT THE RESULTS
C
         CALL T2DOT
C
1000     CONTINUE
C
         STOP
         END


C****************** SUBROUTINE T2DIN ******************
C*** THIS SUBROUTINE IS DEVELOPED TO INPUT ALL NECESSARY
C    PARAMETERS AND DATA FOR RUNNING THE MAIN PROGRAM T2D
C
         SUBROUTINE T2DIN
C
         INCLUDE 'DIM.FOR'
C
         OPEN(UNIT=1,NAME='KX.DAT',STATUS='OLD')
         OPEN(UNIT=2,NAME='KY.DAT',STATUS='OLD')
C
C--- EVALUATE CONSTANT PARAMETERS
```

```
C
        ALFA=1.0
        G=980.          ! CM/SECOND**2
        CN=0.02         ! CM**(1/6.)
        DC=0.           ! CM**2/SEC.
        F=0.73E-4       ! 1/SEC.
        EPSLON=1.E-4
        HMIN=20.        ! CM
        DELT=180.       ! SECOND
        ATA=50.         ! CM
        WOMAGA=0.14E-3   ! 1/SEC.
        NCG=20
        KIO=32
        KJO=51
C
C--- INPUT THE POSITION OF THE BOUNDARY KX1(J),KX2(J),KY1(I),KY2(I)
C
        DO 2 J=1,KJO
         READ(1,3)KX1(J),KX2(J)
2       CONTINUE
3       FORMAT(6X,I2,5X,I2)
        CLOSE(1)
C
        DO 5 I=1,KIO
         READ(2,6)KY1(I),KY2(I)
5       CONTINUE
6       FORMAT(7X,I2,5X,I2)
        CLOSE(2)
C
C--- INPUT GRID STRUCTURE
        CALL GRID
C
C--- INITIATE VARIABLES
C
        CALL INIT
C
C--- EVALUATE WEIGHTING COEFFICIENTS
C
        DO 10 I=1,KIO
        DO 10 J=1,KJO
         IF(I.EQ.1) THEN
          GAMA(I,J)=0.5
         ELSE
          GAMA(I,J)=DELX(I)/(DELX(I)+DELX(I-1))
         ENDIF
10      CONTINUE
C
        RETURN
        END
C
C**************SUBROUTINE GRID *****************
C** PERFORM CONSTRUCTION OF THE GRID
C
        SUBROUTINE GRID
```

```
C
      INCLUDE 'DIM.FOR'
C
C--- EVALUATE THE SCALE OBTAINED FROM THE MAP OF LAKE OKEECHOBEE.
C
C     (1+7/16)IN=2 MILES
C     1 MILE=1.609 KM= 1609M =160900 CM
      SCALE=2*160900./(1.+7/16.)    ! CM/INCH
C
C--- INPUT THE GRID SIZE IN THE X-DIRECTION
C    AT (U,V) GRID
C
      DO 10 I=1,KIO-1
       IF(I.LE.15) THEN
        DELX(I)=0.5*SCALE
       ELSE
        IF(I.LE.20) THEN
         DELX(I)=DELX(I-1)+0.1*SCALE
        ELSE
         DELX(I)=1.*SCALE
        ENDIF
       ENDIF
10    CONTINUE
      DELX(KIO)=DELX(KIO-1)
C
C    AT Z GRID
C
      DO 20 I=2,KIO-1
       ZDELX(I)=(DELX(I)+DELX(I-1))/2.
20    CONTINUE
       ZDELX(1)=DELX(1)
       ZDELX(KIO)=ZDELX(KIO-1)
C
C--- INPUT GRID SIZE IN THE Y-DIRECTION
C    AT (U,V) GRID
C
      DO 30 J=1, KJO-1
       DELY(J)=0.5*SCALE
30    CONTINUE
       DELY(KJO)=DELY(KJO-1)
C
C    AT Z GRID
C
      DO 40 J=2,KJO-1
       ZDELY(J)=(DELY(J)+DELY(J-1))/2.
40    CONTINUE
       ZDELY(1)=DELY(1)
       ZDELY(KJO)=ZDELY(KJO-1)
C
      RETURN
      END
C
C***************** SUBROUTINE INIT **************
C** INPUT INITIAL VALUES OF VARIABLES
```

```
C
      SUBROUTINE INIT
C
      INCLUDE 'DIM.FOR'
      DIMENSION WDP(100,100),WDP1(100,100)
C
      OPEN (UNIT=1,NAME='DEPTH1.DAT',STATUS='OLD')
      OPEN (UNIT=2,NAME='DEPTH2.DAT',STATUS='OLD')
      OPEN (UNIT=3,NAME='DEPTH3.DAT',STATUS='OLD')
C
C--- READ WATER DEPTH AT Z GRID POINTS
C
      DO 10 J=2,KJO
       READ(1,11) WDP(2,J),WDP(3,J),WDP(4,J),WDP(5,J),WDP(6,J),
     &            WDP(7,J),WDP(8,J),WDP(9,J),WDP(10,J),WDP(11,J)
       READ(2,11) WDP(12,J),WDP(13,J),WDP(14,J),WDP(15,J),WDP(16,J),
     &            WDP(17,J),WDP(18,J),WDP(19,J),WDP(20,J),WDP(21,J)
       READ(3,12) WDP(22,J),WDP(23,J),WDP(24,J),WDP(25,J),WDP(26,J),
     &            WDP(27,J),WDP(28,J),WDP(29,J),WDP(30,J),WDP(31,J),
     &            WDP(32,J)
10    CONTINUE
11    FORMAT (4X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,
     &        2X,F4.1,2X,F4.1,2X,F4.1)
12    FORMAT (4X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,
     &        2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1)
C
      CLOSE(1)
      CLOSE(2)
      CLOSE(3)
C
C--- SMOOTH WATER DEPTH
C
      DO 5 I=2,KIO
      DO 5 J=2,KJO
       WDP1(I,J)=(WDP(I-1,J)+WDP(I,J)+WDP(I+1,J)+WDP(I-1,J+1)+WDP(I,J+1)
     &           +WDP(I+1,J+1)+WDP(I-1,J-1)+WDP(I,J-1)+WDP(I+1,J-1))/9.
5     CONTINUE
C
C--- TRANSFORM THE UNIT
C
      DO 15 J=2,KJO
      DO 15 I=2,KIO
       H(I,J)=WDP1(I,J)*30.48 ! CM
15    CONTINUE
C
      DO 16 J=2,KJO
       H(1,J)=H(2,J)
16    CONTINUE
C
      DO 17 I=1,KIO
       H(I,1)=H(I,2)
17    CONTINUE
C
      DO 18 I=1,KIO
```

```
        DO 18 J=1,KJO
         IF(H(I,J).LT.HMIN) THEN
          H(I,J)=HMIN
         ENDIF
18      CONTINUE
C
C--- EVALUATE THE WATER SURFACE ELEVATION  AT 'Z' GRID
C
        DO 20 I=1,KIO
        DO 20 J=1,KJO
         Z(I,J)=2000.      ! CM
20      CONTINUE
C
C--- EVALUATE THE BED ELEVATION ZB(I,J)
C
        DO 30 I=1,KIO
        DO 30 J=1,KJO
         ZB(I,J)=Z(I,J)-H(I,J)      ! CM
30      CONTINUE
C
C--- EVALUATE WATER DEPTH AT (U,V) GRID
C
        DO 40 J=1,KJO-1
        DO 40 I=1,KIO-1
         HUV(I,J)=(H(I,J)+H(I,J+1)+H(I+1,J)+H(I+1,J+1))/4.
40      CONTINUE
        DO 41 J=1,KJO
         HUV(KIO,J)=HUV(KIO-1,J)
41      CONTINUE
        DO 42 I=1,KIO
         HUV(I,KJO)=HUV(I,KJO-1)
42      CONTINUE
C
C--- INITIATE UNIT-WIDTH DISCHARGES AND VELOCITIES
C
        DO 50 I=1,KIO
        DO 50 J=1,KJO
         U(I,J)=0.      ! CM**2/SEC.
         V(I,J)=0.
         UD(I,J)=0.
         VD(I,J)=0.
         UAD(I,J)=0.
         VAD(I,J)=0.
         PU(I,J)=0.     ! CM/SEC.
         PV(I,J)=0.
50      CONTINUE
C
C--- INITIATE THE INCREMENT OF WATER SURFACE ELEVATION
C
        DO 60 I=1,KIO
        DO 60 J=1,KJO
         DZ(I,J)=0.     ! CM
60      CONTINUE
C
```

