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

FUNDY5 USERS' MANUAL

Christopher E. Naimie and Daniel R. Lynch
Dartmouth College, NH

March 30, 1993

FUNDY5 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 FUNDY5 system is presented. Detailed specifications are provided for the required data files and user-written FORTRAN subroutines. A comprehensive example is provided. The model theory and numerical method are described in an appendix.

OVERVIEW

FUNDY5 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 (1992) (herein, LW87 and LWGL92). The model is limited to the linearized equations, with an externally specified density field. The SI (MKS) system of units is used. 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 FUNDY5 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 Appendix A. Standard data file conventions are detailed in Appendix B.

Comparison with FUNDY4 - Though a variety of changes were made to improve the code, the implementation of FUNDY5 is very similar to FUNDY4. The input file, Subroutine BC and Subroutine ATMOS have not changed with this revision; however, User Subroutines VERTGRID and OUTPUT must be replaced with the newer versions named VERTGRID5 and OUTPUT5. The changes made range from computational differences to enhance the solution and code restructuring to increase code efficiency to the inclusion of additional subroutines to reduce the requirement for variants to the software. Appendix C contains a listing of the differences between FUNDY4 and FUNDY5.

3-D Mesh Generation - The model uses a conventional horizontal grid of linear triangles, which is specified by the user in an input file. A 3-D mesh is automatically constructed within FUNDY5 by projecting the horizontal mesh downward to the bottom in perfectly vertical lines, with each line 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 and 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).

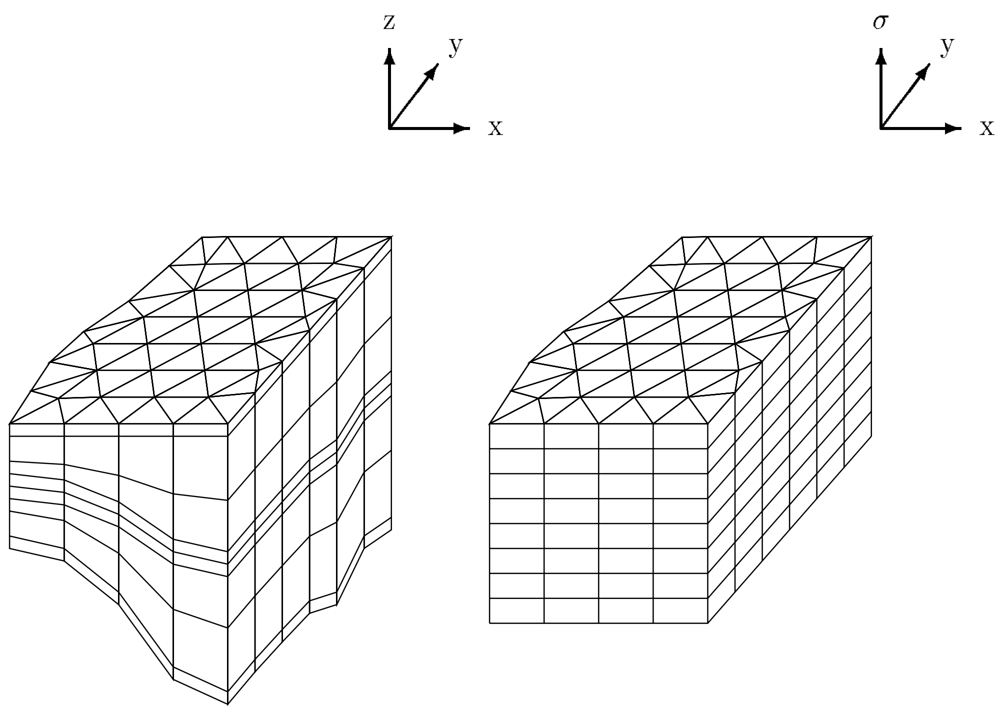


Figure 1. Main features of the layered mesh

- element sides perfectly vertical
- variable vertical mesh spacing allows resolution of boundary and internal layers
- mesh spacing is uniform in mapped (x,y,σ) coordinate system

Program Structure (See Figure 2):

Main Program and Fixed Subroutines - FUNDY5 is the core program, which performs all finite element assembly and solution operations, according to Appendix A. FUNDY5 reads a formatted input file and writes a formatted echo file, the latter containing a summary of the input file and run data.

