THAYER SCHOOL OF ENGINEERING
DARTMOUTH COLLEGE

NUMERICAL METHODS LABORATORY

THREE-DIMENSIONAL DIAGNOSTIC MODEL FOR BAROCLINIC, WIND-DRIVEN AND TIDAL CIRCULATION IN SHALLOW SEAS

FUNDY4 USERS' MANUAL

Daniel R. Lynch

October 15, 1990

FUNDY4 USERS' MANUAL

ABSTRACT

A 3-D diagnostic model for continental shelf circulation studies is described. The model solves the linearized shallow water equations, forced by tidal or other barotropic boundary conditions, wind, and/or density gradient, using linear Finite Elements. Solutions are obtained in the frequency domain; the limit of zero frequency represents the steady state. The model is written in ANSI FORTRAN 77.

The overall organization of the FUNDY4 system is presented. Detailed specifications are provided for required data files and user-written FORTRAN subroutines. A comprehensive example is given. The model theory and numerical method are described in an appendix.

OVERVIEW

FUNDY4 is a FORTRAN 77 implementation of a finite element solution of the 3-D shallow water equations, as described in Lynch and Werner (1987) and Lynch et al (1990) (herein, LW87 and LWGL90). The model is limited to the linearized equations, with externally specified density field. Solution is obtained in the frequency domain for the complex amplitudes of the fluid velocity and sea surface elevation. The model is forced by tidal or other barotropic boundary conditions, wind, and/or fixed baroclinic pressure gradient, all acting at a single frequency (including zero) and specified by the user. Eddy viscosity closure is used in the vertical, with a linearized partial-slip condition enforced at the bottom. The spatial distribution of viscosity and bottom stress coefficient is arbitrary, and at the discretion of the user. The primary use of FUNDY4 is for preliminary tidal and diagnostic seasonal (steady-state) computations, as a prelude to more complete nonlinear and/or prognostic computations. The governing equations and numerical method are detailed in the Appendix.

The model uses a conventional horizontal grid of linear triangles, which must be provided by the user. A 3-D mesh is automatically constructed within FUNDY4 from these as follows. The horizontal mesh is projected downward to the bottom in perfectly vertical lines, and each line is discretized into the same number of vertical elements. These are then connected horizontally in the identical topology as the original 2-D mesh, thereby filling the volume with 6-node linear elements. Effectively, this creates an (x, y, σ) coordinate system. The detailed local vertical mesh spacing is arbitrary, at the discretion of the user. There is no requirement for uniform vertical meshing; but the above procedure does require that the number of nodes on each vertical line be the same. (See Figure 1).

The software system is illustrated in Figure 2. There are three main programs:

FUNDY4

This is the core program, which performs all finite element assembly and solution operations, according to LW87 and LWGL90. FUNDY4 reads a formatted input file and writes a formatted output file, the latter containing a summary of the input. On execution the program asks for the names of these files. The user need never see the details of FUNDY4.

USER.Subs

FUNDY4 must be linked to four user-built subroutines in order to specify the physical forcing, the vertical structure, and the manner in which results are to be written.

Subroutine BC specifies barotropic boundary conditions. There are three types of barotropic boundary conditions which require data from Subroutine BC:

- fixed elevation
- fixed nonzero normal velocity
- fixed nonzero velocity (used only at mesh corners).

Other types of barotropic boundary conditions (including geostrophic outflow and land

boundaries) are mathematically homogenous and require no data input here. In addition to the BC's, this subroutine also establishes the frequency of the motion.

Subroutine ATMOS specifies the atmospheric forcing, i.e. the combined effects of wind stress and barometric pressure gradient.

Subroutine VERTGRID specifies the number of nodes in the vertical; the vertical node positioning under each horizontal node; and the distribution of eddy viscosity, bottom slip coefficient, and density.

Subroutine OUTPUT is responsible for all output of computed results.

These four subroutines must be provided in ANSI FORTRAN 77 and linked to FUNDY4. Specifications for their construction are given herein. In addition a set of subroutine shells is available with complete data declarations and instructions. A set of example subroutines built from these shells is also available. The overall software strategy allows the user maximum flexibility in the use of formulae and data I/O for specifying the physical forcing, vertical structure, and output quantity and format.

TRIGTOFUNDY4

This program assembles and writes a formatted input file for FUNDY4, from node and element files describing the horizontal mesh. The INP.dat file is a slightly modified version of the WAVETL input file (Lynch 1980). The node and element file formats are compatible with output from the TRIGRID mesh generation package (Henry 1988).

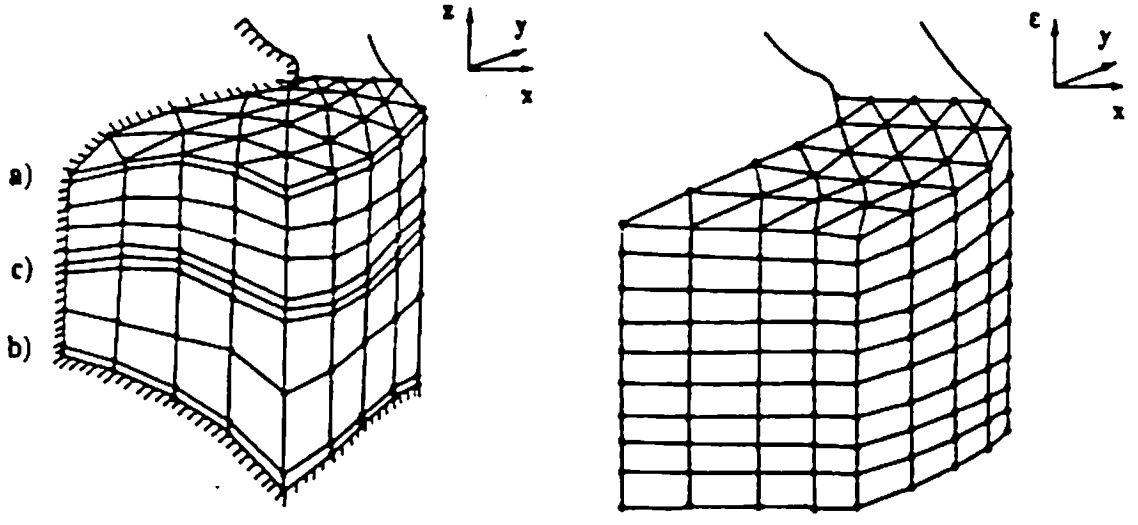

Figure 1. Main features of the layered mesh:
- **element sides perfectly vertical**
- **variable mesh spacing to resolve boundary and internal layers (a,b,c)**
- **uniform mesh spacing in mapped (x, y, ϵ) system**

