User's Manual for University of Texas Bays and Estuaries Simulator (UTBEST)

S. Chippada, C. N. Dawson, and M. F. Wheeler

Center for Subsurface Modeling (CSM)
Texas Institute for Computational & Applied Mathematics (TICAM)
University of Texas
Austin, TX 78712

Submitted to the Texas Water Development Board in partial fulfillment of
the requirements for Contract No. 97-483-204

September, 1997

User's Manual for University of Texas Bays and Estuaries Simulator (UTBEST)

S. Chippada, C. N. Dawson, and M. F. Wheeler

Center for Subsurface Modeling (CSM)
Texas Institute for Computational & Applied Mathematics (TICAM)
University of Texas
Austin, TX 78712

Submitted to the Texas Water Development Board in partial fulfillment of
the requirements for Contract No. 97-483-204

September, 1997

User's Manual for University of Texas Bays and Estuaries Simulator (UTBEST)

S. Chippada, C. N. Dawson, and M. F. Wheeler

Center for Subsurface Modeling (CSM)
Texas Institute for Computational & Applied Mathematics (TICAM)
University of Texas
Austin, TX 78712.

September 15, 1997

Abstract

UTBEST is a bays and estuaries simulator developed at the Center for Subsurface Modeling (CSM), University of Texas at Austin. This numerical simulator models long wavelength phenomena such as tidal waves in shallow coastal systems. It solves the shallow water equations numerically using a Godunov-type finite volume method and unstructured triangular meshes. The physical variables such as fluid depth and fluid velocities/discharges are represented as piece-wise constants or linears on each triangle. The Riemann shock-tube problem at each cell edge is solved in an approximate manner using Roe's technique. First-order and second-order Runge-Kutta time discretizations have been implemented. The lateral eddy diffusivity terms are incorporated using mixed/hybrid finite element method and lowest-order Raviart-Thomas spaces. The numerical algorithm can handle both supercritical and subcritical flows. In addition, it is also capable of handling wetting and drying of coastal boundaries. The program is easy to use and numerical experimentation has shown the method to be robust, accurate and efficient.

1 INTRODUCTION

UTBEST is a bays and estuaries simulator developed at the Center for Subsurface Modeling (CSM), University of Texas at Austin. It is based on a Godunov-type finite volume algorithm and solves the local Riemann shock-tube problem at the cell interface in an approximate manner using Roe's technique. It is based on unstructured triangular meshes and represents physical variables such as fluid depth and fluid velocity by piecewise constants or piecewise linears. The numerical algorithm implements lateral diffusion terms in a time-implicit manner using the mixed/hybrid finite element method and lowest-order Raviart-Thomas spaces. The algorithm can handle all types of flows, i.e., both supercritical and subcritical flows. Further, it is also capable of handling wetting and drying of coastal boundaries.

The primary purpose of this document is to serve as user's manual for UTBEST version 1.0. The mathematical model is described in §2 and the numerical algorithm is described in §3. Inputs to the numerical code are described in §4. The various outputs that can be obtained from the code are expained in §5 and some concluding remarks are given in §6.

2 MATHEMATICAL MODEL

UTBEST solves numerically the 2-D shallow water equations (SWE), which themselves are derived by vertically averaging the 3-D incompressible Navier-Stokes equations along with assumptions of vertically uniform velocity profiles and hydrostatic pressure distribution. The SWE can be used to study important physical phenomena such as tidal surges, tidal fluctuations, tsunami waves and contaminant and salinity transport. The primary variables are the fluid depth H, and the fluid velocities $\mathbf{u} = (u, v)$. The water-air interface (free surface) deflection from the mean sea level (MSL) is denoted by ξ and is related to total fluid depth as: $\xi = H - h_b$, where h_b is the bathymetric depth. The definition of ξ, h_b and H are shown in Fig.1. The fluid discharge is defined as velocity times the fluid depth, and is written as: $U = uH$, $V = vH$. Chosing ξ, and $\mathbf{U} = (U, V)$ as our primary dependent variables, the SWE are given by:

$$\frac{\partial \xi}{\partial t} + \nabla \cdot \mathbf{U} = 0, \quad \text{and} \tag{1}$$

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \left(\frac{\mathbf{UU}}{H} \right) + gH\nabla\xi + \tau_{bf}\mathbf{U} + f_c \mathbf{k} \times \mathbf{U} = E_h \triangle \mathbf{U} + gH\nabla(\alpha\eta - p_a) + \tau_{\mathbf{ws}}. \tag{2}$$

Eq.1 represents conservation of mass, and Eq.2 represents conservation of momentum. The definitions of the various variables that appear in the above equations are given in Appendix A. As is obvious from the above equations, the shallow water equations are a set of conservation equations that are highly nonlinear and subject to various types of external forcings. Eventhough they are derived from incompressible Navier-Stokes equations, the shallow water equations themselves exhibit compressible behaviour. In analogy with compressible gas dynamics, the fluid depth H plays the role of density and the pressure can be written as: $g\left(H^2 - h_b^2\right)/2$. Gravity wave plays the role of sound wave and its speed is approximately: $\sqrt{gH}$.

Land, open sea and river are some of the commonly encountered boundaries. In the case of a land boundary we apply a no normal flux condition in the following manner:

$$\nabla\xi \cdot \nu = 0, \quad \mathbf{U} \cdot \nu = 0, \quad \text{and} \quad \nabla(\mathbf{U} \cdot \tau) \cdot \nu = 0. \tag{3}$$

On the open sea, due to lack of better data, we apply a dirichlet condition on the elevation (ξ). The velocity boundary conditions are adhoc and the hope is that error introduced by inaccurate boundary conditions does not propagate too far into the interior of the domain. The open sea boundary conditions are:

$$\xi = \hat{\xi}, \quad \text{and} \quad \nabla(\mathbf{U}) \cdot \nu = 0. \tag{4}$$

A river boundary is nothing but an inflow boundary with dirichlet boundary conditions for all variables written as:

$$\xi = \hat{\xi}, \quad \mathbf{U} \cdot \nu = \hat{\mathbf{U}}_\nu, \quad \text{and} \quad \mathbf{U} \cdot \tau = \hat{\mathbf{U}}_\tau. \tag{5}$$

In addition to the above physical conditions, some times an artificial radiation-type boundary condition may have to be specified. These boundaries arise from truncation of physical domains. Radiation-type boundaries are also referred to as absorbing or non-reflecting boundaries.