```
C--- INPUT THE WIND SPEED
C
        DO 70 J=2,KJO
         IF(J.LT.25) THEN
          N1=KX1(J-1)+1
          N2=KX2(J-1)
         ELSE
          N1=KX1(J)+1
          N2=KX2(J)
         ENDIF
        DO 70 I=N1,N2-1
         W(I,J)=2000.      ! CM/SEC.
70      CONTINUE
C
        RETURN
        END


C************** SUBROUTINE ADVT1 ***********************
C**THIS SUBROUTINE IS USED FOR SOLVING THE ADVECTION. USING THE METHOD
C  OF FRACTIONAL STEPS, THE TWO-DIMENSIONAL ADVECTION IS DIVIDED AS TWO
C  ONE-DIMENSIONAL ADVCTIONS WHICH ARE X-ADVECTION AND Y-ADVECTION. THE
C  IMPLICIT CHARACTERISTIC NUMERICAL SCHEME IS USED TO SOLVE EACH
C  ADVECTION. IT IS NOTED THAT THE FOLLOWING PROGRAM IS DEVELOPED
C  ACCORDING TO A RECTANGULAR BASIN. UX(I,J),VX(I,J): VELOCITY
C  COMPONENTS IN THE X- AND Y-DIRECTIONS AFTER X-ADVECTION. UY(I,J),
C  VY(I,J): VELOCITIES COMPONENTS IN THE X- AND Y-DIRECTIONS AFTER
C  Y-ADVECTION.
C
        SUBROUTINE ADVT
C
        INCLUDE 'DIM.FOR'
        DIMENSION A(100),B(100),C(100),D(100),X(100)
        DIMENSION UX(100,100),VX(100,100),UY(100,100),VY(100,100)
C
C--- X-ADVECTION
C       COMPUTATION OF UX(100,100)
C
        KI2=KIO-1
        KJ2=KJO-1
        DO 10 J=2,KJ2
        DO 20 I=2,KI2
C
         A(I)=1.
         B(I)=PU(I,J)*DELT/(DELX(I)+DELX(I-1))
         C(I)=-PU(I,J)*DELT/(DELX(I)+DELX(I-1))
         D(I)=PU(I,J)
C
        IF(I.EQ.2) THEN
         A(I)=A(I)+C(I)
         C(I)=0.
        ENDIF
C
        IF (I.EQ.KI2) THEN
```

```
          B(I)=0.
         ENDIF
C
20       CONTINUE
C
         KM=KI2-1
         CALL CUGA(X,A,B,C,D,KM)
         DO 30 I=2,KI2
          UX(I,J)=X(I)
30       CONTINUE
10       CONTINUE
C
         DO 40 J=2,KJ2
          UX(1,J)=UX(2,J)
40       CONTINUE
C
C        COMPUTATION OF VX(100,100)
C
         DO 50 J=2,KJ2
         DO 60 I=2,KI2
          A(I)=1.
          B(I)=PU(I,J)*DELT/(DELX(I)+DELX(I-1))
          C(I)=-PU(I,J)*DELT/(DELX(I)+DELX(I-1))
          D(I)=PV(I,J)
         IF(I.EQ.2) THEN
          A(I)=A(I)+C(I)
          C(I)=0.
         ENDIF
         IF (I.EQ.KI2) THEN
          A(I)=A(I)+B(I)
          B(I)=0.
         ENDIF
60       CONTINUE
C
         KM=KI2-1
         CALL CUGA(X,A,B,C,D,KM)
         DO 70 I=2,KI2
          VX(I,J)=X(I)
70       CONTINUE
50       CONTINUE
C
         DO 80 J=2,KJ2
          VX(1,J)=VX(2,J)
80       CONTINUE
C
C--- Y-ADVECTION
C        COMPUTATION OF UY(100,100)
C
         DO 110 I=2,KI2
         DO 120 J=2,KJ2
          A(J)=1.
          B(J)=VX(I,J)*DELT/(DELY(J)+DELY(J-1))
          C(J)=-VX(I,J)*DELT/(DELY(J)+DELY(J-1))
          D(J)=UX(I,J)
```

```
          IF(J.EQ.2) THEN
           A(J)=A(J)+C(J)
           C(J)=0.
          ENDIF
          IF(J.EQ.KJ2) THEN
           A(J)=A(J)+B(J)
           B(J)=0.
          ENDIF
120       CONTINUE
C
          KM=KJ2-1
          CALL CUGA(X,A,B,C,D,KM)
          DO 130 J=2,KJ2
           UY(I,J)=X(J)
130       CONTINUE
110       CONTINUE
C
C         COMPUTAITON OF VY(100,100)
C
          DO 150 I=2,KI2
          DO 160 J=2,KJ2
           A(J)=1.
           B(J)=VX(I,J)*DELT/(DELY(J)+DELY(J-1))
           C(J)=-VX(I,J)*DELT/(DELY(J)+DELY(J-1))
           D(J)=VX(I,J)
          IF(J.EQ.2) THEN
           C(J)=0.
          ENDIF
          IF(J.EQ.KJ2) THEN
           B(J)=0.
          ENDIF
160       CONTINUE
C
          KM=KJ2-1
          CALL CUGA(X,A,B,C,D,KM)
          DO 170 J=2,KJ2
           VY(I,J)=X(J)
170       CONTINUE
150       CONTINUE
C
C--- IMPOSE BOUNDARY CONDITIONS
C
          DO 200 J=2,KJ2
           UY(1,J)=UY(2,J)
           VY(1,J)=VY(2,J)
           UY(KI,J)=0.
           VY(KI,J)=VY(KI2,J)
200       CONTINUE
C
          DO 220 I=1,KIO
           UY(I,1)=UY(I,2)
           UY(I,KJO)=UY(I,KJ2)
           VY(I,1)=0.
           VY(I,KJO)=0.
```

```
220     CONTINUE
C
C--- TRANSFER UY,VY INTO UAD,VAD
C
        DO 240 I=1,KIO
        DO 240 J=1,KJO
         UAD(I,J)=UY(I,J)*HUV(I,J)
         VAD(I,J)=VY(I,J)*HUV(I,J)
240     CONTINUE
C
        RETURN
        END


C*************** SUBROUTINE ADVT2 **************
C**THIS SUBROUTINE IS USED FOR THE CASE OF NO ADVECTION.
C
        SUBROUTINE ADVT2
        INCLUDE 'DIM.FOR'
C
        DO 20 I=1,KIO
        DO 20 J=KY1(I),KY2(I)
         UAD(I,J)=U(I,J)
         VAD(I,J)=V(I,J)
20      CONTINUE
C
        RETURN
        END


C*************** SUBROUTINE DIFN1 *****************
C**PERFORM THE CALCULATION OF THE DIFFUSION
C  THE DIFFUSION IS SOLVED USING THE ADI METHOD.
C
        SUBROUTINE DIFN1
C
        DIMENSION A(150),B(150),C(150), D(150),X(150)
        DIMENSION AA(150),BB(150),CC(150),DD(150)
        INCLUDE 'DIM.FOR'
C
C--- COMPUTE UD(I,J)
C
        KJ2=KJO-1
        DO 50 J=2,KJ2
         N1=KX1(J)+1
         N2=KX2(J)-1
        DO 60 I=N1,N2
         IF (J.EQ.2) THEN
          TERM1=2*DC*(UAD(I,J+1)-UAD(I,J))
     &           /DELY(J)/(DELY(J)+DELY(J-1))
         ELSE
          IF (J.EQ.KJ2) THEN
           TERM1=-2*DC*(UAD(I,J)-UAD(I,J-1))
     &            /DELY(J-1)/(DELY(J-1)+DELY(J))
```