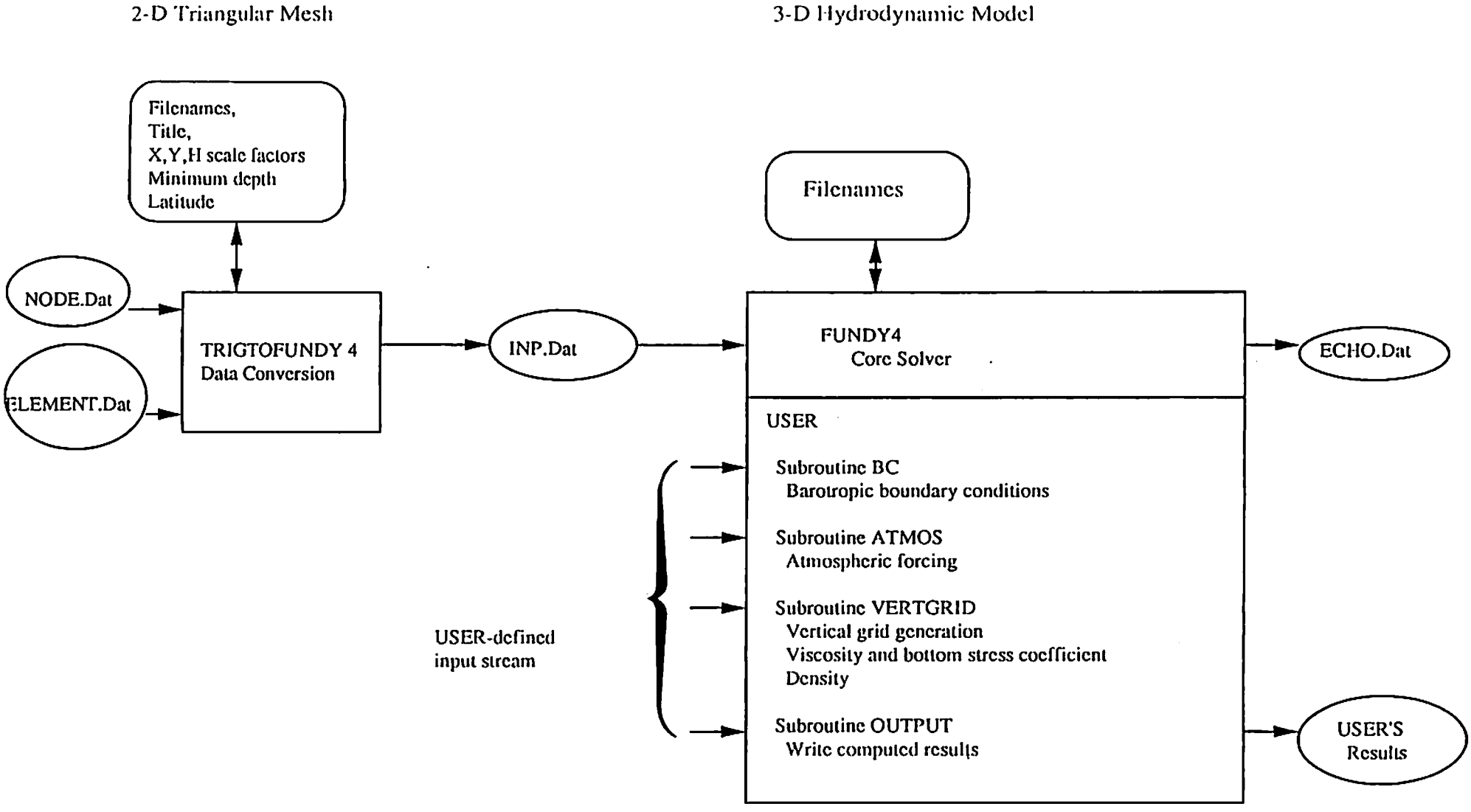

Figure 2. Overview of FUNDY4 and related programs and files.

GENERAL NOTES

1. The triangular horizontal grid node numbering system is retained intact by use of a double-subscript node numbering convention. Node (I,J) indicates horizontal position I, vertical position J. All 3-D arrays use this convention. For example U(I,J) indicates the x-component of velocity at horizontal node I, vertical node J.

2. Recall that there are the same number of vertical nodes everywhere, but their physical and/or relative spacing is arbitrary and at the user's discretion. Therefore equal values of J does not necessarily imply equal values of either z or z/h

3. The vertical coordinate z and the vertical node numbering are positive upward:

$$Bottom : J = 1, z = -h$$

$$Top : J = NNV, z = 0$$

where NNV is the number of vertical nodes and h is the bathymetric depth.

4. FUNDY4 is coded in FORTRAN with single precision COMPLEX data types for the hydrodynamic variables, and single precision REAL data types for the geometric variables. These data types must be respected in user-defined subroutines. The declarations provided in the USER subroutine shells are complete and unambiguous. There are no implicit declarations of data types; the standard FORTRAN 77 default is assumed. Use care relative to mixed-mode computations; and declare any complex local variables which are created.

5. Use SI (MKS) units everywhere. All physical quantities retain their original dimensions.

6. Time variation of the form $A\,exp(i\omega t)$ is assumed for all hydrodynamic variables. ω is the radian frequency; time is in seconds; A is the complex amplitude. A may be reexpressed in terms of the real-valued amplitude a and phaselag in degrees: $A \equiv a\,exp(\frac{-i\pi\phi}{180})$. Equivalently, a is the complex absolute value of A; and $tan(\phi) = -Im(A)/Re(A)$.

USER NOTES

USER NOTES

NODE.dat

NODE.dat is a data file containing a list of the horizontal coordinates (X,Y) and bathymetric depth (H) of each node in the horizontal triangular mesh. Also included is an integer classification NTYPE for each node, indicating the type of boundary condition to be enforced at that node.

Line 1: a single integer NN, the number of nodes in the file. Format is (*).

Line 2: a single integer MAXCON, indicating the maximum number of neighbors for any one node in the mesh. This data is ignored. Format is (*).

Line 3: four real numbers indicating the range of the (X,Y) data following: Xmax, Ymax, Xmin, Ymin. Format is (*). These data are ignored.

Balance of the file: exactly NN lines, each containing the data I, X(I), Y(I), NTYPE(I), H(I) for a single node. Format is (*).

Conventions for NTYPE are as follows:

0: interior point
1: land boundary
2: island boundary
3: nonzero normal velocity
4: geostrophic outflow
5: elevation

6: corner: elevation with land or island
7: corner: geostrophic with land or island
8: corner: nonzero normal velocity with land or island
9: corner: elevation with geostrophic
10: corner: both components of velocity = zero
11: corner: both components of velocity nonzero

ELEMENT.dat

ELEMENT.dat is a data file containing a list of the triangles in the horizontal mesh. For each triangle, there must be one line containing four integers: (L, N1, N2, N3), where L is the element number and N1, N2, N3 are the three node numbers defining that element. The node numbers must be ordered counterclockwise. Format is (*).