In addition to boundary conditions appropriate initial conditions need to be specified. Shallow water systems are dynamic systems and to obtain a complete state of the system is very difficult. It is common practice to start with the system completely at rest, i.e. with zero velocities and free surface deflections,

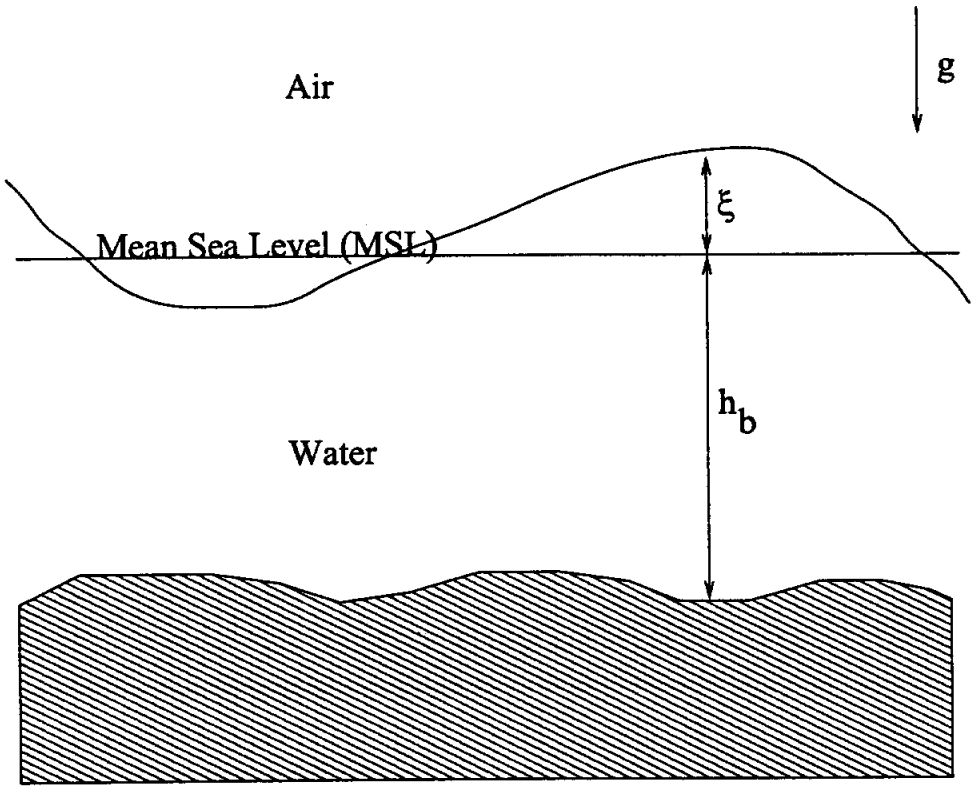

Figure 1: Definition of elevation, bathymetry and fluid depth.

everywhere in the domain and to introduce the boundary conditions and body forces in a gradual manner through the use of a ramp function. This is commonly referred to as a cold start as against hot start, where the initial velocity and elevation field throughout the domain is specified.

3 NUMERICAL ALGORITHM

The numerical algorithm used in UTBEST is a Godunov-type finite volume method and unstructured triangular grids. On each triangle, the physical variables (ξ, U, V) are approximated as piecewise polynomials that are not necessarily continuous across the cell edges. Both, piecewise constants and piecewise linears have been implemented in the current version of UTBEST. At the cell interface, the flow field is discontinuous and the numerical flux is computed by solving Riemann shock-tube problem. The Godunov-type finite volume method is described next. This is followed by a description of higher-order extensions and incorporation of viscous terms through mixed/hybrid finite element method.

3.1 Godunov-type finite volume method

Dropping viscous terms, which are second-order in space, reduces the shallow water equations to a first-order hyperbolic system of conservation laws that can be written in the following form:

$$\frac{\partial \mathbf{c}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} = \mathbf{h} \tag{6}$$

where,

$$\mathbf{c} = \begin{pmatrix} \xi \\ U \\ V \end{pmatrix}; \quad \mathbf{f} = \begin{pmatrix} U \\ \frac{U^2}{H} - \frac{1}{2}g\left(H^2 - h_b^2\right) \\ \frac{UV}{H} \end{pmatrix}; \quad \mathbf{g} = \begin{pmatrix} V \\ \frac{UV}{H} \\ \frac{V^2}{H} - \frac{1}{2}g\left(H^2 - h_b^2\right) \end{pmatrix}; \tag{7}$$

and,

$$\mathbf{h} = \begin{pmatrix} 0 \\ -\tau_{bf}U + f_c V + g\xi\frac{\partial h_b}{\partial x} + gH\left(\frac{\partial}{\partial x}(\alpha\eta) - \frac{\partial}{\partial x}p_a\right) + \tau_{wsx} \\ -\tau_{bf}V - f_c U + g\xi\frac{\partial h_b}{\partial y} + gH\left(\frac{\partial}{\partial y}(\alpha\eta) - \frac{\partial}{\partial y}p_a\right) + \tau_{wsy} \end{pmatrix}. \tag{8}$$

The primary variables (ξ, U, V) are discretized as piece-wise constants within the area enclosed by the linear triangles, as shown in Fig.2. These element averages are updated each time step through the fluxes crossing the element edges and the body forces acting on the volume of the element. The integral formulation for the hyperbolic system (Eq.6) is given by:

$$\frac{\partial}{\partial t}\int_{\Omega_e} \mathbf{c}\, d\Omega_e + \int_{\Gamma_e} \mathbf{f_n}\, d\Gamma_e = \int_{\Omega_e} \mathbf{h}\, d\Omega_e \tag{9}$$

where, $\mathbf{f_n} = \mathbf{f}n_x + \mathbf{g}n_y$ is the normal flux crossing the boundary of the control volume. In the above equations, Ω_e and Γ_e are, respectively, the area and the boundary of the element, $\mathbf{n} = (n_x, n_y)$ is the outward pointing unit normal vector of the boundary Γ_e.

For triangular control volumes such as that shown in Fig.2, the time discretized equations would be:

$$\frac{\mathbf{c_e}^{n+1} - \mathbf{c_e}^{n}}{\Delta t}\Omega_e + \sum_{i=1}^{i=3}\left(\mathbf{f}^n_{n_i}\Gamma_i\right) = \mathbf{h}^n\Omega_e. \tag{10}$$

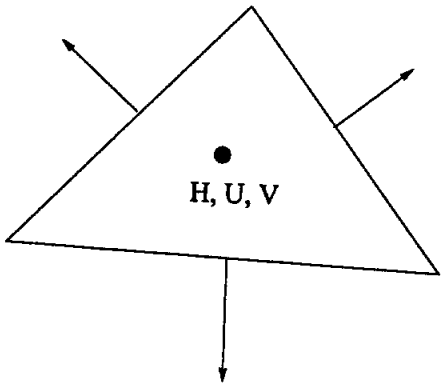

Figure 2: Typical control volume

The superscript represents the time level, and the element averages of the primary variables are updated explicitly each time step, once the normal fluxes at the element edges are known. In Godunov method, the variables are approximated as averages within the element volumes, and the advective flux at the cell interface is computed by solving the Riemann shock-tube problem. Thus, the fluxes computed at the element edges are the exact analytical fluxes we would have had if there were two constant states to either side of the edge. The discontinuities propagate with the right velocities and without any spurious oscillations. All variables are locally conserved and the classical Godunov approach gives us a stable, monotonic numerical algorithm (LeVeque 1992). It is possible to solve the Riemann problem at the cell interface exactly. However, this often requires the solution of a set of nonlinear algebraic equations which can be time consuming. Moreover, the higher order accuracy obtained through the exact calculation of the fluxes is lost due to the cell averaging done at the end of the flux calculation, since the primary variables are only represented as averages within cell volumes. Thus, a number of approximate Riemann solvers have been constructed to solve the Riemann problem in an efficient manner and Roe's approximation is one of them. In UTBEST, the Riemann problem at the cell interface is solved in an approximate manner using Roe's technique.