User Subroutines - Four user-built subroutines must be linked to FUNDY5 to specify the boundary conditions, the physical forcing, the vertical structure, and the manner in which the results are to be written. FUNDY5.USER.f is a shell from which these subroutines may be constructed.

Include File - The include file FUNDY5ARRAYS.DIM assigns values to parameters required for dimensioning purposes and SI (MKS) values of the reference density and gravitational acceleration.

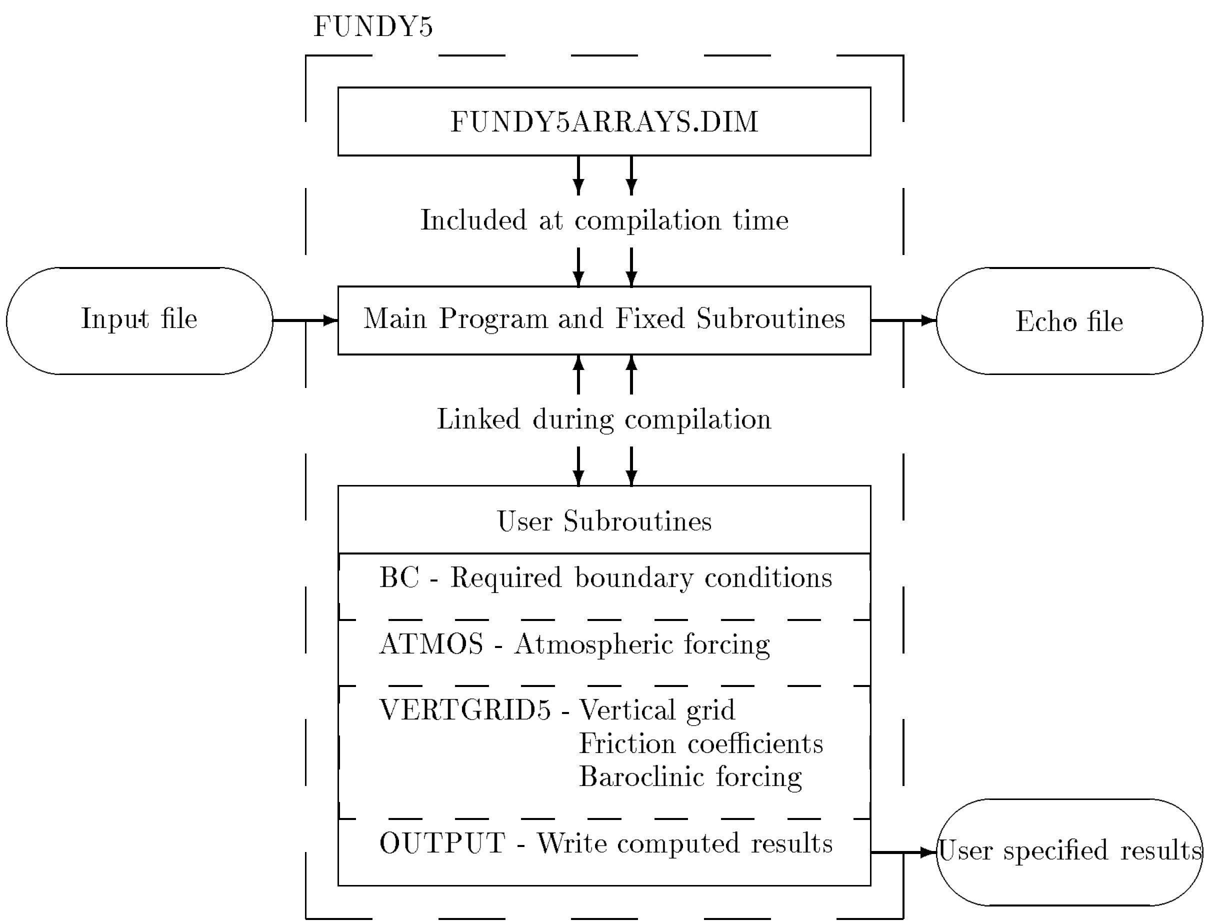


Figure 2. FUNDY5 program structure

User Interface:

Creating an input file - Prior to running FUNDY5, the user must first generate an input file that conforms to the **.inp** Standard (See next section). This can be accomplished by using either the FORTRAN 77 program TRIGTOFUNDY4 or the FORTRAN 77 program CONVCODES. The head of the resulting input file contains incomplete FORTRAN statements for every node that requires data input in Subroutine BC.

Creating a set of User Subroutines - Four user subroutines must be linked to FUNDY5 to specify the boundary conditions, the physical forcing, the vertical structure, and the manner in which the results are to be written. Specifications for these subroutines are available in their respective shells, which contain the overall structure, argument list, dimensioning, and declarations sufficient as starting points for their construction.

Running FUNDY5 - The user will be asked to specify the input and echo files at the beginning of the run. Any other user interaction/output is dependent on the user's treatment of the User Subroutines.

Analyzing the echo file - The echo file should be studied closely after the first run with a new input file and/or a new set of User Subroutines. The echo file contains useful information regarding the size of dimensioned arrays, boundary condition consistency with the input file, and run statistics. When the run is successfully completed, the last line of the echo file will be "FUNDY5 run successfully completed". If the run is terminated due to dimensioning or boundary condition problems, the last line of the echo file is a message describing the error.

CREATING AN INPUT FILE

INPUT FILE SPECIFICATIONS

The input file 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. The FUNDY5 input file format is identical to the FUNDY4 input file format. Users with existing FUNDY4 input files may use them without alteration for FUNDY5 runs.

The input file (**.inp** Standard) contains all of the mesh data required to run FUNDY5 and identifies what boundary conditions will need to be specified in Subroutine BC. The required boundary conditions are determined by an integer classification NTYPE. All NTYPE classifications except 0 correspond to boundary nodes. Conventions for NTYPE are as follows:

0: interior point
1: land
2: island
3: nonzero normal velocity
4: geostrophic outflow
5: surface elevation
6: corner - elevation and land or island
7: corner - geostrophic outflow and land or island
8: corner - nonzero normal velocity and land or island
9: corner - elevation and geostrophic
10: corner - both components of velocity equal zero
11: corner - both components of velocity nonzero

TRIGTOFUNDY4

This FORTRAN 77 program assembles and writes a formatted input file for FUNDY4 or FUNDY5 from node and element files describing the horizontal mesh. The node and element file formats are compatible with output from the TRIGRID mesh generation package (Henry 1988).

This program first asks for the names of the TRIGRID NODE[1] and ELEMENT[2] files to be read, and also for the name of the FUNDY4 input file to be written. During execution the program also asks for the following quantities:

[1] The format of the TRIGRID NODE file is similar to the **.nei** Standard. The difference between them is that the TRIGRID NODE file contains the NTYPE codes for the desired boundary conditions while the node type codes for the **.nei** file are limited to 0,1,or 2 [See Appendix B].

[2] The format of the TRIGRID ELEMENT file is identical to the **.ele** Standard [See Appendix B].

title - an ASCII string to be written into the the **.inp** file to identify it
latitude to be used in computing the Coriolis parameter
scaling factors for X, Y
scaling factor for bathymetry and minimum bathymetric depth Hmin

As written by TRIGTOFUNDY4, the input file contains extra information at the head of the file. First, any diagnostics generated during the processing are written. Following these, incomplete FORTRAN statements appear for every boundary node which requires data in Subroutine BC. A complete Subroutine BC can be constructed by completing these statements and inserting them in the appropriate points in the Subroutine BC shell, described below.