TRIGTOFUNDY4

This program will first ask for the names of the NODE and ELEMENT files to be read, and also for the name of the FUNDY4 input file INP.dat to be written. During execution the program will also ask for a "title" i.e. an ASCII string to be written into the the INP.dat file to identify it; latitude to be used in computing the Coriolis parameter; for

scaling factors for X, Y, and bathymetry; and for a minimum bathymetric depth Hmin. These will be written into INP.dat file along with all other nodal and element information needed.

TRIGTOFUNDY4 will also generate core FORTRAN statements for constructing Subroutine BC. These will be written at the top of the INPUT file, along with any diagnostic messages generated en route. (See the description of INPUT.dat below.)

INP.dat

INP.dat is a formatted file conforming to the input requirements of WAVETL (Lynch 1980), with minor modifications and extensions. Users of that model will recognize in the examples the simple modifications required of old INPUT.dat files.

As written by TRIGTOFUNDY4, INPUT.dat contains extra information at the top of the file. First, any diagnostics generated during the processing are written. Following those, incomplete FORTRAN statements appear for every boundary node which needs data in Subroutine BC. For nodes where elevation is specified (NTYPE = 5, 6, or 9) a line of the form

U(I) =

will be written, with I the actual node number. For nodes on a specified nonzero normal velocity boundary (NTYPE = 3), an identical statement will be written:

U(I) =

For nodes on corners where both components of velocity are specified to be nonzero (NTYPE = 11), two such statements are written:

U(I) =

V(I) =

A special case is NTYPE = 8 corners, where the nonzero total velocity is to be specified in terms of the current speed and direction rather than U and V. In this case the file will contain the 3-line sequence

VNORM =

U(I) = NX*VNORM

V(I) = NY*VNORM

for each such node, where (NX,NY) are numbers representing the x- and y-components of the **outward-pointing unit vector, parallel to the land at that corner.** This is the direction of flow required at such nodes.

A complete Subroutine BC can be built by filling out these statements and inserting them at the appropriate points in the BC shell, described below.

Following all this information, the appropriate input information for FUNDY4 is written, beginning with the line "WAVE SI" and ending with the line "XXXX". All information above "WAVE SI" and below "XXXX" will be ignored by FUNDY4; the balance will be processed.

FUNDY4

FUNDY4 is the core solver. On execution it will ask for the name of the INPUT file, and read it. It will also ask for the name of an ECHO file, into which it will write a summary of the input as read. As execution proceeds the user-written subroutines BC, ATMOS, VERTGRID, and OUTPUT will be called. These must be linked to FUNDY4 at run time.

ECHO.dat

This file is written by FUNDY4. It contains a header identifying the version of FUNDY4 which produced it, the title of the INPUT file, and a summary of the input read. The file contents explain themselves.

USER.Subs

USER.Subs is a collection of user-written FORTRAN subroutines which are called by FUNDY4. Complete specifications for these four subroutines is available in their respective shells, which contain the overall strucure, argument list, dimensioning and declarations sufficient for a starting point for their construction. Detailed comments in the shells provide sufficient instructions; these are excerpted below.

Subroutine BC.

This subroutine must be modified by the user to assign barotropic boundary conditions. There are four sections which require modification, beginning at statement labels 3,4,5,and 6 in the BC shell. Each section must begin with the appropriately labelled continue statement, and end with a return statement.

Frequency (label 3). The first section assigns to WW the frequency of the forcing, in radians per second. Time variation of the form $exp(i\omega t)$ is assumed. A return statement must follow this assignment.

Elevaton (label 4). Elevation BC's are assigned in the second section. One statement of the form

U(J) = CVALUE

must appear for each set elevation node, where J is the node number and CVALUE is the complex amplitude of elevation for node J. A return statement must follow the last depth bc assignment.

Normal Velocity (label 5). Nonzero normal velocity BC's are assigned in the third section. One statement of the form

U(J) = CVALUE

must appear for each set nonzero normal velocity node, where J is the node number and CVALUE is the complex amplitude of normal velocity for node J. Positive normal velocity indicates flow **out** of the system. A return statement must follow the last normal velocity

BC assignment. Note that nodes where normal velocity is a priori zero are handled by default in FUNDY4; they are **not** dealt with here.

Velocity (label 6). Nonzero velocity BC's are assigned in the last section. Two statements of the form

U(J) = CXVEL

V(J) = CYVEL

must appear for each set nonzero velocity node, where J is the node number, (CXVEL, CYVEL) are the complex amplitudes of the (X, Y) velocities for node J. A return statement must follow the last velocity BC assignment.

Subroutine ATMOS.

This subroutine must be modified by the user to assign proper values to arrays ATMx and ATMy, the kinematic stress at the top of the water column, in the (X, Y) system:

$$N\frac{\partial \mathrm{V}}{\partial z} = \mathrm{ATM}$$

(See equation 2 in the appendix; in that notation, $\mathbf{ATM} = h\Psi$.) For each node where the atmospheric forcing is nonzero, two statements of the form

ATMx(I) = XVALUE

ATMy(I) = YVALUE

must appear, where I is the node number and (XVALUE, YVALUE) are the complex amplitudes of the forcing. If no value is assigned for a node, the value zero will be assigned to that node by default in the main program. A return statement must follow the last pair of assignments.

Subroutine VERTGRID.

This subroutine must be modified by the user to assign proper values to NNV (the actual number of nodes in the vertical) and to the four arrays

Z(I,J) (vertical node coordinate)

ENZ(I,J) (vertical viscosity, Nz),

AK(I) (bottom slip coefficent, k), and

RHO(I,J) (density anomaly divided by ref. density).

for all values of I (horizontal index) and J (vertical index). The first line of VERTGRID following the dimensioning should assign NNV. (On return to the main program, a check of NNV relative to the main program dimensioning will be made.) Then, for all (I=1,NN) and (J=1,NNV) there must be three assignments of the form

Z(I,J) = ZVALUE

ENZ(I,J) = NVALUE

RHO(I,J) = RHOVALUE

Finally, for all (I=1,NN) there must be an assignment of the form

AK(I) = KVALUE

All quantities are real and have dimensions, except RHO. RHO is complex and dimensionless (it is the density anomoly divided by the reference density). Keep in mind that **Z and J are positive upward:**

Bottom: J=1, Z=-HDOWN
Top: J=NNV, Z=0

A default subroutine UNIGRID is available within FUNDY4 to assign uniform ENZ, AK, and RHO. Its usage is explained in the VERTGRID shell.

Subroutine OUTPUT.

This routine is responsible for writing any and all computed output to the user's specifications. All information is passed to OUTPUT, and the OUTPUT shell contains complete specifications, array typing, etc. A standard routine DUMP is available within FUNDY4; it writes all computed results to a user-specified file, in ASCII format. DUMP takes time and produces a large file; use only when needed. Specifications for DUMP are contained in the OUTPUT shell.