Before getting into Roe's approximation, it is worthwhile to look at the quasi-linear form and the eigen system that arises in the context of shallow water equations. The quasi-linear form of the hyperbolic system (Eq.6) is given by:

$$\frac{\partial \mathbf{c}}{\partial t} + \mathbf{A}\frac{\partial \mathbf{c}}{\partial x} + \mathbf{B}\frac{\partial \mathbf{c}}{\partial y} = \mathbf{h} \tag{11}$$

where $\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{c}}$, and $\mathbf{B} = \frac{\partial \mathbf{g}}{\partial \mathbf{c}}$. This is a nonlinear problem with $\mathbf{A}$ and $\mathbf{B}$ being functions of $\mathbf{c}$. Note that, the quantity of interest in the integral formulation (Eq.9) is the normal flux ($\mathbf{f}_n$) crossing the cell interface. The normal Jacobian matrix $\mathbf{A}_n$ is found to be:

$$\mathbf{A}_n = \begin{bmatrix} 0 & n_x & n_y \\ n_x\left(a^2 - u^2\right) - n_y uv & 2n_x u + n_y v & n_y u \\ n_y\left(a^2 - v^2\right) - n_x uv & n_x v & 2n_y v + n_x u \end{bmatrix} \tag{12}$$

where, $a^2 = gH$. The eigenvalues and the corresponding eigenvectors for the normal Jacobian ($\mathbf{A}_n$) are

given by:

$$\lambda_1 = u_n - \sqrt{gH} \;\; ; \;\; \mathbf{r}_1 = \left\{ \begin{array}{c} 1 \\ u - \sqrt{gH} n_x \\ v - \sqrt{gH} n_y \end{array} \right\} \tag{13}$$

$$\lambda_2 = u_n \;\; ; \;\; \mathbf{r}_2 = \left\{ \begin{array}{c} 0 \\ -n_y \\ n_x \end{array} \right\} \tag{14}$$

$$\lambda_3 = u_n + \sqrt{gH} \;\; ; \;\; \mathbf{r}_3 = \left\{ \begin{array}{c} 1 \\ u + \sqrt{gH} n_x \\ v + \sqrt{gH} n_y \end{array} \right\} \tag{15}$$

In the case of a pure 1-D problem, we would have only the eigenvalues: $(u_n - \sqrt{gH},\, u_n + \sqrt{gH})$. An additional eigenvalue equal to u_n arises in the case of 2-D. Eigenvalues λ_1 and λ_3 result in either a compression wave (shock) or a rarefaction wave depending on the left and right states. All primary variables are discontinuous across the shock and continuous across rarefaction waves. The eigenvalue λ_2 is linearly degenerate and gives rise to a contact discontinuity, where, only the tangential velocity is discontinuous and the normal velocity and fluid depth are continuous.

In Roe's approximation, the nonlinear problem is linearized at the cell interface (Roe 1981, LeVeque 1992). At the cell interface we have a discontinuity with state $\mathbf{c}_L$ on the left side and state $\mathbf{c}_R$ on the right side. In Roe's approximation, Eq.11 is linearized as follows:

$$\frac{\partial \mathbf{c}}{\partial t} + \hat{\mathbf{A}}\left(\mathbf{c}_L, \mathbf{c}_R\right) \frac{\partial \mathbf{c}}{\partial x} + \hat{\mathbf{B}}\left(\mathbf{c}_L, \mathbf{c}_R\right) \frac{\partial \mathbf{c}}{\partial y} = \mathbf{h} \tag{16}$$

where, $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ are linearized forms of $\mathbf{A}$ and $\mathbf{B}$, respectively. Eventhough there is more than one way to compute the linearized matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$, it is desired that the linearized matrices satisfy certain properties. Important among them is that the Rankine-Hugoniot jump conditions be satisfied across the disconituity, i.e.,

$$[\mathbf{f}_n] = \hat{\mathbf{A}}_n \, [\mathbf{c}] \tag{17}$$

where, [ ] represents the jump across the interface. Going through the procedure described by Roe (Roe 1981, LeVeque 1992), we find that if the following type of averaging is done, then the resulting linearized matrices preserve the jump conditions at the interface.

$$\hat{\mathbf{c}} = \frac{H_L^{-1/2}\mathbf{c}_L + H_R^{-1/2}\mathbf{c}_R}{H_L^{-1/2} + H_R^{-1/2}} \;\; ; \;\; \hat{\mathbf{A}} = \hat{\mathbf{A}}\left(\hat{\mathbf{c}}\right) \;\; ; \;\; \hat{\mathbf{B}} = \hat{\mathbf{B}}\left(\hat{\mathbf{c}}\right) \tag{18}$$

The Roe-linearized normal Jacobian matrix $\hat{\mathbf{A}}_n$ is found to be:

$$\hat{\mathbf{A}}_n = \left[\begin{array}{ccc} 0 & n_x & n_y \\ n_x\left(\hat{a}^2 - \hat{u}^2\right) - n_y \hat{u}\hat{v} & 2n_x\hat{u} + n_y\hat{v} & n_y\hat{u} \\ n_y\left(\hat{a}^2 - \hat{v}^2\right) - n_x \hat{u}\hat{v} & n_x\hat{v} & 2n_y\hat{v} + n_x\hat{u} \end{array} \right] \tag{19}$$

where, $\hat{a}^2 = g\left(H_L + H_R\right)/2$. The Roe-linearized normal Jacobian matrix $(\hat{\mathbf{A}}_n)$ is same as the original nonlinear Jacobian matrix $(\mathbf{A}_n)$, except that the Roe-linearized velocities and fluid depth are used in place of the actual values. Thus, the eigenvalues and the eigenvectors have the same form as that given by Eqs.13-15, with the Roe-linearized variables used in place of the actual values.

Once the cell interface problem has been linearized, it is a simple matter to compute the normal flux $\hat{\mathbf{f}}_n$. It is well known from the theory of linear hyperbolic systems that an initial state that is discontinuous across an interface evolves in time in such a manner that the discontinuities propagate with the characteristic speeds given by the eigenvalues. The evolution of the cell interface problem in time is shown schematically in Fig.3. The following relations give the relation between the various states across the discontinuities:

$$\mathbf{c}_{m1} = \mathbf{c}_L + \alpha_1 \hat{\mathbf{r}}_1, \tag{20}$$

$$\mathbf{c}_{m2} = \mathbf{c}_{m1} + \alpha_2 \hat{\mathbf{r}}_2, \tag{21}$$

$$\mathbf{c}_R = \mathbf{c}_{m2} + \alpha_3 \hat{\mathbf{r}}_3. \tag{22}$$

Knowing $\mathbf{c}_L$ and $\mathbf{c}_R$, the weighting factors α_1, α_2 and α_3 can be computed and are found to be:

$$\begin{array}{rcl} \alpha_1 & = & \frac{a+\hat{u}_n}{2a}(H_R - H_L) - \frac{n_x}{2a}(U_R - U_L) - \frac{n_y}{2a}(V_R - V_L), \\ \alpha_2 & = & (\hat{u}n_y - \hat{v}n_x)(H_R - H_L) - n_y(U_R - U_L) + n_x(V_R - V_L), \quad \text{and} \\ \alpha_3 & = & \frac{a-\hat{u}_n}{2a}(H_R - H_L) + \frac{n_x}{2a}(U_R - U_L) + \frac{n_y}{2a}(V_R - V_L). \end{array} \tag{23}$$