```
          ELSE
           TM=(UAD(I,J+1)-UAD(I,J))/DELY(J)
     &        -(UAD(I,J)-UAD(I,J-1))/DELY(J-1)
           TERM1=2*DC*TM/(DELY(J-1)+DELY(J))
          ENDIF
         ENDIF
C
         IF(I.EQ.N1) THEN
          A(I)=1.+DC*DELT/DELX(I)/DELX(I)
          B(I)=-DC*DELT/DELX(I)/DELX(I)
          C(I)=0.
          D(I)=UAD(I,J)+DELT*(F*VAD(I,J)+TERM1)
         ELSE
          IF(I.EQ.N2) THEN
           A(I)=1.+2*DC*DELT/DELX(I)/DELX(I-1)
           B(I)=0.
           C(I)=-2*DC*DELT/DELX(I)/DELX(I-1)
           D(I)=UAD(I,J)+DELT*(F*VAD(I,J)+TERM1)
          ELSE
           A(I)=1.+2*DC*DELT/DELX(I)/DELX(I-1)
           B(I)=-2*DC*DELT/DELX(I)/(DELX(I)+DELX(I-1))
           C(I)=-2*DC*DELT/DELX(I-1)/(DELX(I)+DELX(I-1))
           D(I)=UAD(I,J)+DELT*(F*VAD(I,J)+TERM1)
          ENDIF
         ENDIF
60       CONTINUE
C
        N=N2-N1+1
        DO 65 I=2,N+1
         AA(I)=A(N1-2+I)
         BB(I)=B(N1-2+I)
         CC(I)=C(N1-2+I)
         DD(I)=D(N1-2+I)
65      CONTINUE
C
        CALL CUGA(X,AA,BB,CC,DD,N)
C
        DO 70 I=N1,N2
         UD(I,J)=X(I-N1+2)
70      CONTINUE
C
50      CONTINUE
C
C--- COMPUTE VD(I,J)
C
        KI2=KI0-1
        DO 110 I=2,KI2
         N1=KY1(I)+1
         N2=KY2(I)-1
        DO 120 J=N1,N2
         IF (I.EQ.2) THEN
           TM=(VAD(I+1,J)-VAD(I,J))/DELX(I)
          TERM1=DC*TM/DELX(I)
         ELSE
```

```
          IF(I.EQ.KI2) THEN
            TM=-(VAD(I,J)-VAD(I-1,J))/DELX(I-1)
           TERM1=2*DC*TM/(DELX(I)+DELX(I-1))
          ELSE
            TM=(VAD(I+1,J)-VAD(I,J))/DELX(I)
     &         -(VAD(I,J)-VAD(I-1,J))/DELX(I-1)
           TERM1=2*DC*TM/(DELX(I)+DELX(I-1))
          ENDIF
         ENDIF
C
         IF (J.EQ.N1) THEN
          A(J)=1.+2*DC*DELT/DELY(J)/DELY(J-1)
          B(J)=-2*DC*DELT/DELY(J)/(DELY(J)+DELY(J-1))
          C(J)=0.
          D(J)=VAD(I,J)+DELT*(-F*UAD(I,J)+TERM1)
         ELSE
          IF (J.EQ. N2) THEN
           A(J)=1.+2*DC*DELT/DELY(J-1)/DELY(J-1)
           B(J)=0.
           C(J)=-2*DC*DELT/DELY(J-1)/(DELY(J)+DELY(J-1))
           D(J)=VAD(I,J)+DELT*(-F*UAD(I,J)+TERM1)
          ELSE
           A(J)=1.+2*DC*DELT/DELY(J)/DELY(J-1)
           B(J)=-2.*DC*DELT/DELY(J)/(DELY(J)+DELY(J-1))
           C(J)=-2.*DC*DELT/DELY(J-1)/(DELY(J)+DELY(J-1))
           D(J)=VAD(I,J)+DELT*(-F*UAD(I,J)+TERM1)
          ENDIF
         ENDIF
120      CONTINUE
C
         N=N2-N1+1
         DO 130 J=2,N+1
          AA(J)=A(N1-2+J)
          BB(J)=B(N1-2+J)
          CC(J)=C(N1-2+J)
          DD(J)=D(N1-2+J)
130      CONTINUE
C
         CALL CUGA(X,AA,BB,CC,DD,N)
C
         DO 140 J=N1,N2
          VD(I,J)=X(J-N1+2)
140      CONTINUE
110      CONTINUE
C
C        IMPOSE THE CLOSE BOUNDARY CONDITIONS
C
         DO 200 J=1,KJO
          UD(1,J)=0.
          VD(1,J)=0.
          UD(KIO,J)=0.
          VD(KIO,J)=0.
200      CONTINUE
         DO 210 I=2,KIO-1
```

```
          IF(I.LT.25) THEN
           N1=KY1(I-1)
           N2=KY2(I-1)
          ELSE
           N1=KY1(I+1)
           N2=KY2(I+1)
          ENDIF
         DO 220 J=1,N1
          UD(I,J)=0.
          VD(I,J)=0.
220       CONTINUE
         DO 230 J=N2,KJO
          UD(I,J)=0.
          VD(I,J)=0.
230      CONTINUE
210      CONTINUE
C
         RETURN
         END


C*************** SUBROUTINE DIFN2 *************
C**THIS SUBROUTINE IS USED FOR THE CASE OF NO DIFFUSION.
C
         SUBROUTINE DIFN2
         INCLUDE 'DIM.FOR'
C
         DO 10 J=2,KJO-1
         DO 20 I=1,KIO-1
          UD(I,J)=UAD(I,J)+DELT*F*VAD(I,J)
          VD(I,J)=VAD(I,J)-DELT*F*UD(I,J)
20       CONTINUE
10       CONTINUE
C
C--- IMPOSE THE BOUNDARY CONDITIONS
C
         DO 30 J=1,KJO-1
          UD(KIO,J)=0.
          VD(KIO,J)=VD(KIO-1,J)
30       CONTINUE
         DO 40 I=1,KIO-1
          UD(I,1)=UD(I,2)
          UD(I,KJO)=UD(I,KJO-1)
          VD(I,1)=0.
          VD(I,KJO)=0.
40       CONTINUE
C
         RETURN
         END
```

```
C*************** SUBROUTINE PROP *************
C** PERFORM THE COMPUTATION OF THE PROPAGATION
C   THE PROPAGATION IS SOLVED USING THE CONJUGATE GRADIENT
C   ITERATION METHOD.
C
      SUBROUTINE PROP(NCG,EPSLON)
C
      INCLUDE 'DIM.FOR'
      DIMENSION Q(150,150),P(150,150),R(150,150)
C
C
C--- PERFORM THE CONJUGATE GRADIENT ITERATION
C
      DO 50 N=1,NCG
C
      IF (NT.EQ.1) THEN
       IF(N.EQ.1) THEN
        DO 60 I=2,KIO
         IF(I.LT.25) THEN
          N1=KY1(I-1)+1
          N2=KY2(I-1)
         ELSE
          N1=KY1(I)+1
          N2=KY2(I)
         ENDIF
        DO 60 J=N1,N2
         Q(I,J)=0.
60       CONTINUE
       ENDIF
      ENDIF
C
C     COMPUTE DZ1(I,J) AND DZ2(I,J)
C
      IF(N.EQ.1) THEN
       CALL CIWSE(Q)
      ELSE
       CALL CIWSE1(Q)
      ENDIF
C
C     CALCULATE THE SUM OF ERRORS: TE
C
      TE=0.
      DO 80 I=2,KIO
       IF(I.LT.25) THEN
        N1=KY1(I-1)+1
        N2=KY2(I-1)
       ELSE
        N1=KY1(I)+1
        N2=KY2(I)
       ENDIF
      DO 80 J=N1,N2
       TE=TE+(DZ1(I,J)-DZ2(I,J))**2
80    CONTINUE
C
```

```
      IF (TE.LT.EPSLON) GOTO 200
C
C     EVALUATE THE COEFFICIENT A2
C
      IF(N.NE.1) THEN
       A2=TE/TE1
      ENDIF
       TE1=TE
C
C     COMPUTE RESIDUAL VECTOR R(I,J)
C
      DO 90 I=2,KIO
       IF(I.LT.25) THEN
        N1=KY1(I-1)+1
        N2=KY2(I-1)
       ELSE
        N1=KY1(I)+1
        N2=KY2(I)
       ENDIF
      DO 90 J=N1,N2
       R(I,J)=DZ1(I,J)-DZ2(I,J)
90    CONTINUE
C
C     COMPUTE THE DIRECTION VECTOR P(I,J)
C
      DO 100 I=2,KIO
       IF(I.LT.25) THEN
        N1=KY1(I-1)+1
        N2=KY2(I-1)
       ELSE
        N1=KY1(I)+1
        N2=KY2(I)
       ENDIF
      DO 100 J=N1,N2
       IF (N.EQ.1) THEN
        P(I,J)=R(I,J)
       ELSE
        P(I,J)=R(I,J)+A2*P(I,J)
       ENDIF
100   CONTINUE
C
C     COMPUTE YM1(I,J) AND YM2(I,J)
C
      CALL COYM1(P)
C
C     EVALUATE THE COEFFICIENT A1
C
      PDZ=0.
      PYM=0.
      DO 120 I=2,KIO
       IF(I.LT.25) THEN
        N1=KY1(I-1)+1
        N2=KY2(I-1)
       ELSE
```