ADDITIONAL SUBPROGRAMS

Several useful subprograms are bundled within the core solver FUNDY4.

Function **PHASELAG**(Z): real-valued phase lag (in radians) of a complex scalar Z.

Function **PHASELAGD**(Z): real-valued phase lag (in degrees) of a complex scalar Z.

Function **CABS**(Z): real-valued amplitude of a complex scalar Z (FORTRAN standard).

Subroutine **ELLIPSE**(U,V,amaj,amin,ainc,g,apl,amn,gpl,gmin): for given complex velocity amplitudes U,V, computes 8 real-valued tidal ellipse parameters defined in Foreman (1978). Adapted from TEMP.FOR (IOS).

Subroutine **CSOLVE**(...): complex unsymmetric banded matrix solver, using LU decomposition (LABLIB standard).

Subroutine **CTHOMAS**(...): complex unsymmetric tridiagonal matrix solver, using the Thomas algorithm (LABLIB standard).

Subroutine **UNIGRID**(...): default uniform vertical structure generator, to be called from VERTGRID. Documentation contained in VERTGRID shell.

Subroutine **DUMP**(...): default ASCII write of all results, to be called from OUTPUT. Documentation contained in OUTPUT shell.

Subroutine **VERTAVG**(F,FBAR,Z,NN,NNV,NNDIM,NNVDIM): computes the complex vertical averages FBAR(I) of complex 3-D array F(I,J), for all I=1,NN. Z(I,J) is the real-valued vertical coordinate of node (I,J). (NN,NNV) are the actual numbers of nodes in the horizontal and vertical. (NNDIM,NNVDIM) are the corresponding dimensioning parameters used in the main program FUNDY4. The vertical averaging is by the trapezoidal

rule, exact for linear elements.

VARIANTS OF FUNDY4

Two variants of FUNDY4 are available. Both differ only in the call to Subroutine VERTGRID.

FUNDY4G asks for complex values of the density **gradient** in Subroutine VERTGRIDG.

FUNDY4R asks for complex values of the right-hand side of the 3-D momentum equation, **R**, (see equation (1), appendix) in Subroutine VERTGRIDR.

The shells for VERTGRIDG and VERTGRIDR contain complete instructions. There are no departures from the basic structure of VERTGRID.

FILES

The directory FUNDY contains the compiled programs FUNDY4.obj, FUNDY4G.obj, FUNDY4R.obj, and the executable TRIGTOFUNDY4.exe. The shells for the four subroutines BC, ATMOS, VERTGRID, and OUTPUT which comprise USER.Subs are contained within USER.Shell, as FORTRAN source code. The shells VERTGRIDG.Shell and VERTGRIDR.Shell are available as separate files in the FUNDY directory.

EXAMPLE APPLICATION

As an example, the application of FUNDY4 to the southwest coast of Vancouver Island is presented. The 2-D triangular mesh used by Foreman and Walters (1990) appears in fig. 3, marked with NTYPE codes appropriate for a tidal simulation (Fig. 3a) and for a steady diagnostic simulation (Fig. 3b). Application files are available in the directory SWVI.

For a tidal simulation, the relevant files are

NODE.dat = NGH81-TRIG-REORD.dat
ELEMENT.dat = TRIANG81-REORD.dat.
INP.dat = SWVI.inp
USER.Subs = SWVI.for
ECHO.dat = OUT.out

An example run stream is recorded in SWVI.out, which resulted from submission of the command file SWVI.com. The file SWVIM2.dat contains the output of this run.

For the diagnostic simulation, the comparable files are

NODE.dat = NGH81-TRIG-REORD-SS.dat
ELEMENT.dat = TRIANG81-REORD.dat.

INPUT.dat = SWVISS.inp
USER.Subs = SWVISS.for
ECHO.dat = OUTSS.out

An example run stream is recorded in SWVISS.out, which resulted from submission of the command file SWVISS.com. The files SWVISSWINDZ.dat and SWVISSWINDV.dat contain the output of this run.