Knowing the weighting factors the Roe-Riemann flux at the interface is given by:

$$\hat{\mathbf{f}}_n(\mathbf{c}_L, \mathbf{c}_R) = \mathbf{f}_n(\mathbf{c}_L) + \sum_{p=1}^{p=3} \hat{\lambda}_p^- \alpha_p \hat{\mathbf{r}}_p \tag{24}$$

where, $\hat{\lambda}_p^- = min(\hat{\lambda}_p, 0)$. Once the numerical fluxes are computed, the state at the next time level is easily determined in an explicit fashion through Eq.10. Since this is a completely explicit procedure, the time step is limited by the CFL condition stated as:

$$\Delta t \leq inf \frac{h_e}{\lambda_{max}} \tag{25}$$

where, the infimum is taken over all numerical elements in the domain, h_e is a suitable measure of the size of the cell, and λ_{max} is the maximum eigenvalue for the cell.

3.2 Higher-order extension

The basic Godunov method, as described above is conservative and monotone, but is only first-order accurate in space and time. In this section, we discuss the extension of the basic Godunov-type finite volume method described above to higher-order spatial accuracy, without losing the conservative and monotone property of the numerical scheme.

First-order upwind methods are monotone and resolve discontinuities without producing any oscillations. However, in smooth portions of the flow, the first-order accuracy is not sufficient. Second-order methods based on central-difference type approximations give good accuracy in smooth portions of the flow, but produce oscillations in the vicinity of discontinuities, and thus are not monotone. Several high resolution methods have been constructed that give higher-order accuracy in smooth portions of the flow but reduce to a first-order method near discontinuities. In this manner, higher-order (that is better than first-order) monotone methods can be constructed. UTBEST uses a slope-limiter approach to construct higher-order total variation bounded (TVB) schemes.

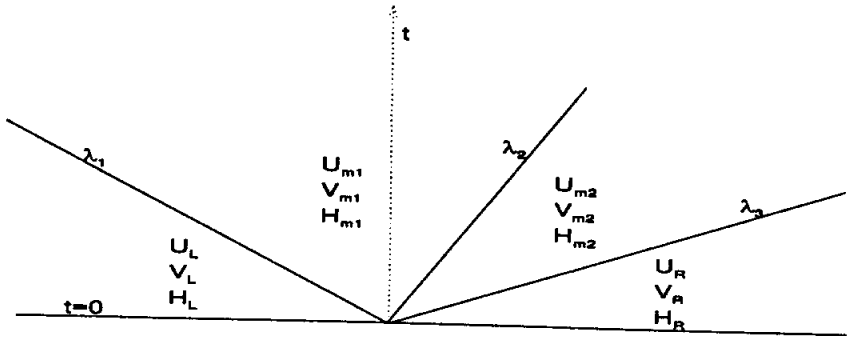

Figure 3: Evolution of the linearized cell interface problem

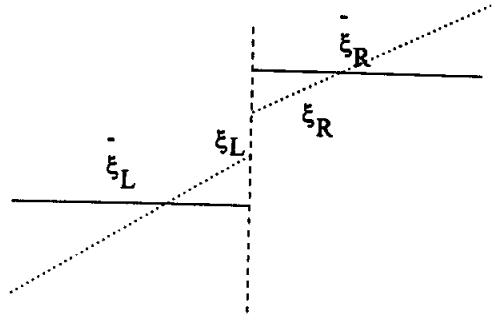

Figure 4: The left and right states at a cell interface.

In this method, the basic numerical algorithm remains the same as described in the previous section. It aims to increase the accuracy of the flux calculations at the cell interface by getting a better estimate of the left state and the right state as shown in Fig.4. Linear approximations within each cell gives a better approximation of the left and right states at the cell interface resulting in a more accurate flux calculation, which ultimately increases the accuracy of the numerical scheme. The numerical algorithm is same as that described in the previous section. The only additional calculation is the reconstruction of slopes from cell averages and the use of this reconstructed flow field in computing the numerical flux at the cell interface.

Construction of piecewise linears for a scalar variable (say ξ) is described next. The slope computation can be thought to consist of two steps, namely, the reconstruction step, and the limiting step. The reconstruction is as follows. Within each cell, ξ is expressed as a piecewise linear function of the following form:

$$\xi_i^R = \bar{\xi}_i + \left(\frac{\partial \xi}{\partial x}\right)_i (x - \bar{x}_i) + \left(\frac{\partial \xi}{\partial y}\right)_i (y - \bar{y}_i) \tag{26}$$

where, ξ_i^R is the reconstructed field, $\bar{\xi}_i$ is the cell-average, and $(\bar{x}_i, \bar{y}_i)$ is the centroid of the cell. The subscript i denotes the cell number, and the superscript R stands for the reconstructed field. The piecewise constant field is constructed in such a way that when averaged over the cell we get back the original cell-average,

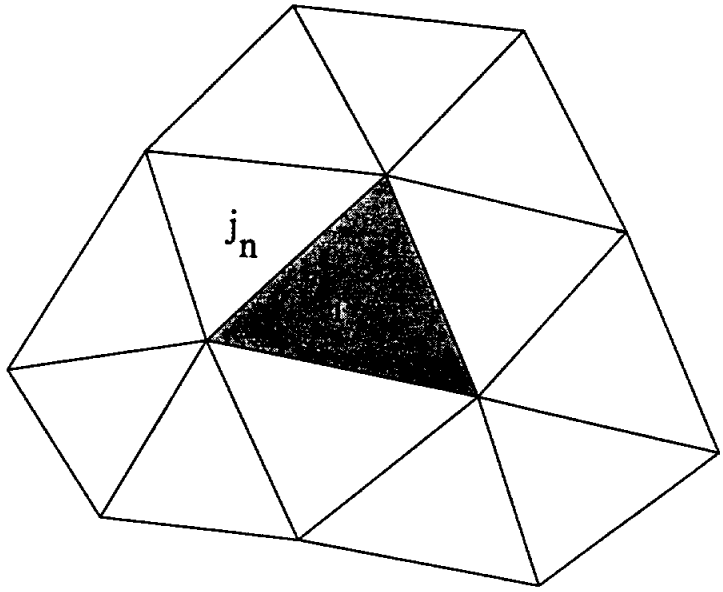

Figure 5: The patch used for computing slopes

thus maintaining the conservative property of the overall numerical scheme. The slopes are computed using cell-averages in the neighboring cells (Fig.5) with the following constraints:

$$\int_{A_{j_n}} \xi_i^R \, dA_{j_n} = \bar{\xi}_{j_n} A_{j_n}, \quad n = 1, \ldots, N \tag{27}$$

where j_n are neighbouring elements (Fig.5) and N is the total number of neighboring elements of element i. Since typically we have more constraints than unknowns, the slopes are computed through least-squares minimization.