```
          N1=KY1(I)+1
          N2=KY2(I)
         ENDIF
        DO 120 J=N1,N2
         PDZ=PDZ+(DZ1(I,J)-DZ2(I,J))*P(I,J)
         PYM=PYM+(YM1(I,J)+YM2(I,J))*P(I,J)
120     CONTINUE
        A1=PDZ/PYM
C
C       UPDATE THE LAGRANGE COORDINATE Q(I,J)
C
        DO 140 I=2,KIO
         IF(I.LT.25) THEN
          N1=KY1(I-1)+1
          N2=KY2(I-1)
         ELSE
          N1=KY1(I)+1
          N2=KY2(I)
         ENDIF
        DO 140 J=N1,N2
         Q(I,J)=Q(I,J)+A1*P(I,J)
140     CONTINUE
C
50      CONTINUE
C
C--- EVALUATE DZ(I,J)
C
200     DO 160 I=2,KIO
         IF(I.LT.25) THEN
          N1=KY1(I-1)+1
          N2=KY2(I-1)
         ELSE
          N1=KY1(I)+1
          N2=KY2(I)
         ENDIF
        DO 160 J=N1,N2
         DZ(I,J)=(DZ1(I,J)+DZ2(I,J))/2.
160     CONTINUE
C
C--- IMPOSE THE BOUNDARY CONDITIONS
C
        DO 170 I=2,KIO
         IF(I.LT.25) THEN
          DZ(I,KY1(I-1))=DZ(I,KY1(I-1)+1)
          DZ(I,KY2(I-1)+1)=DZ(I,KY2(I-1))
         ELSE
          DZ(I,KY1(I))=DZ(I,KY1(I)+1)
          DZ(I,KY2(I)+1)=DZ(I,KY2(I))
         ENDIF
170     CONTINUE
        DO 180 J=KY1(1),KY2(1)
         DZ(1,J)=DZ(2,J)    !CLOSE BOUNDARY AT I=1
180     CONTINUE
        DO 190 J=KY1(KIO),KY2(KIO)
```

```
          DZ(KIO+1,J)=DZ(KIO,J)  !CLOSE BOUNDARY AT I=KIO
190       CONTINUE
          DO 195 J=2,KJO
           IF(J.LT.25) THEN
            DZ(I,KX1(J-1))=DZ(I,KX1(J-1)+1)
            DZ(I,KX2(J-1)+1)=DZ(I,KX2(J-1))
           ELSE
            DZ(I,KX1(J))=DZ(I,KX1(J)+1)
            DZ(I,KX2(J)+1)=DZ(I,KX2(J))
           ENDIF
195       CONTINUE
C
          RETURN
          END


C****************** SUBROUTINE CIWSE ******************
C** PERFORM THE COMPUTATION OF DZ1(I,J) AND DZ2(I,J),GIVEN Q(I,J)
C
          SUBROUTINE CIWSE(Q)
C
          DIMENSION Q(150,150)
          DIMENSION A(150),B(150),C(150),D(150),X(150)
          INCLUDE 'DIM.FOR'
C
C--- COMPUTATION OF DZ1(I,J)
C
         DO 20 J=2,KJO
          IF (J.LT.25) THEN
           N1=KX1(J-1)+1
           N2=KX2(J-1)
          ELSE
           N1=KX1(J)+1
           N2=KX2(J)
          ENDIF
         DO 30 I=N1,N2
          DX=(ZDELX(I)+ZDELX(I-1))/2.
          IF(I.EQ.N2) THEN
           TERM1=1./2./G/DELT/DELT
            TM=HX(I-1,J)+(Z(I,J)-Z(I-1,J))*GAMA(I-1,J)
           TERM2=ALFA**2*TM/DX/ZDELX(I-1)
            TM=UD(I,J)/HX(I,J)-GAMA(I-1,J)*UD(I-1,J)/HX(I-1,J)
           TERM3=ALFA*TM/G/DELT/DX
          AX(I,J)=TERM1+TERM2+TERM3
C
          BX(I,J)=0.
C
            TM=-HX(I-1,J)+(1-GAMA(I-1,J))*(Z(I,J)-Z(I-1,J))
           TERM1=ALFA**2*TM/DX/ZDELX(I-1)
            TM=(1-GAMA(I-1,J))*UD(I-1,J)
           TERM2=-TM*ALFA/G/DELT/HX(I-1,J)/DX
          CX(I,J)=TERM1+TERM2
C
            TM=(1-ALFA)*(U(I,J)-U(I-1,J))+ALFA*(UD(I,J)-UD(I-1,J))
```

```
      TERM1=-TM/G/DELX(I-1)/DELT
       TM=-HX(I-1,J)*(Z(I,J)-Z(I-1,J))
      TERM2=ALFA*TM/DX/ZDELX(I-1)
        FC=116.*CN**2/HR(I,J)**(1./3.)
       TM1=SQRT(U(I,J)**2+V(I,J)**2)*U(I,J)*FC/8./HX(I,J)**2
        FC=116.*CN**2/HR(I-1,J)**(1./3.)
       TM2=SQRT(U(I-1,J)**2+V(I-1,J)**2)*U(I-1,J)
     &            *FC/8./HX(I-1,J)**2
       TM3=3.*0.000001*ABS(W(I,J))*W(I,J)
       TM4=3.*0.000001*ABS(W(I-1,J))*W(I-1,J)
      TERM3=ALFA*(TM1-TM2-TM3+TM4)/G/DX
     DDX(I,J)=TERM1+TERM2+TERM3-Q(I,J)
C
     ELSE
C
      TERM1=1./2./G/DELT/DELT
       TM1=HX(I,J)-(Z(I+1,J)-Z(I,J))*(1-GAMA(I,J))
       TM2=HX(I-1,J)+(Z(I,J)-Z(I-1,J))*GAMA(I-1,J)
       TM=TM1/ZDELX(I)+TM2/ZDELX(I-1)
      TERM2=ALFA**2*TM/DX
       TM1=UD(I,J)*(1-GAMA(I,J))/HX(I,J)
       TM2=UD(I-1,J)*GAMA(I-1,J)/HX(I-1,J)
      TERM3=ALFA*(TM1-TM2)/G/DELT/DX
     AX(I,J)=TERM1+TERM2+TERM3
C
      TERM1=-ALFA**2*HX(I,J)/DX/ZDELX(I)
      TERM2=-ALFA**2*(Z(I+1,J)-Z(I,J))*GAMA(I,J)/DX/ZDELX(I)
      TERM3=ALFA*UD(I,J)*GAMA(I,J)/G/DELT/HX(I,J)/DX
     BX(I,J)=TERM1+TERM2+TERM3
C
      TERM1=-ALFA**2*HX(I-1,J)/DX/ZDELX(I-1)
      TERM2=ALFA**2*(Z(I,J)-Z(I-1,J))
     &         *(1-GAMA(I-1,J))/DX/ZDELX(I-1)
      TERM3=-ALFA*UD(I-1,J)*(1-GAMA(I-1,J))
     &         /G/DELT/HX(I-1,J)/DX
     CX(I,J)=TERM1+TERM2+TERM3
C
      TERM1=-(1-ALFA)*(U(I,J)-U(I-1,J))/G/DELT/DELX(I-1)
      TERM2=-ALFA*(UD(I,J)-UD(I-1,J))/G/DELT/DELX(I-1)
       TM1=HX(I,J)*(Z(I+1,J)-Z(I,J))/ZDELX(I)
       TM2=HX(I-1,J)*(Z(I,J)-Z(I-1,J))/ZDELX(I-1)
      TERM3=ALFA*(TM1-TM2)/DX
        FC=116.*CN**2/HR(I,J)**(1./3.)
       TM1=SQRT(U(I,J)**2+V(I,J)**2)*U(I,J)*FC/8./HX(I,J)**2
        FC=116.*CN**2/HR(I-1,J)**(1./3.)
       TM2=SQRT(U(I-1,J)**2+V(I-1,J)**2)*U(I-1,J)
     &           *FC/8./HX(I-1,J)**2
       TM3=3.*0.000001*ABS(W(I,J))*W(I,J)
       TM4=3.*0.000001*ABS(W(I-1,J))*W(I-1,J)
      TERM4=ALFA*(TM1-TM2-TM3+TM4)/G/ZDELX(I-1)
     DDX(I,J)=TERM1+TERM2+TERM3+TERM4-Q(I,J)
C
     IF(I.EQ.N1) THEN
      AX(I,J)=AX(I,J)+CX(I,J)    !CLOSE BOUNDARY AT I=2
```