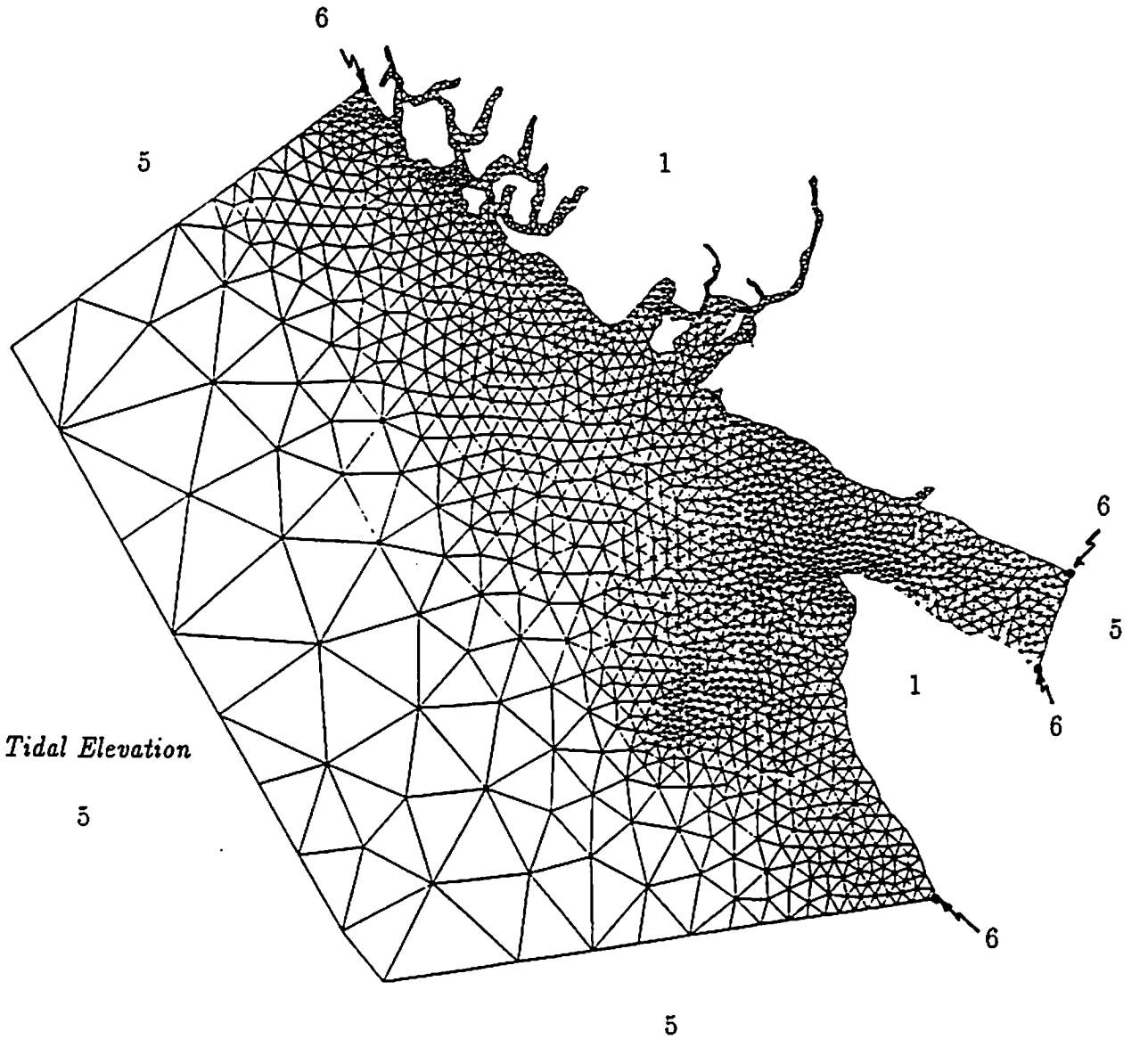

Figure 3a. Boundary condition codes for tidal simulation on the southwest coast of Vancouver Island. The finite element mesh is from Foreman and Walters (1990).

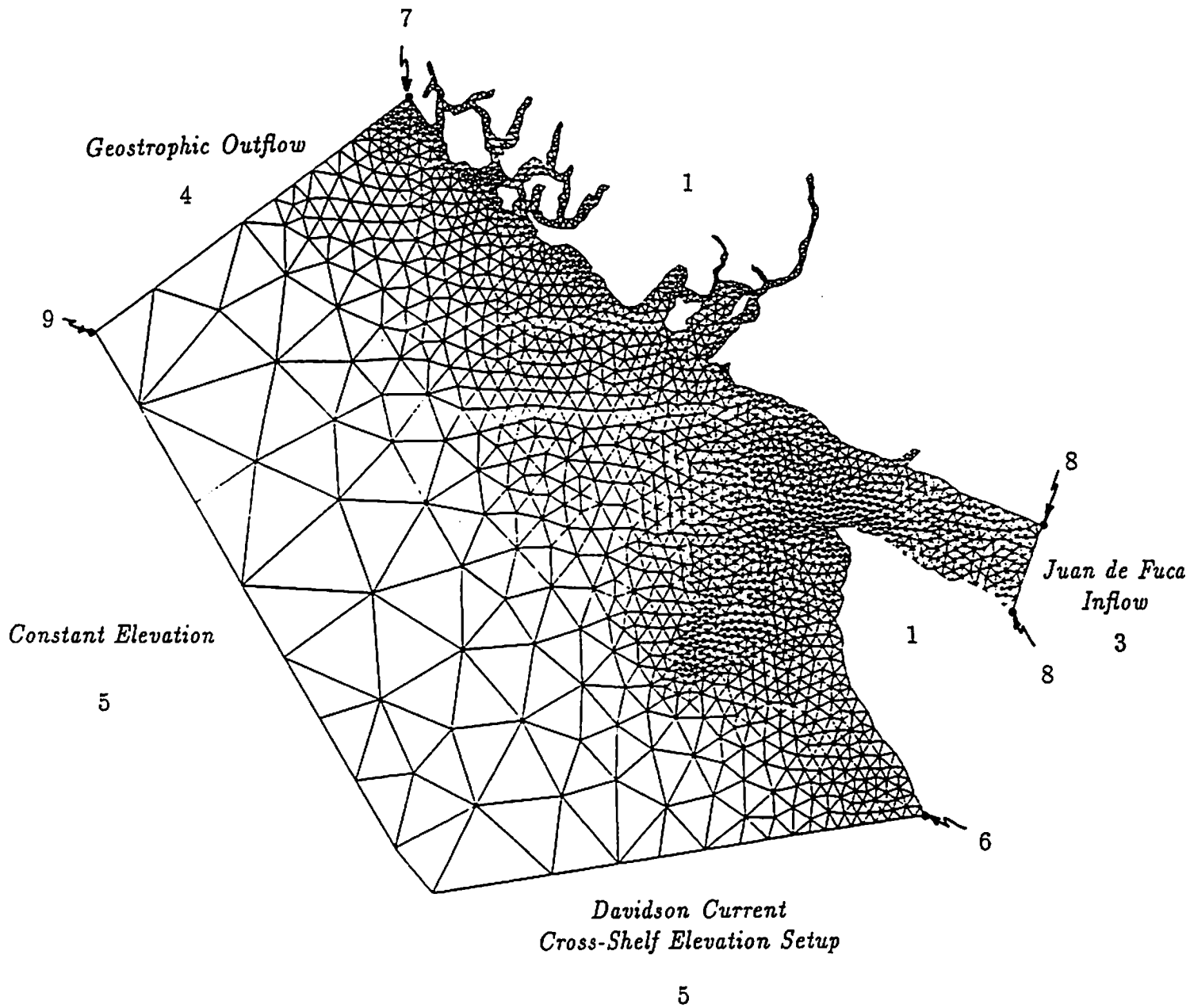

Figure 3b. As in figure 3a, with boundary condition codes for steady diagnostic simulation.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation, Grant CEE-8352226; and by the University of New Hampshire Seagrant Program, Grant R/MD-110. Michael Foreman and Francisco Werner were essential in testing and improving early model releases.

REFERENCES

Foreman, M.G.G. (1978) Manual for tidal currents analysis and prediction. Pacific Marine Science Report 78-6, Institute of Ocean Sciences, Sidney, BC, 70pp.

Foreman, M.G.G. and R.A. Walters (1990) A finite element tidal model for the southwest coast of Vancouver Island. *Atmosphere-Ocean* (in press).

Henry, R.F. (1988) Interactive design of irregular triangular grids. *Proc* VII Int. Conf. on Computational Methods in Water Resources, MIT, Cambridge, June 1988. Vol II, pp 445-450.

LABLIB: Numerical Methods Laboratory Library, Dartmouth College, Hanover, NH, July 1987.

Lynch, D.R. Mass balance in shallow water simulations, *Comm. in Appl. Num. Methods*, 1, 153-159, (1985).

Lynch, D.R. and F.E. Werner (1987) Three-dimensional hydrodynamics on finite elements. Part I: Linearized Harmonic Model. *Int. J. for Numerical Methods in Fluids.* 7, 871-909.

Lynch, D.R. and F.E. Werner (1990) Three-dimensional hydrodynamics on finite elements. Part II: Nonlinear time-stepping model. *Int. J. for Numerical Methods in Fluids* 11.

Lynch, D.R., F.E. Werner, D.A. Greenberg, J.W. Loder (1990) Diagnostic model for baroclinic, wind-driven and tidal circulation in shallow seas. *Continental Shelf Research* (in press).

Lynch, D.R. (1980) An explicit model for two-dimensional tidal circulation using triangular finite elements: WAVETL User's Manual. *Water Res. Investigation* 80-42, USGS.