This reconstruction step is followed by a limiting step. The reconstructed field is checked to see that no new extrema are created in the domain. In the case of piecewise linears, the extrema occur at the cell vertices. In each triangle, the reconstructed field is checked to verify that no new extrema are created at its vertices, and if necessary the slopes are adjusted by iterating over the vertices. The reconstruction step and the limiting step together give a second-order method that is conservative and total variation bounded (TVB).

A straight forward extension of the above procedure to each of the primary variables in the case of a system of conservation laws might not always work. After extensive testing the following procedure was found. Piecewise linear reconstruction for elevations ξ can be done as described above. The slopes for U and V are calculated as follows:

$$\frac{\partial U}{\partial x} = u\frac{\partial H}{\partial x}; \quad \frac{\partial U}{\partial y} = u\frac{\partial H}{\partial y}; \quad \frac{\partial V}{\partial x} = v\frac{\partial H}{\partial x}; \quad \frac{\partial V}{\partial y} = v\frac{\partial H}{\partial y}. \tag{28}$$

The slopes for fluid depth H are obtained by adding bathymetric depth slopes to elevation slopes. The bathymetric depth itself, being physical data, is approximated as continuous piecewise linears. This type of slope calculations for U and V imply that the velocities are still piecewise constant within each element.

This also implies that the velocity field is both divergence free and vorticity free. It is believed that it is this important property that stabilizes the higher-order reconstruction.

The slope reconstruction procedure as described above results in a Godunov-type finite volume procedure that is spatially second-order accurate in smooth portions of the flow, and first order accurate near discontinuities. To make the scheme truly second-order, the temporal accuracy needs to be increased as well and this is done using the two step Runge-Kutta procedure proposed by Shu and Osher (1989). Let $L(\mathbf{c})$ represent the linear operator consisting of the higher-order fluxes obtained after slope reconstruction and the body forces. The SWE system can then be represented as:

$$\frac{\partial \mathbf{c}}{\partial t} = L(\mathbf{c}). \tag{29}$$

The two step Runge-Kutta procedure can then be written as:
Predictor step:

$$\hat{\mathbf{c}}^{n+1} = \mathbf{c}^n + \Delta t L(\mathbf{c}^n) \tag{30}$$

Corrector step:

$$\mathbf{c}^{n+1} = \mathbf{c}^n + \frac{\Delta t}{2}\left[L(\hat{\mathbf{c}}^{n+1}) + L(\mathbf{c}^n)\right] \tag{31}$$

3.3 Treatment of Viscous Terms

In the discussion so far, the second-order viscous terms have been dropped which reduces the shallow water equations to a system of first-order hyperbolic conservation laws. In most physical situations bottom friction dominates lateral diffusion and dispersion and this is a reasonable assumption. In certain flows, especially those involving recirculations, lateral diffusion is an important process and needs to be modeled. In UTBEST, the viscous terms are handled through a mixed/hybrid finite element method and lowest-order Raviart-Thomas spaces.

The viscous terms are decoupled from advection terms through operator splitting. Within each time step, there are two smaller steps. In the first step all terms except the diffuion terms are handled in a manner described previously using Godunov finite volume method and an intermediate discharge field $\tilde{\mathbf{U}}^{n+1} = \left(\tilde{U}^{n+1}, \tilde{V}^{n+1}\right)$ is computed. To this intermediate discharge field, the viscous terms are added in a time-implicit manner and the final discharge field $\mathbf{U}^{n+1} = \left(U^{n+1}, V^{n+1}\right)$ is computed. The diffusion step can be written as:

$$\mathbf{U}^{n+1} = \tilde{\mathbf{U}}^{n+1} + \Delta t E_h \Delta \mathbf{U}^{n+1}. \tag{32}$$

Note that, the eddy diffusion coefficient E_h could have spatial variations. In the present implementation of UTBEST, only a spatially constant E_h is considered. It is expected that spatially varying E_h would be implemented in future releases of UTBEST. Since viscous terms are incorporated in time-implicit manner, this step poses no additional limitations on the size of time step Δt.

The central idea of a mixed finite element method is to replace a second-order system with a first-order system through the use of additional intermediate variables. The viscous step given by Eq.32 can be rewritten as a first-order system in the following manner:

$$\begin{aligned} \mathbf{U}^{n+1} &= \tilde{\mathbf{U}}^{n+1} - \Delta t \nabla \cdot \mathbf{z}^{n+1}, \text{ and} \\ \mathbf{z}^{n+1} &= -E_h \nabla \mathbf{U}^{n+1}. \end{aligned} \tag{33}$$

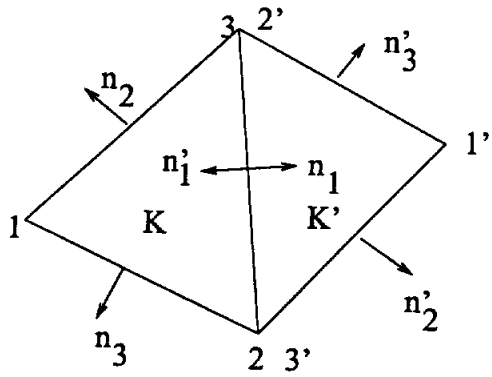

Figure 6: Lowest-order Raviart Thomas Spaces on triangles

In the above system, $\mathbf{z}$ is the diffusion flux of discharge field $\mathbf{U}$. Note that $\mathbf{z}$ is a second-order tensor. In lowest-order Raviart-Thomas spaces, $\mathbf{U}$ is approximated as piecewise constant within each triangle, whereas the diffusion flux $\mathbf{z}$ is approximated as piecewise linears within each triangle with only the normal components of the diffusion fluxes being continuous across the cell edges. Let K and K' be triangles that share an edge as shown in Fig.6. A_K and $A_{K'}$ are the areas of the triangles K and K' respectively. ∂K and $\partial K'$ are the boundaries of triangles K and K'. 1, 2 and 3 are vertices of element K, with 1', 2' and 3' being the vertices of element K'. The edges are also numbered, with edge 1 being the edge opposite to node 1, and so on. The lengths of the edges are denoted by l_i and the outward pointing normals are denoted by $\mathbf{n}_i$, with subscript i taking values 1, 2 and 3 in element K, and values 1', 2' and 3' in element K'. Further, we have $n_1 = -n_1'$. The fluid discharges are represented as piecewise constants in each triangle in the following manner:

$$\mathbf{U}|_K = \mathbf{U}_K. \tag{34}$$

The diffusion flux is represented as piecewise linears within each element in the following manner:

$$\mathbf{z}|_K = \sum_{i=1}^{3} (\mathbf{z}_K \cdot \mathbf{n}_i)\, \mathbf{N}_i. \tag{35}$$

The shape functions $\mathbf{N}_i$ are vector functions and are given the following variation:

$$\mathbf{N}_i = \left\{ \begin{array}{c} \alpha_i + \beta_i x \\ \gamma_i + \beta_i y \end{array} \right\} \tag{36}$$