```
          CX(I,J)=0.
         ENDIF
C
         ENDIF
30      CONTINUE
C
        N=N2-N1+1
        DO 35 I=2,N+1
         A(I)=AX(N1-2+I,J)
         B(I)=BX(N1-2+I,J)
         C(I)=CX(N1-2+I,J)
         D(I)=DDX(N1-2+I,J)
35      CONTINUE
C
        CALL CUGA(X,A,B,C,D,N)
C
        DO 40 I=N1,N2
         DZ1(I,J)=X(I-N1+2)
40      CONTINUE
C
20      CONTINUE
C
C--- COMPUTATION OF DZ2(I,J)
C
        DO 50 I=2,KIO
         IF(I.LT.25) THEN
          N1=KY1(I-1)+1
          N2=KY2(I-1)
         ELSE
          N1=KY1(I)+1
          N2=KY2(I)
         ENDIF
        DO 60 J=N1,N2
         DY=(ZDELY(J)+ZDELY(J-1))/2.
         IF (J.EQ.N1) THEN
          TERM1=1./2./G/DELT/DELT
           TM=-HY(I,J)+(1-GAMA(I,J))*(Z(I,J+1)-Z(I,J))
          TERM2=-ALFA**2*TM/DY/ZDELY(J)
           TM=(1-GAMA(I,J))*VD(I,J)
          TERM3=ALFA*TM/G/DELT/DY/HY(I,J)
         AY(I,J)=TERM1+TERM2+TERM3
C
           TM=HY(I,J)+GAMA(I,J)*(Z(I,J+1)-Z(I,J))
          TERM1=-ALFA**2*TM/DY/ZDELY(J)
           TM=GAMA(I,J)*VD(I,J)
          TERM2=ALFA*TM/G/DELT/DY/HY(I,J)
         BY(I,J)=TERM1+TERM2
C
         CY(I,J)=0.
C
          TERM1=-((1-ALFA)*V(I,J)+ALFA*VD(I,J))/G/DELT/DELY(J-1)
          TERM2=ALFA*HY(I,J)*(Z(I,J+1)-Z(I,J))/ZDELY(J)/DY
            FC=116.*CN**2/HR(I,J)**(1./3.)
           TM1=FC*SQRT(U(I,J)**2+V(I,J)**2)*V(I,J)/8./HY(I,J)**2
```

```
          TERM3=ALFA*TM1/G/DY
          DDY(I,J)=TERM1+TERM2+TERM3+Q(I,J)
C
          ELSE
           IF(J.EQ.N2) THEN
C
            TERM1=1./2./G/DELT/DELT
             TM=HY(I,J-1)+(Z(I,J)-Z(I,J-1))*GAMA(I,J-1)
            TERM2=ALFA**2*TM/ZDELY(J-1)/DY
             TM=GAMA(I,J-1)*VD(I,J-1)
            TERM3=-ALFA*TM/G/DELT/DY/HY(I,J-1)
           AY(I,J)=TERM1+TERM2+TERM3
C
           BY(I,J)=0.
C
             TM=-HY(I,J-1)+(Z(I,J)-Z(I,J-1))*(1-GAMA(I,J-1))
            TERM1=ALFA**2*TM/ZDELY(J-1)/DY
             TM=(1-GAMA(I,J-1))*VD(I,J-1)
            TERM2=-ALFA*TM/G/DELT/DY/HY(I,J-1)
           CY(I,J)=TERM1+TERM2
C
             TM1=(1-ALFA)*V(I,J-1)+ALFA*VD(I,J-1)
            TERM1=TM1/G/DELT/DELY(J-1)
            TERM2=-ALFA*HY(I,J-1)*(Z(I,J)-Z(I,J-1))/DY/ZDELY(J-1)
              FC=116.*CN**2/HR(I,J-1)**(1./3.)
             TM1=FC*SQRT(U(I,J-1)**2+V(I,J-1)**2)*V(I,J-1)
     &                 /8./HY(I,J-1)**2
            TERM3=-ALFA*TM1/G/DY
           DDY(I,J)=TERM1+TERM2+TERM3+Q(I,J)
C
           ELSE
C
            TERM1=1./2./G/DELT/DELT
             TM1=-HY(I,J)+(Z(I,J+1)-Z(I,J))*(1-GAMA(I,J))
             TM2=-HY(I,J-1)-(Z(I,J)-Z(I,J-1))*GAMA(I,J-1)
             TM=TM1/ZDELY(J)+TM2/ZDELY(J-1)
            TERM2=-ALFA**2*TM/DY
             TM1=VD(I,J)*(1-GAMA(I,J))/HY(I,J)
             TM2=VD(I,J-1)*GAMA(I,J-1)/HY(I,J-1)
            TERM3=ALFA*(TM1-TM2)/G/DELT/DY
           AY(I,J)=TERM1+TERM2+TERM3
C
            TERM1=-ALFA**2*(HY(I,J)+GAMA(I,J)
     &          *(Z(I,J+1)-Z(I,J)))/DY/ZDELY(J)
            TERM2=ALFA*GAMA(I,J)*VD(I,J)/G/DELT/DY/HY(I,J)
           BY(I,J)=TERM1+TERM2
C
            TERM1=-ALFA**2*HY(I,J-1)/DY/ZDELY(J-1)
            TERM2=ALFA**2*(Z(I,J)-Z(I,J-1))
     &          *(1-GAMA(I,J-1))/DY/ZDELY(J-1)
            TERM3=-ALFA*VD(I,J-1)*(1-GAMA(I,J-1))
     &           /G/DELT/DY/HY(I,J-1)
           CY(I,J)=TERM1+TERM2+TERM3
C
```

```
            TERM1=-(1-ALFA)*(V(I,J)-V(I,J-1))/G/DELT/DELY(J-1)
            TERM2=-ALFA*(VD(I,J)-VD(I,J-1))/G/DELT/DELY(J-1)
            TERM3=ALFA*HY(I,J)*(Z(I,J+1)-Z(I,J))/DY/ZDELY(J)
            TERM4=-ALFA*HY(I,J-1)*(Z(I,J)-Z(I,J-1))/DY/ZDELY(J-1)
C
-------COMPUTATION OF FRICTION-------
C
              FC=116.*CN**2/HR(I,J)**(1./3.)
             TM1=FC*SQRT(U(I,J)**2+V(I,J)**2)*V(I,J)/8./HY(I,J)**2
              FC=116.*CN**2/HR(I,J-1)**(1./3.)
             TM2=FC*SQRT(U(I,J-1)**2+V(I,J-1)**2)
     &           *V(I,J-1)/8./HY(I,J-1)**2
           TERM5=ALFA/G/DY*(TM1-TM2)
          DDY(I,J)=TERM1+TERM2+TERM3+TERM4+TERM5+Q(I,J)
C
           ENDIF
          ENDIF
60       CONTINUE
C
         N=N2-N1+1
         DO 65 J=2,N+1
          A(J)=AY(I,N1-2+J)
          B(J)=BY(I,N1-2+J)
          C(J)=CY(I,N1-2+J)
          D(J)=DDY(I,N1-2+J)
65       CONTINUE
C
         CALL CUGA(X,A,B,C,D,N)
C
         DO 70 J=N1,N2
          DZ2(I,J)=X(J-N1+2)
70       CONTINUE
C
50       CONTINUE
C
         RETURN
         END