APPENDIX

THEORY AND SOLUTION PROCEDURE

THEORY

We solve the linearized 3-D shallow water equations with conventional hydrostatic and Boussinesq assumptions, and eddy viscosity closure in the vertical. The density field is presumed known and constitutes a fixed baroclinic pressure gradient. The response (3-D velocity field plus barotropic pressure) to this forcing, combined with wind and barotropic forcing at open water boundaries, is sought on detailed topography. For generality and for compatibility with previous model development (LW87) we assume periodic-in-time solutions of the form $q(\mathbf{x},t) = Re(Q(\mathbf{x})e^{j\omega t})$, with Q the complex amplitude of q and ω the frequency. The steady responses are simply the limiting case $\omega = 0$.

The horizontal momentum equation is

$$j\omega\mathbf{V} + \mathbf{f}\times\mathbf{V} - \frac{\partial}{\partial z}\left(N\frac{\partial \mathbf{V}}{\partial z}\right) = \mathbf{G} + \mathbf{R} \tag{1}$$

$$N\frac{\partial \mathbf{V}}{\partial z} = h\Psi \qquad (z = 0) \tag{2}$$

$$N\frac{\partial \mathbf{V}}{\partial z} = k\mathbf{V} \qquad (z = -h) \tag{3}$$

in which

$\mathbf{R}(x,y,z) \equiv -\frac{g}{\rho_0}\int_z^0 \nabla\rho dz$ is the baroclinic pressure gradient, assumed known
$\mathbf{G}(x,y) \equiv -g\nabla\zeta$ is the barotropic pressure gradient, assumed unknown
$\rho(x,y,z,t)$ is the fluid density
$\zeta(x,y)$ is the free surface elevation
$\mathbf{V}(x,y,z)$ is the horizontal velocity
$W(x,y,z)$ is the vertical velocity
ω is the radian frequency
j is the imaginary unit, $\sqrt{-1}$
$h(x,y)$ is the bathymetric depth
$\mathbf{f} \equiv f\hat{\mathbf{z}}$ is the Coriolis vector
$N(x,y,z)$ is the vertical eddy viscosity
g is gravity
(x,y) are the horizontal co-ordinates
z is the vertical co-ordinate, positive upward with $z = 0$ at the surface
∇ is the *horizontal* gradient $(\partial/\partial x,\ \partial/\partial y)$
$h\Psi(x,y)$ is the atmospheric forcing
k is a linear bottom stress coefficient.

All hydrodynamic variables are represented as complex amplitudes of time-periodic motions; and throughout we indicate by an overbar the vertical average of any quantity. The vertical average of (1) is

$$j\omega\bar{\mathbf{V}} + \mathbf{f}\times\bar{\mathbf{V}} + \frac{k}{h}\mathbf{V}(-h) = \mathbf{G} + \Psi + \bar{\mathbf{R}} \tag{4}$$

In addition we have the continuity equation

$$\frac{\partial W}{\partial z} + \nabla \cdot \mathbf{V} = 0 \tag{5}$$

and its vertical average

$$j\omega\zeta + \nabla \cdot (h\bar{\mathbf{V}}) = 0 \tag{6}$$

We record also the weak form of (6):

$$\langle j\omega\zeta\, \phi_i \rangle - \langle h\bar{\mathbf{V}} \cdot \nabla\phi_i \rangle = -\oint h\bar{\mathbf{V}} \cdot \hat{\mathbf{n}}\phi_i\, ds \tag{7}$$

where $\langle\,\rangle$ is a domain integral over (x, y); $\oint ds$ is the enclosing boundary integral; $\hat{\mathbf{n}}$ is the unit normal, directed outward; and $\phi_i(x, y)$ is an arbitrary weighting function. Note that conventional horizontal boundary conditions will be enforced on either ζ or $h\bar{\mathbf{V}} \cdot \hat{\mathbf{n}}$ to close the boundary-value problem.

The momentum equation is simplified by introduction of the surrogate velocity variables

$$\nu^+ = \frac{V_x + jV_y}{2}; \qquad \nu^- = \frac{V_x - jV_y}{2} \tag{8}$$

$$V_x = \nu^+ + \nu^-; \qquad jV_y = \nu^+ - \nu^-. \tag{9}$$

which removes the Coriolis coupling:

$$j(\omega \pm f)\nu^\pm - \frac{\partial}{\partial z}\left(N\frac{\partial \nu^\pm}{\partial z}\right) = G^\pm + R^\pm \tag{10}$$

$$N\frac{\partial \nu^\pm}{\partial z} = h\psi^\pm \qquad (z = 0) \tag{11}$$

$$N\frac{\partial \nu^\pm}{\partial z} = k\nu^\pm \qquad (z = -h) \tag{12}$$

with forcing terms defined as

$$G^\pm = \frac{G_x \pm jG_y}{2} \tag{13}$$

$$\psi^\pm = \frac{\psi_x \pm j\psi_y}{2} \tag{14}$$

$$R^\pm = \frac{R_x \pm jR_y}{2} \tag{15}$$

By inspection, the solution to (10-12) can be written as

$$\nu^{\pm}(z) = G^{\pm}P_1^{\pm}(z) + \psi^{\pm}P_2^{\pm}(z) + P_3^{\pm}(z) \tag{16}$$

where the functions $P_i^{\pm}$ each satisfy the simple diffusion equation (10), forced as follows:*

$$\begin{aligned} P_1^{\pm} &: G = 1; \psi = 0; R = 0. \\ P_2^{\pm} &: G = 0; \psi = 1; R = 0. \\ P_3^{\pm} &: G = 0; \psi = 0; R = R^{\pm} \end{aligned} \tag{17}$$

Recovery of V from $\nu^{\pm}$ is then straightforward, using (9):

$$\begin{aligned} \mathbf{V}(z) = \mathbf{G}\left(\frac{P_1^+ + P_1^-}{2}\right) - j\left(\frac{P_1^+ - P_1^-}{2}\right)\hat{\mathbf{z}} \times \mathbf{G} \\ + \Psi\left(\frac{P_2^+ + P_2^-}{2}\right) - j\left(\frac{P_2^+ - P_2^-}{2}\right)\hat{\mathbf{z}} \times \Psi \\ + \hat{\mathbf{x}}\left(P_3^+ + P_3^-\right) - j\hat{\mathbf{y}}\left(P_3^+ - P_3^-\right) \end{aligned} \tag{18}$$

expressing a superposition of responses to barotropic, wind, and density gradient forcing. Note that the six functions $P_i^{\pm}$ can be obtained independently at any horizontal position by any of several methods for solving the 1-D diffusion equation.