The parameters α_i, β_i and γ_i can be determined from the 3×3 system:

$$\mathbf{N}_i \cdot \mathbf{n}_j = \delta_{ij}, \quad j = 1, 2, 3. \tag{37}$$

Solving the above 3×3 system, α_i, β_i and γ_i are found to be given by:

$$\begin{array}{rcl} \alpha_i & = & -\frac{x_i l_i}{2A_K} \\ \beta_i & = & \frac{l_i}{2A_K} \\ \gamma_i & = & -\frac{y_i l_i}{2A_K} \end{array} \tag{38}$$

In addition to fluid discharges and diffusion fluxes, a new variable that $\lambda = (\lambda_U, \lambda_V)$ that only lives on the edges of triangle is introduced in the following manner:

$$\lambda|_K = \sum_{i=1}^{3} \lambda_i \mu_i \tag{39}$$

where,

$$\mu_i = \begin{cases} 1 \text{ on edge i of element K} \\ 0 \text{ otherwise.} \end{cases} \tag{40}$$

The λ_i's are Lagrange multipliers which ensure continuity of diffusion fluxes along edge i. This use of Lagrange multipliers is referred to as hybrid finite element method, and thus the numerical method is referred to as mixed/hybrid finite element method. The continuiuty of normal diffusion flux across the cell edges ensures that the numerical scheme is conservative.

The weak form of the viscous step over element K is given by:

$$\left(\mathbf{U}_K^{n+1}, 1\right)_K = \left(\tilde{U}_K^{n+1}\right)_K - \Delta t \left(\nabla \cdot \mathbf{z}, 1\right)_K \tag{41}$$

$$\left(E_h^{-1}\mathbf{z}_K^{n+1}, \mathbf{N}_i\right)_K = \left(\mathbf{U}_K^{n+1}, \nabla \cdot \mathbf{N}_i\right)_K - < \lambda^{n+1}, \mathbf{N}_i \cdot \mathbf{n}_K >_{\partial K} \tag{42}$$

$$< [\mathbf{z}^{n+1} \cdot \mathbf{n}_1], 1 >_{\partial K \cap \partial K'} = 0 \tag{43}$$

$[\mathbf{z}_K^{n+1} \cdot \mathbf{n}_1] = \mathbf{z}_K \cdot \mathbf{n}_1 - \mathbf{z}_{K'} \cdot \mathbf{n}_1 = 0$ is the jump in diffusion flux across edge 1. The unknowns are $\mathbf{U}^{n+1}$, $\mathbf{z}$ and λ which need to be solved from the above system. However, $\mathbf{U}$ and $\mathbf{z}$ can be eliminated from the above system resulting is a global system that needs to be solved for λ. Knowing λ, $\mathbf{U}$ can be easily computed from the above system.

4 DESCRIPTION OF INPUT

The input consists of run parameters, grid data and edge data. Each of this data is read from separate disk files and is decribed in the next 3 subsections.

4.1 RUN PARAMETERS

Run parameters are read from unit 15. The default input file is set to *fort.15*. The input parameters that are read from this file as well as the FORTRAN-77 format statements used are given below:

READ(15,'(A32)') RUNDES : a 32 character alpha-numeric description of the model run.

READ(15,'(A24)') RUNID : a 24 character alpha-numeric description of the model run identification.

READ(15,*) IHOT : selection of either hot start or cold start. If IHOT equals 0, cold start is chosen and the initial conditions for velocities and elevation are set to zero everywhere in the domain. If IHOT is not 0, then hot start is chosen. If hotstart is chosen, the initial values of the free surface deflection and velocities are read from a file. If IHOT equals 67, then the initial state is read from *fort.67* and if IHOT equals 68, then the initial state is read *fort.68*. This comes in especially handy, if for some reason the simulation crashes during a run. Then using IHOT, one can restart from where the program crash occurred instead of starting from the beginning again. It should be mentioned that the hot-start features of the code have not been tested. Further, the program does n't store a copy of its simulation state on the disk file. So, it would be safe to use only cold-start in the current version of UTBEST. The hot-start version will be implemented and tested in later versions of UTBEST.

READ(15,*) ICS : to chose between cartesian or spherical co-ordinate system. If ICS equals 1, cartesian co-ordinate system is chosen and the code expects to read x and y co-ordinates from the grid file. If ICS equals 2, a spherical co-ordinate system is chosen, and the code expects to read latitudes (SFEA) and longitudes (SLAM) in degrees from the grid file. If spherical co-ordinate system is chosen, the code uses a Carte-Parallelogram Projection (CPP) centered about a reference point (SLAM0,SFEA0), to project the spherical grid onto a plane surface. The latitudes and longitudes are calculated into equivalent x and y coordinates on this plane. This mapping results in certain corrections, which need to be accounted for in the calculation of normals, areas and gradients. It is expected that these corrections are small. In the present implementation of UTBEST, the CPP projection is implemented but the resulting corrections are not implemeneted in the calculation of areas and gradients.

READ(15,*) NOLIBF : to chose linear or non-linear bottom friction. Linear bottom friction is chosen if NOLIBF equals 0, and a non-linear bottom friction coefficient is chosen if it equals 1. The linear friction coefficient is named TAU, and is read in later. The nonlinear bottom friction is implemented using Chezy's equations and the friction coefficient is named CF and read in later.

READ(15,*) NWP : spatial variation of the bottom friction coefficient and the lateral diffusion coefficient. If NWP equals 0, then they are spatially invariant. If NWP equals 1, then they are spatially varying and their nodal values need to be read from a separate file. The spatially varying bottom friction and diffusion coefficients have not been tested and are not recommended at this time.

READ(15,*) NCOR : spatial variation of the Coriolis parameter. If NCOR equals 0, then a spatially constant Coriolis parameter is used. This is fairly accurate for small fluid domains. If NCOR equals 1, then the Coriolis parameter is spatially varying and is computed as $2\Omega \sin\phi$, where Ω is the angular speed of earth's rotation and ϕ is the latitude. A spatially varying Coriolis acceleration is appropriate for large fluid domains. Note that if a spatially varying Coriolis acceleration is chosen, it is preferable to chose CPP coordinate system.

READ(15,*) NTIP : to select Newtonian tide potential, which arises from variations in the value of the gravitational acceleration due to changing positions of moon, sun and planets. If NTIP equal 0, then this feature is turned off. If NTIP equals 1, then the Newtonian tide potential is turned on. Normally, the tidal forces are imposed as boundary conditions at the open ocean boundaries. This is accurate enough in small fluid domains. In very large fluid domains however, the Newtonian tide potential phenomena is important and cannot be neglected. The number of tidal potential frequencies is read in later. It is recommended that CPP coordinates be used when Newtonian tide potential is turned on.

READ(15,*) NWS : to turn on wind stress. If NWS equals 0, then wind stress is turned off. If NWS equals 1, then it is turned on. The wind stresses themselves are read from a *fort.22* disk file. The wind stress feature has not been completely tested and is not recommended at this time.

READ(15,*) NRAMP : to turn on ramping function. When a cold start is chosen it is desirable to impose the boundary conditions and body forces in a gradual manner. This feature is turned on when NRAMP equals 1 and turned off when it equals 0. The gradual ramping is done using a hyperbolic tangenet function and the ramping time in days is read in later.