C****************** SUBROUTINE CIWSE1 ************
C** PERFORM THE COMPUTATION OF DZ1(I,J) AND DZ2(I,J)
C
         SUBROUTINE CIWSE1(Q)
C
         DIMENSION Q(150,150)
         DIMENSION A(150),B(150),C(150),D(150),X(150)
         INCLUDE 'DIM.FOR'
C
C--- COMPUTATION OF DZ1(I,J)
C
        DO 20 J=2,KJ0
         IF(J.LT.25) THEN
          N1=KX1(J-1)+1
          N2=KX2(J-1)
```

```
          ELSE
           N1=KX1(J)+1
           N2=KX2(J)
          ENDIF
         N=N2-N1+1
         DO 30 I=2,N+1
          A(I)=AX(N1-2+I,J)
          B(I)=BX(N1-2+I,J)
          C(I)=CX(N1-2+I,J)
          D(I)=DDX(N1-2+I,J)-Q(N1-2+I,J)
30       CONTINUE
C
        CALL CUGA(X,A,B,C,D,N)
C
         DO 40 I=N1,N2
          DZ1(I,J)=X(I-N1+2)
40       CONTINUE
C
20      CONTINUE
C
C--- COMPUTATION OF DZ2(I,J)
C
         DO 50 I=2,KIO
          IF(I.LT.25) THEN
           N1=KY1(I-1)+1
           N2=KY2(I-1)
          ELSE
           N1=KY1(I)+1
           N2=KY2(I)
          ENDIF
         N=N2-N1+1
         DO 60 J=2,N+1
          A(J)=AY(I,N1-2+J)
          B(J)=BY(I,N1-2+J)
          C(J)=CY(I,N1-2+J)
          D(J)=DDY(I,N1-2+J)+Q(I,N1-2+J)
60       CONTINUE
C
         CALL CUGA(X,A,B,C,D,N)
C
         DO 70 J=N1,N2
          DZ2(I,J)=X(J-N1+2)
70       CONTINUE
C
50      CONTINUE
C
        RETURN
        END


C*************** SUBROUTINE COYM1 ***************
C** PERFORM THE COMPUTATION OF YM1(I,J) AND YM2(I,J),
C   GIVEN P(I,J)
C
```

```
      SUBROUTINE COYM1(P)
C
      DIMENSION P(150,150)
      DIMENSION A(150),B(150),C(150),D(150),X(150)
      INCLUDE 'DIM.FOR'
C
C--- COMPUTATION OF YM1(I,J)
C
      DO 20 J=2,KJO
       IF(J.LT.25) THEN
        N1=KX1(J-1)+1
        N2=KX2(J-1)
       ELSE
        N1=KX1(J)+1
        N2=KX2(J)
       ENDIF
       N=N2-N1+1
       DO 30 I=2,N+1
        A(I)=AX(N1-2+I,J)
        B(I)=BX(N1-2+I,J)
        C(I)=CX(N1-2+I,J)
        D(I)=P(N1-2+I,J)
30     CONTINUE
C
       CALL CUGA(X,A,B,C,D,N)
C
       DO 40 I=N1,N2
        YM1(I,J)=X(I-N1+2)
40     CONTINUE
C
20    CONTINUE
C
C--- COPUTATION OF YM2(I,J)
C
      DO 50 I=2,KIO
       IF(I.LT.25) THEN
        N1=KY1(I-1)+1
        N2=KY2(I-1)
       ELSE
        N1=KY1(I)+1
        N2=KY2(I)
       ENDIF
       N=N2-N1+1
       DO 60 J=2,N+1
        A(J)=AY(I,N1-2+J)
        B(J)=BY(I,N1-2+J)
        C(J)=CY(I,N1-2+J)
        D(J)=P(I,N1-2+J)
60     CONTINUE
C
       CALL CUGA(X,A,B,C,D,N)
C
       DO 70 J=N1,N2
        YM2(I,J)=X(J-N1+2)
```

```
70       CONTINUE
C
50      CONTINUE
C
        RETURN
        END


C****************** SUBROUTINE CUV ***************
C** ADVANCE THE VELOCITIES
C
        SUBROUTINE CUV
        DIMENSION ZP(150,150),HP(150,150),HUVP(150,150)
        INCLUDE 'DIM.FOR'
C
C--- RESTORE THE OLD WATER SURFACE ELEVATION AND WATER DEPTH
C    TO ZP(I,J), HP(I,J) AND HUVP(I,J)
C
        DO 20 I=1,KIO
        DO 20 J=1,KJO
         ZP(I,J)=Z(I,J)
         HP(I,J)=H(I,J)
         HUVP(I,J)=HUV(I,J)
20      CONTINUE
C
C--- ADVANCE WATER SURFACE ELEVATION AND WATER DEPTH
C
        CALL DEPT
C
C--- ADVANCE UNIT-WIDTH DISCHARGES U(I,J) AND V(I,J)
C
        DO 60 I=2,KIO-1
         IF(I.LT.25) THEN
          N1=KY1(I-1)+1
          N2=KY2(I-1)-1
         ELSE
          N1=KY1(I+1)+1
          N2=KY2(I+1)-1
         ENDIF
        DO 60 J=N1,N2
         UV=(U(I,J)**2+V(I,J)**2)**0.5
C
C       COMPUTATION OF U(I,J)
C
          TM1=(Z(I+1,J)+Z(I+1,J+1))/2.-(Z(I,J)+Z(I,J+1))/2.
          TM2=(ZP(I+1,J)+ZP(I+1,J+1))/2.-(ZP(I,J)+ZP(I,J+1))/2.
         TERM1=-DELT*G*(ALFA*HUV(I,J)*TM1
     &         +(1-ALFA)*HUVP(I,J)*TM2)/ZDELX(I)
         TM1=(Z(I+1,J)+Z(I+1,J+1)+Z(I,J)+Z(I,J+1))/4.
         TM2=(ZP(I+1,J)+ZP(I+1,J+1)+ZP(I,J)+ZP(I,J+1))/4.
         TERM2=UD(I,J)*(TM1-TM2)/HUVP(I,J)
          TM1=UV*U(I,J)
          FC=116.*CN**2/HR(I,J)**(1/3.)
          TRM1=FC*TM1/8./HUVP(I,J)**2
```

```
          TRM2=3.*0.000001*ABS(W(I,J))*W(I,J)
          TERM3=-DELT*(TRM1-TRM2)
         U(I,J)=UD(I,J)+TERM1+TERM2+TERM3
C
C        COMPUTAION OF V(I,J)
C
          TM1=(Z(I,J+1)+Z(I+1,J+1))/2.-(Z(I,J)+Z(I+1,J))/2.
          TM2=(ZP(I,J+1)+ZP(I+1,J+1))/2.-(ZP(I,J)+ZP(I+1,J))/2.
          TERM1=-DELT*G*(ALFA*HUV(I,J)*TM1
     &          +(1.-ALFA)*HUVP(I,J)*TM2)/ZDELY(J)
          TM1=(Z(I+1,J)+Z(I+1,J+1)+Z(I,J)+Z(I,J+1))/4.
          TM2=(ZP(I+1,J)+ZP(I+1,J+1)+ZP(I,J)+ZP(I,J+1))/4.
          TERM2=VD(I,J)*(TM1-TM2)/HUVP(I,J)
          TM1=UV*V(I,J)
          TERM3=-DELT*FC*TM1/8./HUVP(I,J)**2
         V(I,J)=VD(I,J)+TERM1+TERM2+TERM3
60       CONTINUE
C
C---IMPOSE CLOSE BOUNDARY CONDITIONS
C
         DO 70 J=1,KJO
          U(1,J)=0.
          V(1,J)=0.
          U(KIO,J)=0.
          V(KIO,J)=0.
70       CONTINUE
C
         DO 80 I=2,KIO-1
          IF(I.LT.25) THEN
           N1=KY1(I-1)
           N2=KY2(I-1)
          ELSE
           N1=KY1(I+1)
           N2=KY2(I+1)
          ENDIF
         DO 85 J=1,N1
          U(I,J)=0.
          V(I,J)=0.
85       CONTINUE
         DO 90 J=N2,KJO
          U(I,J)=0.
          V(I,J)=0.
90       CONTINUE
80       CONTINUE
C
C--- ADVANCE VELOCITIES: PU(I,J) AND PV(I,J)
C
         DO 100 I=1,KIO
         DO 100 J=KY1(I),KY2(I)
          PU(I,J)=U(I,J)/HUV(I,J)
          PV(I,J)=V(I,J)/HUV(I,J)
100      CONTINUE
C
         RETURN
```