The unknown barotropic pressure gradient $\mathbf{G} \equiv -g\nabla\zeta$ in (18) is determined by application of the vertically averaged continuity equation (6). Substitution of (18) into (6) or its weak form (7) eliminates $\bar{\mathbf{V}}$ and produces a scalar Helmholtz-like equation in ζ alone. The resulting weak form is

$$\begin{aligned} \langle j\omega\zeta\,\phi_i\rangle + \left\langle \left[\left(\frac{\bar{P}_1^+ + \bar{P}_1^-}{2}\right)gh\nabla\zeta - j\left(\frac{\bar{P}_1^+ - \bar{P}_1^-}{2}\right)\hat{\mathbf{z}} \times gh\nabla\zeta\right] \cdot \nabla\phi_i \right\rangle = \\ - \oint j\bar{\mathbf{V}} \cdot \hat{\mathbf{n}}\phi_i\, ds + \left\langle \left[\left(\frac{\bar{P}_2^+ + \bar{P}_2^-}{2}\right)h\Psi - j\left(\frac{\bar{P}_2^+ - \bar{P}_2^-}{2}\right)\hat{\mathbf{z}} \times h\Psi\right] \cdot \nabla\phi_i \right\rangle \\ + \left\langle \left[\left(\bar{P}_3^+ + \bar{P}_3^-\right)h\hat{\mathbf{x}} - j\left(\bar{P}_3^+ - \bar{P}_3^-\right)h\hat{\mathbf{y}}\right] \cdot \nabla\phi_i \right\rangle \end{aligned} \tag{19}$$

This 2-D, horizontal equation is especially amenable to Galerkin finite element solution on simple linear triangular elements. Its solution provides the barotropic pressure response

* The distinction between P_2 and P_3 is maintained here for its interpretive content; but is not necessary. The merger of P_2 into P_3, by the alternate definition ($P_3^{\pm} : G = 0; \psi = \psi^{\pm}; R = R^{\pm}$) allows use of all the formulae below with the simplification $P_2^{\pm} = 0$ everywhere.

which accompanies the imposed wind, density field, and open-water barotropic boundary conditions.

Integration of (10) from $z = -h$ to $z = 0$, and use of (11,12), provides a useful relation between $\bar{\nu}^{\pm}$ and $\nu^{\pm}(-h)$:

$$j(\omega \pm f)\bar{\nu}^{\pm} - \psi^{\pm} + \frac{k}{h}\nu^{\pm}(-h) = \bar{G}^{\pm} + \bar{R}^{\pm} \tag{20}$$

As in LW87, we find for the various $P_i(z)$:

$$\bar{P}^{\pm}_{1,2} = \frac{1}{j(\omega \pm f)}\left[1 - \frac{k}{h}P^{\pm}_{1,2}(-h)\right] \tag{21}$$

$$\bar{P}^{\pm}_{3} = \frac{1}{j(\omega \pm f)}\left[\bar{R}^{\pm} - \frac{k}{h}P^{\pm}_{3}(-h)\right] \tag{22}$$

which may be used to avoid the calculation of the vertical averages.**

An alternate route to an equivalent statement of the horizontal problem is available following LW87. This approach takes advantage of the fact that the bottom stress may be expressed in terms of $\bar{\mathbf{V}}$, reducing the vertically averaged momentum equation to an equivalent 2-D form. First, $G^{\pm}$ can be eliminated from (16) by use of its vertical average:

$$G^{\pm} = \frac{1}{\bar{P}^{\pm}_{1}}\left[\bar{\nu}^{\pm} - \psi^{\pm}\bar{P}^{\pm}_{2} - \bar{P}^{\pm}_{3}\right] \tag{23}$$

and therefore

$$\nu^{\pm}(z) = \left(\frac{P_1(z)}{\bar{P}_1}\right)^{\pm}\bar{\nu}^{\pm} + \left(P_2(z) - P_1(z)\frac{\bar{P}_2}{\bar{P}_1}\right)^{\pm}\psi^{\pm} + \left(P_3(z) - P_1(z)\frac{\bar{P}_3}{\bar{P}_1}\right)^{\pm} \tag{24}$$

It follows that

$$k\nu^{\pm}(-h) = \tau^{\pm}h\bar{\nu}^{\pm} - \alpha^{\pm}h\psi^{\pm} - \beta^{\pm}h \tag{25}$$

with

$$\tau^{\pm} \equiv \frac{kP^{\pm}_{1}(-h)}{h\bar{P}^{\pm}_{1}} \tag{26}$$

$$\alpha^{\pm} \equiv \tau^{\pm}\bar{P}^{\pm}_{2} - \frac{k}{h}P^{\pm}_{2}(-h) \tag{27}$$

$$\beta^{\pm} \equiv \tau^{\pm}\bar{P}^{\pm}_{3} - \frac{k}{h}P^{\pm}_{3}(-h) \tag{28}$$

Recovery of the bottom stress in the original (x, y) system, via (9), yields

** If as suggested above P_2 and P_3 are merged, replace $\bar{R}^{\pm}$ with $\bar{R}^{\pm} + \psi^{\pm}$ in (22)

$$k\mathbf{V}(-h) = \left(\frac{\tau^+ + \tau^-}{2}\right) h\bar{\mathbf{V}} - j\left(\frac{\tau^+ - \tau^-}{2}\right)\hat{\mathbf{z}} \times h\bar{\mathbf{V}}$$
$$-\left(\frac{\alpha^+ + \alpha^-}{2}\right) h\Psi + j\left(\frac{\alpha^+ - \alpha^-}{2}\right)\hat{\mathbf{z}} \times h\Psi$$
$$-\left(\beta^+ + \beta^-\right) h\hat{\mathbf{x}} + j\left(\beta^+ - \beta^-\right) h\hat{\mathbf{y}} \tag{29}$$

Finally, use of (29) in (4) to eliminate the bottom velocity gives the equivalent 2-D system

$$j\omega\bar{\mathbf{V}} + \mathbf{f}' \times \bar{\mathbf{V}} + \tau'\bar{\mathbf{V}} = \mathbf{G} + \Psi' + \bar{\mathbf{R}}' \tag{30}$$

where the prime quantities $\mathbf{f}'$, τ', Ψ' and $\bar{\mathbf{R}}'$ all contain contributions from the bottom stress:

$$\mathbf{f}' = \mathbf{f} - j\left(\frac{\tau^+ - \tau^-}{2}\right)\hat{\mathbf{z}} \tag{31}$$

$$\tau' = \frac{\tau^+ + \tau^-}{2} \tag{32}$$

$$\Psi' = \Psi\left[1 + \left(\frac{\alpha^+ + \alpha^-}{2}\right)\right] - j\left(\frac{\alpha^+ - \alpha^-}{2}\right)\hat{\mathbf{z}} \times \Psi \tag{33}$$

$$\bar{\mathbf{R}}' = \bar{\mathbf{R}} + \hat{\mathbf{x}}(\beta^+ + \beta^-) - j\hat{\mathbf{y}}(\beta^+ - \beta^-) \tag{34}$$