READ(15,*) G : is the gravitational acceleration. The value specified for G determines the physical units employed. Moreover, when CPP coordinate system is chosen or when variable Coriolis force or

Newtonian tide potential is chosen, only SI units are allowed.

READ(15,*) NQUAD : number of Gaussian quadrature points to be used in integrations over the triangular elements.

READ(15,*) XI(I), YI(I), W(I) : the quadrature points and weights.

READ(15,*) NDTVAR : variable time step selection parameter. A constant pre-determined time step is taken if NDTVAR equals 0. If NDTVAR equals 1, then the code choses time step automatically based on the CFL criteria.

READ(15,*) DT : time step size, and is specified in seconds. DT is unchanged if NDTVAR equal 0. Otherwise, it is recalculated within the code every time step such that DT equals the maximum possible time step size based on stability criteria.

READ(15,*) STATIM : the starting time for the model run, and is specified in days.

READ(15,*) REFTIM : the reference time in days.

READ(15,*) RNDAY : the total length of the simulation in days.

READ(15,*) IRK : selection of Runge-Kutta time stepping scheme. If IRK equals 1, first-order Runge-Kutta which is nothing but explicit Euler forward scheme is chosen. If IRK equals 2, second-order 2-step Runge-Kutta scheme is chosen. In this version, these are the only schemes that have been implemented. The diffusion terms are added through simple first-order operating.

READ(15,*) ISLOPE : selection of slope reconstruction. If ISLOPE equals 0, then the basic first-order Godunov finite volume method is chosen and all physical variables are approximated as piece-wise constants within each cell. If ISLOPE equals 1, then the physical variables are represented as discontinuous piece-wise linears, and the slopes themselves are reconstructed from cell averages through least-squares and slope limiting, so that we obtain second-order spatial accuracy without losing monotonicity in the vicinity of discontinuities. It is recommended that if slope reconstruction is chosen that 2-step Runge-Kutta time stepping scheme be hosen so that we have second-order accuracy both in space and time.

READ(15,*) ITRANS : stedy-state or transient simulation. If ITRANS equals 0, then a transient simulation is chosen and the code will run for the total length of RNDAY days. If ITRANS equals 1, then steady-state simulation is chosen and the code runs untill the convergence criteria (CONVCR) is met or untill the total length of simulation equals RNDAY days.

READ(15,*) CONVCR : convergence criteria used in stopping steady-state simulations.

READ(15,*) DRAMP : ramping period in days. At the end of DRAMP days, the ramp function equals 0.96 approximately and approaches 1.0 for times greater than DRAMP.

READ(15,*) H0 : the minumum fluid depth allowed. If fluid depth in a cell falls below H0, then it is reset to H0 and the velocities in that cell are set to zero. This condition is checked every time step.

READ(15,*) SLAM0, SFEA0 : the reference co-ordinates in degrees for CPP projection.

READ(15,*) TAU : the linear bottom friction coefficient.

READ(15,*) CF ; the non-linear (Chezy's) bottom friction coefficient.

READ(15,*) NVISC : selection of lateral eddy viscosity. If NVISC equals 0, the lateral eddy viscosity is neglected. Otherwise, the lateral eddy viscosity is accounted for through mixed/hybrid finite element method.

READ(15,*) ESL : the lateral eddy (kinematic) viscosity with units length square per unit time.

READ(15,*) CORI : the Coriolis parameter. If spatially varying Coriolis is chosen this is not used, and the Corilois parameter is computed internally within the code.

READ(15,*) NTIF : number of tidal potential forcing frequencies. NTIF can be greater than zero only when NTIP equals 1, i.e., only when Newtonian tide potential feature is turned on.

READ(15,'(A5)') TIPOTAG(I) : an alpha-numeric descripton of the tidal constituent.

READ(15,*) TPK(I),AMIGT(I),ETRF(I),FFT(I),FACET(I) : the tidal potential amplitude, frequency, earth tide potential reduction factor, nodal factor and equlibrium argument in degrees.

READ(15,*) NBFR : number of tidal forcing frequencies on the open boundaries.

READ(15,'(A5)') BOUNTAG(I) : an alpha-numeric description of each tidal constituent.

READ(15,*) AMIG(I), FF(I), FACE(I) : forcing frequency, nodal factor and equilibrium argumnet in degrees for tidal forcing on open ocean boundaries.

READ(15,*) NOUTE, TOUTSE, TOUTFE, NSPOOLE : If NOUTE equals 1, interpolated elevations at elevation recording stations are spooled to unit 61 every NSPOOLE time steps starting from time TOUTSE untill time TOUTFE. If NOUTE equals 0, no elevation recordings are not spooled.

READ(15,*) NSTAE : total number of elevation recording stations.

READ(15,*) XEL(I), YEL(I) : the input coordinates of elevation recording stations. This input statement is executed only when cartesian coordinate system is chosen (ICS=1).

READ(15,*) SLEL(I), SFEL(I) : when spherical coordinate system is chosen, the latitudes and longitudes of elevation recording stations are read.

READ(15,*) NOUTV, TOUTSV, TOUTSV, NSPOOLV : If NOUTV equals 1, interpolated velocities at velocity recording stations are spooled to unit 62 every NSPOOLV time steps starting from time TOUTSV untill time TOUTFV. If NOUTV equals 0, no velocity recordings are not spooled.

READ(15,*) NSTAV : total number of velocity recording stations.

READ(15,*) XEV(I), YEV(I) : the input coordinates of velocity recording stations. This input statement is executed only when cartesian coordinate system is chosen (ISC=1).

READ(15,*) SLEV(I), SFEV(I) : when spherical coordinate system is chosen, the latitudes and longitudes of velocity recording stations are read.

READ(15,*) NOUTGE, TOUTSGE, TOUTFGE, NSPOOLGE : if NOUTGE equals 1, global elevation is spooled to unit 63 every NSPOOLGE time steps between times TOUTSGE and TOUTFGE. If NOUTGE equals 0, no global elevation is spooled.

READ(15,*) NOUTGV, TOUTSGV, TOUTFGV, NSPOOLGV : if NOUTGV equals 1, global velocity field is spooled to unit 64 every NSPOOLGV time steps between times TOUTSGV and TOUTFGV. If NOUTGV equals 0, no global velocity field is spooled.

READ(15,*) NHSTAR, NHSINC : if NHSTAR equals 1, the hot start file is generated every NHSINC time steps. If NHSTAR equals 0, no hot start file is generated.

4.2 NUMERICAL GRID

READ(14,'(A24)') AGRID : an alphanumeric description of the grid.

READ(14,*) NE, NP : number of elements and number of nodes.

READ(14,*) JKI, X(JKI), Y(JKI), DP(JKI) : the x and y coordinates and the bathymetric depth. This read statement used if ICS equals 1, i.e., is cartesian coordinate system chosen.

READ(14,*) JKI, SLAM(JKI), SFEA(JKI), DP(JKI) : the longitude and latitude (in degrees) and the bathymetric depth. This read statement used if ICS equals 2, i.e., if spherical coordinates along with CPP projection is chosen.