```
      END
C
C
C************** SUBROUTINE DEPT **************
C** ADVANCE WATER DEPTH AND WATER SURFACE ELEVATION
C
      SUBROUTINE DEPT
      INCLUDE 'DIM.FOR'
C
      DO 30 I=2,KIO
      DO 30 J=2,KJO
       Z(I,J)=Z(I,J)+DZ(I,J)
       H(I,J)=Z(I,J)-ZB(I,J)
30    CONTINUE
      DO 29 I=1,KIO
       Z(I,1)=Z(I,2)
       H(I,1)=H(I,2)
29    CONTINUE
      DO 31 J=1,KJO
       Z(1,J)=Z(2,J)
       H(1,J)=H(2,J)
31    CONTINUE
C
      DO 32 I=1,KIO
      DO 32 J=1,KJO
       IF(H(I,J).LT.HMIN) THEN
       H(I,J)=HMIN
       Z(I,J)=ZB(I,J)+HMIN
       ENDIF
32    CONTINUE
C
      DO 35 J=1,KJO
      DO 35 I=1,KIO-1
       HX(I,J)=GAMA(I,J)*H(I+1,J)+(1-GAMA(I,J))*H(I,J)
35    CONTINUE
      DO 36 J=1,KJO
       HX(KIO,J)=H(KIO,J)
36    CONTINUE
C
      DO 40 J=1,KJO-1
      DO 40 I=1,KIO
       HUV(I,J)=(HX(I,J)+HX(I,J+1))/2.
40    CONTINUE
      DO 45 I=1,KIO
       HUV(I,KJO)=H(I,KJO)
45    CONTINUE
C
      RETURN
      END


C*************** SUBROUTINE FMOVB **************
C** PERFORM THE COMPUTAION OF THE TIDAL FLODDING AND DRYING
C
```

```
      SUBROUTINE FMOVB
C
      INCLUDE 'DIM.FOR'
      COMMON/W/BAT(150,150)
C
C--- CHECK TO SEE IF THE FLOW COMPUTED FROM THE NAVIER-STOKES
C    EQUATIONS IS CONSISTENT WITH THE FLOW CONTROLLED BY THE
C    BOTTOM FRICTION
C
       AFF=2.
      DO 5 J=2,KJO-1
       N1=KX1(J)
       N2=KX2(J)
      DO 5 I=N1,N2
C
C     EXAMINE U(I,J)
C
      SGR=AFF*ABS(G*(Z(I+1,J)-Z(I,J))/DELX(I))
       FC=116.*CN**2/HUV(I,J)**(1/3.)
      BFR=FC/8.*U(I,J)**2/HUV(I,J)**3.
C
      IF(H(I,J).LT.50.) THEN
       IF(BFR.GT.SGR) THEN
         TM=G*ABS(Z(I+1,J)-Z(I,J))/DELX(I)*HUV(I,J)**3.*8/FC
        U(I,J)=AFF*SQRT(TM)*U(I,J)/ABS(U(I,J))
       ELSE
        U(I,J)=U(I,J)
       ENDIF
      ENDIF
C
C     EXAMINE V(I,J)
C
      SGR1=AFF*ABS(G*(Z(I,J+1)-Z(I,J))/DELY(J))
      FC=116.*CN**2/HUV(I,J)**(1/3.)
      BFR1=FC/8.*V(I,J)**2/HUV(I,J)**3.
C
      IF(H(I,J).LT.50.) THEN
       IF(BFR1.GT.SGR1) THEN
         TM=G*ABS(Z(I,J+1)-Z(I,J))/DELY(J)*HUV(I,J)**3.*8/FC
        V(I,J)=AFF*SQRT(TM)*V(I,J)/ABS(V(I,J))
       ELSE
        V(I,J)=V(I,J)
       ENDIF
      ENDIF
C
5     CONTINUE
C
C--- UPDATE THE WEIGHTING COEFFICIENTS DUE TO THE MOTION
C    OF THE BOUNDARY
C
      DO 10 J=2,KJO-1
       IF(J.LT.25) THEN
        N1=KX1(J-1)+1
        N2=KX2(J-1)
```

```
          ELSE
           N1=KX1(J)+1
           N2=KX2(J)
          ENDIF
         DO 10 I=N1,N2
          IF(U(I,J).LT.0) THEN
C
C         FLOODING
C
            TM=3.*(Z(I,J)-Z(I+1,J))
           TMM=(TM-H(I+1,J)+H(I,J))
           IF(TMM.EQ.0) THEN
            BAT(I,J)=0.5
           ELSE
            BAT(I,J)=(TM-H(I+1,J))/TMM
           ENDIF
           IF (BAT(I,J).LT.0.) THEN
            BAT(I,J)=0.5
           ELSE
            IF(BAT(I,J).GT.1.) THEN
             BAT(I,J)=1.
            ELSE
             BAT(I,J)=BAT(I,J)
            ENDIF
           ENDIF
           GAMA(I,J)=BAT(I,J)
C
          ELSE
C
C        DRYING
C
            IF(Z(I,J).LT.ZB(I+1,J)) THEN
             BAT(I,J)=1.
            ELSE
               TM=3.*(Z(I+1,J)-Z(I,J))
              TMM=(TM-H(I,J)+H(I+1,J))
             IF(TMM.EQ.0) THEN
              BAT(I,J)=0.5
             ELSE
              BAT(I,J)=(TM-H(I,J))/TMM
             ENDIF
             IF (BAT(I,J).LT.0.) THEN
              BAT(I,J)=0.2
             ELSE
              IF(BAT(I,J).GT.1.) THEN
               BAT(I,J)=1.
              ELSE
               BAT(I,J)=BAT(I,J)
              ENDIF
             ENDIF
            ENDIF
            GAMA(I,J)=1-BAT(I,J)
C
          ENDIF
```

```
10      CONTINUE
C
        RETURN
        END


C**************** SUBROUTINE UVDEP *************
C** PERFORM CALCULATION OF HX(I,J), HY(I,J), HYDRAULIC RADIUS, AND
C   WATER DEPTH AT (U,V) GRID
C
        SUBROUTINE UVDEP
        INCLUDE 'DIM.FOR'
C
C--- EVALUATE HX(I,J) AND HY(I,J)
C
        DO 10 I=1,KIO-1
        DO 10 J=1,KJO
         HX(I,J)=GAMA(I,J)*H(I+1,J)+(1-GAMA(I,J))*H(I,J)
10      CONTINUE
        DO 20 J=1,KJO
         HX(KIO,J)=H(KIO,J)
20      CONTINUE
C
        DO 30 I=1,KIO
        DO 30 J=1,KJO-1
         HY(I,J)=BATAY(I,J)*H(I,J+1)+(1-BATAY(I,J))*H(I,J)
30      CONTINUE
        DO 40 I=1,KIO
         HY(I,KJO)=H(I,KJO)
40      CONTINUE
C
C--- EVALUATE WATER DEPTH AT (U,V) GRID
C
        DO 50 I=1,KIO
        DO 50 J=1,KJO-1
         HUV(I,J)=(HX(I,J)+HX(I,J+1))/2.
50      CONTINUE
        DO 60 I=1,KIO
         HUV(I,KJO)=H(I,KJO)
60      CONTINUE
C
C--- EVALUATE THE HYDRAULIC RADIUS
C
        DO 70 I=1,KIO
        DO 70 J=1,KJO
         HR(I,J)=HUV(I,J)
70      CONTINUE
C
        RETURN
        END