As in LW87, $\tau^\pm$, $\alpha^\pm$ and $\beta^\pm$ depend on $\omega \pm f$, $N(z)$, h, and k and thus vary with frequency as well as (x, y); and all of the vertical detail is embodied without loss of information in the parameters $\mathbf{f}'$, τ', Ψ', and $\bar{\mathbf{R}}'$. This 2-D system permits the classical expression of $\bar{\mathbf{V}}$ in terms of the gravity, wind, and baroclinic forcing:

$$\bar{\mathbf{V}} = \left(\frac{(j\omega + \tau')(\mathbf{G} + \Psi' + \bar{\mathbf{R}}') - \mathbf{f}' \times (\mathbf{G} + \Psi' + \bar{\mathbf{R}}')}{(j\omega + \tau')^2 + f'^2}\right) \tag{35}$$

This may in turn be substituted into the vertically integrated continuity equation to produce a Helmholtz equation equivalent to (19):

$$\langle j\omega\zeta\, \phi_i \rangle + \left\langle \left[\frac{(j\omega + \tau')gh\nabla\zeta - \mathbf{f}' \times gh\nabla\zeta}{(j\omega + \tau')^2 + f'^2}\right] \cdot \nabla\phi_i \right\rangle =$$
$$-\oint h\bar{\mathbf{V}} \cdot \hat{\mathbf{n}}\phi_i\, ds + \left\langle \left[\frac{(j\omega + \tau')h(\Psi' + \bar{\mathbf{R}}') - \mathbf{f}' \times h(\Psi' + \bar{\mathbf{R}}')}{(j\omega + \tau')^2 + f'^2}\right] \cdot \nabla\phi_i \right\rangle \tag{36}$$

Like (19), this equation allows computation of ζ as a scalar, 2-D problem subject to barotropic boundary conditions. While the derivation of (36) is more circuitous, it provides a simple set of recipes for converting /upgrading any 2-D shallow water solver based on the linearized harmonic equations to the present 3-D diagnostic level. In addition the equivalent 2-D momentum equation (30) provides some insight in the departure of the prime quantities from their conventional 2-D forms, which is not readily obtained from the more direct form (19). This approach is adopted in the implementation of FUNDY4.

SOLUTION PROCEDURE

The numerical solution is implemented in four sequential steps, using a finite element mesh of linear triangles in the horizontal:

1) The vertical structure is computed in terms of τ', f', Ψ' and $\bar{\mathbf{R}}'$ at each node. The six solutions $P^{\pm}_{1,2,3}$ are computed under each node – each requires solution of a 1-D diffusion equation, which we solve by the Galerkin method on 1-D linear finite elements. Because these are tridiagonal systems, they are not a limiting factor in the overall computational method.

2) The surface elevation is obtained by the Galerkin method on the horizontal grid of triangles. Expanding the solution in terms of unknown nodal values ζ_j and the triangular basis functions ϕ_j:

$$\zeta(x,y) = \sum_j \zeta_j \phi_j(x,y) \tag{37}$$

we obtain from (36) the matrix equation

$$[A]\{\zeta\} = \{B\} - \{F\}$$

$$A_{ij} = \langle j\omega\phi_j\phi_i\rangle + \left\langle \left[\frac{(j\omega+\tau')gh\nabla\phi_j - \mathbf{f}'\times gh\nabla\phi_j}{(j\omega+\tau')^2+f'^2}\right]\cdot\nabla\phi_i \right\rangle$$

$$B_i = \left\langle \left[\frac{(j\omega+\tau')h(\Psi'+\bar{\mathbf{R}}') - \mathbf{f}'\times h(\Psi'+\bar{\mathbf{R}}')}{(j\omega+\tau')^2+f'^2}\right]\cdot\nabla\phi_i \right\rangle$$

$$F_i = \oint h\bar{\mathbf{V}}\cdot\hat{\mathbf{n}}\phi_i\, ds \tag{38}$$

In the present implementation all inner products are evaluated numerically, with quadrature points at the nodes of the triangles. Barotropic BC's are enforced on this system in any of three ways:

Type I: Elevation known. In this case the Galerkin equation weighted by ϕ_i is removed in favor of exact specification of ζ_i.

Type II: $\bar{\mathbf{V}}\cdot\hat{\mathbf{n}}$ known. In this case the boundary transport integral F_i is evaluated from the given BC.

Type III: Geostrophically balanced transport. In this case neither elevation nor transport are known, but a geostrophic balance is assumed between them:

$$h\bar{\mathbf{V}}\cdot\hat{\mathbf{n}} = \frac{h}{f}(\mathbf{G}+\Psi'+\bar{\mathbf{R}}')\cdot\hat{\mathbf{t}} \tag{39}$$

where $\hat{\mathbf{t}}$ is the local tangential direction. (Essentially we assume $h\bar{\mathbf{V}}\cdot\hat{\mathbf{t}} = 0$.) This relation is substituted into the transport integral F_i; the known parts $(\Psi'+\bar{\mathbf{R}}')\cdot\hat{\mathbf{t}}$ are moved into B_i; the unknown part $\mathbf{G}\cdot\hat{\mathbf{t}} \equiv -g\sum\zeta_j\frac{\partial\phi_j}{\partial t}$ is moved to the left-side and embedded in the matrix $[A]$:

$$A'_{ij} = A_{ij} - \int \frac{gh}{f} \frac{\partial \phi_j}{\partial t} \phi_i \, ds$$

$$B'_{ij} = B_{ij} - \int \frac{h}{f} (\Psi' + \bar{\mathbf{R}}') \cdot \hat{\mathbf{t}} \, \phi_i \, ds \tag{40}$$

with $\int ds$ indicating integration over the Type III boundary only.

3) Velocity profiles. Once ζ is available, we differentiate it numerically to obtain nodal values of $\mathbf{G}$ by a Galerkin approximation:

$$\sum_j \langle \phi_i \phi_j \rangle \mathbf{G_j} = -\langle g \nabla \zeta \phi_i \rangle \tag{41}$$

Nodal quadrature reduces the mass matrix $\langle \phi_i \phi_j \rangle$ to a diagonal matrix, greatly simplifying this calculation. Once the $\mathbf{G}_i$ are computed, the velocity profiles are either assembled from memory according to (18); or recomputed by a single tridiagonal calculation under each horizontal node.

4) Vertical velocities. Finally, we compute the vertical velocities at every node from the continuity equation (5). To do so requires construction of a 3-D FE mesh in order to differentiate $\mathbf{V}(x, y, z)$, and we follow exactly the procedure given in LW90. The horizontal mesh is projected downward in perfectly vertical lines and each is discretized into the same number of vertical elements. These are then connected horizontally in the identical topology as the original 2-D mesh, thereby filling the volume with 6-node linear elements. Effectively, this creates an (x, y, σ) coordinate system. Unless otherwise stated, the simulations here employ uniform relative vertical mesh spacing everywhere, i.e. uniform $\Delta\sigma$.