READ(14,*) JKI, NHY, NM(1,JKI), NM(2,JKI), NM(3,JKI) : the global connectivity table. NM(1,JKI), NM(2,JKI), and NM(3,JKI) are the three vertices of element JKI and are ordered in counter-clockwise direction. NHY is the number of vertices for element JKI. Since the mesh composes of 3 node triangular elements everywhere, NHY should be set to 3. No other types of elements are supported in this release of UTBEST.

4.3 EDGE RELATED DATA STRUCTURES

READ(17,*) NEDGES : number of edges.

READ(17,*)J,NEDNO(1,J),NEDNO(2,J),NEDEL(1,J),NEDEL(2,J) : the two vertices and the two elements on either side of the edge J. In case of boundary edges an element is present only on the interior side and this is stored in NEDEL(1,J) and NEDEL(2,J) is set to zero. The order order of NEDNO() and NEDEL() should be such a way that the normal is pointing into NEDEL(2,J).

READ(17,*) J,NELED(1,J),NELED(2,J),NELED(3,J) : the three edges of element J. The ordering should in counter-clockwise direction.

READ(17,*) NIEDS : number of interior edges.

READ(17,*) J,NIEDN(J) : the global edge number of interior edge J.

READ(17,*) NLEDS : number of land edges.

READ(17,*) J,NLEDN(J) : the global edge number of land edge J.

READ(17,*) NRAEDS : number of edges with radiation-type boundary conditions.

READ(17,*) J,NRAEDN(J) : global edge number of radiation edge J.

READ(17,*) NRIEDS : number of river edges.

READ(17,*) J, NRIEDN(J) : number of river edges.

READ(17,*) J,NRIEDN(J),ETRI(J),UNRI(J),UTRI(J) : global edge number, the free surface deflection, the normal flow rate and the tangential flow rate of river edge J.

READ(17,*) NSEDS : number of open sea boundary edges.

READ(17,*) J, NSEDN(J) : global edge number of sea edge J.

READ(17,*) EMO(I,J),EFA(I,J) : amplitude and phase (in degrees) of the harmonic forcing function at the open ocean boundaries for frequency I and open ocean bounday edge J.

5 DESCRIPTION OF OUTPUT

The main types of output are the station recordings and the complete global field. The elevation station recordings are spooled to unit 61 and the default file is *fort.61*. The velocity station recordings are spooled to unit 62 with a default file name of *fort.62*. The station values are interpolated within the element in which the station is physically present. In case, the station does n't fall within any element, then the values at the station are obtained by extrapolation from the nearest element.

The global elevation is spooled to unit 63 with a default file name *fort.63*, and the global velocity field is spooled to unit 64 with default file name *fort.64*. The global output comprises of values at the vertices of the triangular elements.

For the purpose of plotting, the global solution is spooled to unit 71 with a default file name *xy.out*. First the initial velocity and elevation field is spooled. Later, global solution is spooled every NSPOOLGE time steps between times TOUTSGE and TOUTFGE. If higher-order Godunov method is chosen, i.e., piecewise linears instead of piecewise constants, slopes of elevations and velocities are spooled to unit 72 with default file name *slope.out*. This slope output is between times TOUTSGE and TOUTFGE and is spooled every NSPOOLGE time steps. In addition, we often need to know the maximum and minimum wave heights that occurred in a time interval. The maximum and minimum wave heights between times TOUTSGE and TOUTFGE are computed and to unit 73 with default file name *wave.out*.

6 CONCLUDING REMARKS

UTBEST has been used to simulate a wide variety of flows. It has been used in studying supercritical channel flows encountering change in cross-section area resulting in the formation of hydraulic jumps and rarefaction waves (Chippada, Dawson, Martinez and Wheeler; 1996).

UTBSET has also been used to simulate coastal flow problems. Among the problems studied were tidal flows in the vicinity of Bahamas islands, Galveston Bay, Gulf of Mexico and the entire east coast of the United States. Comaprisons were made with ADCIRC, a finite element simulator based on wave formulation. The results obtained using UTBEST were of comparable accuracy as that obtained using ADCIRC (Chippada, Dawson, Martinez and Wheeler; 1990).

UTBEST can be improved in several ways. From modeling point-of-view, there is a need for incorporating transport of scalar species such as thermal, salinity and contaminants. Also needed is the three-dimensional modeling capability, and incorporation of turbulence models. From programming point-of-view there are certain things that need to be improved. Some of the features such as hot start, spatially varying viscosity, wind stress and atmospheric pressure gradients, and spherical coordinate systems need to be incorporated and properly tested. Also, the user interface may have to be improved.

7 ACKNOWLEDGEMENTS

National Science Foundation.

References

[1] S. Chippada, C. N. Dawson, M. L. Martinez, and M. F. Wheeler, "A Godunov-type finite volume method for system of shallow water equations," TICAM Report 96-57, Texas Institute for Computational and Applied Mathematics, University of Texas, Austin, TX 78712, (1996).

[2] LeVeque, R. J., *Numerical Methods for Conservation Laws*, Birkhauser Verlag, (1992).

[3] Roe, P. L., "Approximate Riemann solvers, parameter vectors, and difference schemes," *J. Comput. Phys.*, **43**, pp.357-372, (1981).

[4] Shu, C. -W., and Osher, S. J., "Efficient implementation of essentially nonoscillatory shock-capturing schemes, II," *J. Comput. Phys.*, **83**, pp.32, (1989).

A Nomenclature

E_h Lateral eddy diffuivity coefficient $\left[L^2T^{-1}\right]$

f_c Coriolis acceleration $= 2\Omega \sin\phi$, where Ω is the angular speed of earth's rotation about its axis and ϕ is the latitude $\left[T^{-1}\right]$

$\mathbf{g}$ acceleration due to gravity $\left[LT^{-2}\right]$

$\mathbf{H}$ total water column $[L]$

h_b bathymetric depth $[L]$

$\mathbf{k}$ local unit normal vector pointing upwards aligned with the gravitational vector

p_a atmospheric pressure as water column $[L]$

$\mathbf{t}$ time $[T]$

$\mathbf{u}$ (u, v), velocity vector $\left[LT^{-1}\right]$

$\mathbf{U}$ (U, V), discharge vector $\left[L^2T^{-1}\right]$

u velocity in x-direction $\left[LT^{-1}\right]$

U uH, fluid dischrage in x-direction $\left[L^2T^{-1}\right]$

V VH, fluid dischrage in y-direction $\left[L^2T^{-1}\right]$

v velocity in y-direction $\left[LT^{-1}\right]$

$\mathbf{x}$ (x, y), the position vector $[L]$

$\mathbf{x}$ horizontal co-ordinate $[L]$

$\mathbf{y}$ horizontal co-ordinate $[L]$

α Effective earth elasticity factor (≈ 0.69)

η Newtonian equlibrium tide potential $[L]$

ν local normal vector

ξ Deflection of water-air interface from mean sea level $[L]$

τ local tangential vector

τ_{bf} bottom friction coefficient $\left[T^{-1}\right]$

$\tau_{\mathbf{ws}}$ wind stress $\left[L^2T^{-2}\right]$