C**************** SUBROUTINE T2DOT *************
C*** THIS SUBROUTINE IS USED TO OUTPUT THE RESULTS
```

```
C
      SUBROUTINE T2DOT
      INCLUDE 'DIM.FOR'
C
C--- OUTPUT THE VELOCITIES
C
      IF(NT.EQ.200) THEN
       OPEN(UNIT=1,NAME='VELOCITY.DAT',STATUS='NEW')
       DO 10 I=1,KIO
       DO 10 J=1,KJO
        TM1=U(I,J)/HUV(I,J)
        TM2=V(I,J)/HUV(I,J)
        WRITE(1,*) NT,TM1,TM2
10     CONTINUE
       CLOSE(1)
      ENDIF
C
C--- OUTPUT THE LOCATION OF THE BOUNDARY
C
      IF(NT.EQ.100) THEN
       OPEN(UNIT=1,NAME='BOUNDARY.DAT',STATUS='NEW')
40     DO 20 J=1,KJO
       DO 30 I=1,KIO
        IF(H(I+1,J).GT.HMIN) THEN
         WRITE(1,*)J,I
         GOTO 40
        ENDIF
30     CONTINUE
20     CONTINUE
       CLOSE(1)
       ENDIF
C
       RETURN
       END
```

```
C       CUGA----------MATRIX INVERSE
C
        SUBROUTINE CUGA(X,A,B,C,D,M)
C
C       PERFORM TRIDIAGONAL MATRIX INVERSE
C
        DIMENSION X(150),A(150),B(150),C(150),D(150)
        DIMENSION P(200), R(200),T(200),W(200)
C
        MN=M-1
        P(1)=A(2)
        DO 10 K=1,MN
10      R(K)=B(K+1)
        DO 20 K=2,M
        T(K)=C(K+1)/P(K-1)
        P(K)=A(K+1)-T(K)*R(K-1)
20      CONTINUE
        W(1)=D(2)
        DO 30 K=2,M
        W(K)=D(K+1)-T(K)*W(K-1)
30      CONTINUE
        X(M+1)= W(M)/P(M)
        DO 40 KK=1,MN
        K=MN-KK+1
        X(K+1)=(W(K)-R(K)*X(K+2))/P(K)
40      CONTINUE
        RETURN
        END


C**************DIM.FOR*****************
C** DIMENSION FILE
C
        COMMON/A/Z(70,70),H(70,70),HR(70,70),HUV(70,70),HX(70,70)
     &           ,HY(70,70),ZB(70,70)
        COMMON/B/U(70,70),V(70,70),UD(70,70),VD(70,70),UAD(70,70)
     &           ,VAD(70,70)
        COMMON/C/DZ(70,70),DZ1(70,70),DZ2(70,70)
        COMMON/D/ZDELX(70),ZDELY(70),DELX(70),DELY(70)
        COMMON/E/GAMA(70,70),ALFA
        COMMON/F/G,CN,DELT,DC,F,WOMAGA,ATA,HMIN
        COMMON/G/KIO,KJO,NT,KX1(70),KX2(70),KY1(70),KY2(70)
        COMMON/I/YM1(70,70),YM2(70,70)
        COMMON/J/PU(70,70),PV(70,70)
        COMMON/K/AX(70,70),BX(70,70),CX(70,70),DDX(70,70)
        COMMON/L/AY(70,70),BY(70,70),CY(70,70),DDY(70,70)
        COMMON/M/W(70,70)
```

BIBLIOGRAPHY

Beckman, F. S., "The Solution of Linear Equations by Conjugate Gradient Method," Mathematical Methods for Digital Computers, Vol. 1, 1959, pp 62-72.

Benque, J. P., J. A. Cunge, J. F. Hauguel, and F. M. A. Holly, "New Method for Tidal Current Circulation," Journal of the Waterway Port Coastal and Ocean Division, Vol. 108, No. WW3, ASCE, 1982a, pp 396-417.

Benque, J.-P., A. Hauguel, and P.-L. Viollet, Engineering Applications of Computational Hydraulics, Vol. 1, Pitman Advanced Publishing Program, Boston, 1982b.

Benque, J.-P., A. Hauguel, and P.-L. Viollet, Engineering Applications of Computational Hydraulics, Vol. 2, Pitman Advanced Publishing Program, Boston, 1982c.

Garrett, J. R., "Review of Drag Coefficients Over Ocean and Continents," Monthly Weather Review, Vol. 7, 1977, pp 915-929.

Carrier, G. F. and H. P. Greenspan, "Water Waves of Finite Amplitude on a Sloping Beach," Journal of Fluid Mechanics, Vol. 4, 1958, pp 97-109.

Gopalakrishhan, T. C. and C. C. Tung, "Numerical Analysis of a Moving Boundary Problem in Coastal Hydrodynamics," International Journal for Numerical Methods in Fluids, Vol. 3, 1983, pp 179-200.

Gray, W. G. (Editor), Physics-Based Modeling of Lakes, Reservoirs, and Impoundments, ASCE, New York, 1986.

Hirt, C. W. and B. D. Nichols, "Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries," Journal of Computational Physics, Vol. 39, 1981, pp 201-225.

Isaacson, E. and H. B. Keller, Analysis of Numerical Methods, John Wiley & Sons Inc., New York, 1966.

Khosla, P. K. and S. G. Rubin, "A Conjugate Gradient Iterative Method," Computers and Fluids, Vol. 9, 1981, pp 109-121.

Leendertse, J. J., "A Water-Quality Simulation Model for Well- Mixed Estuaries and Coastal Seas: Volume I," Principales of Computation, RM- 6230-RC, The Rand Corporation, Santa Monica, Calif., Feb., 1970.

Lynch, D. R. and W. G. Gray, "Finite Element Simulation of Flow in Deforming Regions," Journal of Computational Physics, Vol. 36, No. 2, 1980, pp 135-153.

Lynch, D. R., "Moving Boundary Numerical Surge Model," Journal of the Waterway Port Coastal and Ocean Division, Vol. 106, No. WW3, ASCE, 1980, pp 425-428.

Marchuk, G. I., Methods of Numerical Mathematics, Springer-Verlag, New York, 1975.

Melkonian, S., Nonlinear Waves on Thin Films and Related Phenomena, Ph. D. Thesis, McGill University, Canada, 1987.

Murty, T. S., Storm Surges, Canadian Government Publishing Centre, Canada, 1984.

Peyret, R. and T. D. Taylor, Computational Methods for Fluid Flow, Springer-Verlag, New York, 1986.

Rahman, M., "Analytical Solutions for Tidal Propagation in a Rectangular Basin," Advances in Water Resources, Vol. 6, 1983, pp 44-53.

Reid, R. O. and B. R. Bodine, "Numerical Model for Storm Surges in Galveston Bay," Journal of the Waterways and Harbors Division, Vol. 94, No. WW1, ASCE, 1968, pp 33-57.

Schmalz, R. A., "A Numerical Investigation of Hurricane Induced Water Level Fluctuations in Lake Okeechobee: Report 1, Forcecasting and Design," US Army Engineer District, Jacksonville, 1986.

Sheng, Y. P., "Mathematical Modeling of Three-Dimensional Coastal Currents and Sediment Dispersion: Model Development and Application," A. R. A. P. Report CERC-83-2, Princeton, 1983.

Smith, G. D., Numerical Solution of Partial Differential Equations, Oxford University Press, London, 1969.

Weare, T. J., " Errors Arising from Irregular boundaries in ADI Solutions of the Shallow-water Equations," International Journal for Numerical Methods in Engineering, Vol. 14, 1979, pp. 921-931.

Yanenko, N. N., The Method of Fractional Steps, Springer-Verlag, New York, 1971.

Yeh, G.-T. and F.-K. Chou, "Moving Boundary Numerical Surge Model," Journal of the Waterway Port Coastal and Ocean Division, Vol. 105, No. WW3, ASCE, 1979, pp 247-263.

Yeh, G.-T. and F.-K. Chou, "Moving Boundary Numerical Surge Model," Journal of the Waterway Port Coastal and Ocean Division, Vol. 107, No. WW1, ASCE, 1981, pp 34-36.

Yeh, G.-T. and F.-F. Yeh, "Generalized Model for Storm Surges," Coastal Engineering, Proceedings of the Fifteenth Coastal Engineering Conference, Honolulu, Hawaii, 1976, Vol. 1, pp 921-940.

Yih, C. S., "Stability of Liquid Flow Down an Inclined Plane," Phys. Fluids 6, 1963, pp 321-331.

Zelt, J. A., "Tsunamis: the Response of Harbors with Sloping Boundary to Long Wave Excitation," Report No. KH-R-47, W. M. Keck Laboratory of Hydraulics and Water Resources, Division of Engineering and Applied Science, California Institute of Technology, 1986.