For each node I where the surface elevation is specified (NTYPE=5, 6, or 9), a line of the following form will be written:

U(I)=

For each node I where a nonzero normal velocity is specified (NTYPE=3), a line of similar form will be written:

U(I)=

For each node I where both components of velocity are specified to be nonzero (NTYPE = 11), two statements of the following form will be written:

U(I)=
V(I)=

A special case is NTYPE=8 (corners). For each node I of this type, the following 3-line sequence will be written:

VNORM=
U(I)=NX*VNORM
V(I)=NY*VNORM

(NX,NY) are numbers representing the x- and y-components of the **outward-pointing unit vector, parallel to the land/island at the corner**. In this case, the user specifies the value of the normal velocity parallel to the land boundary (VNORM).

Following all this information, the appropriate input information for FUNDY5 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 FUNDY5; the balance will be processed.

CONVCODES

This FORTRAN 77 program is designed to perform several mesh data file format conversions to create a formatted input file for FUNDY4 or FUNDY5. CONVCODES produces input files that conform identically to the TRIGTOFUNDY4 format discussed above. Documentation for this code is contained in Wolff (1993).

USER SUBROUTINES

As previously mentioned, four user subroutines must be linked to FUNDY5 to specify the boundary conditions, the physical forcing, the vertical structure, and the manner in which the results are to be written. Specifications for these subroutines are available in their respective shells, which contain the overall structure, argument list, dimensioning, and declarations sufficient as starting points for their construction.

Subroutine BC[3] - This subroutine must be modified by the user to assign proper Boundary Conditions. There are four sections which require modification, beginning at statement labels 3,4,5, and 6. Each section must begin with the appropriately labeled continue statement and end with a return statement.

Frequency (label 3) The first section assigns the frequency of the forcing (in radians per second) to WW. Time variation of the form $exp(i\omega t)$ is assumed.

Surface Elevation (label 4) - Surface elevation BC's are assigned in this section. One statement of the form

U(I)=CVALUE

must appear for each specified surface elevation node[4], where I is the horizontal node number and CVALUE is the complex surface elevation for node I.

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

U(I)=CVALUE

must appear for each set nonzero normal velocity node, where I is the horizontal node number and CVALUE is the complex value of the normal velocity for node I. Positive normal velocity indicates flow out of the system. Note that boundary nodes where the normal velocity is zero are handled by default in FUNDY5 (See Appendix A); they are **not** dealt with here.

Nonzero Velocity (label 6) - Nonzero velocity BC's are assigned in this section. Two statements of the following form must appear for each nonzero velocity node:

U(I)=CXVEL
V(I)=CYVEL

I is the horizontal node number, (CXVEL,CYVEL) are the complex values of the (X,Y) velocities for node I.

[3]Discrepancies between the boundary condition types specified in the **.inp** file and those applied in Subroutine BC are reported to the echo file upon execution of FUNDY5.

[4]The main program initializes all required elevation boundary condition values to zero before Subroutine BC is called. If the user does not specify the boundary condition for a node where one is required, the default of zero elevation is applied.

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:

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

(See Equation 2 in Appendix A; in that notation $\mathbf{ATM} = h\Psi$.) Two statements of the following form must appear for each node where the atmospheric forcing is nonzero:

ATMX(I)=XVALUE
ATMY(I)=YVALUE

I is the node number and (XVALUE,YVALUE) are the complex values of the forcing. The main program initializes all (ATMX,ATMY) values to zero before Subroutine ATMOS is called. Therefore, the default of zero atmospheric forcing is applied if the user does not specify (ATMX,ATMY) values for a node in Subroutine ATMOS.

Subroutine VERTGRID5 - 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 arrays:

Z(I,J) - Vertical coordinate of 3-D nodes
AK(I) - Linear bottom stress coefficient, k
ENZ(I,J) - Vertical viscosity, Nz
RX(I,J) - RHS of x-momentum equation
RY(I,J) - RHS of y-momentum equation
HRBARELX(L) - RHS of elevation equation
HRBARELY(L) - RHS of elevation equation

The first line of vertgrid following the dimensioning should assign NNV (on return to the main program, a check of NNV relative to NNVDIM will be made). Then, there must be assignments to fill the aforementioned REAL (Z,AK,ENZ) and COMPLEX (RX,RY,HRBARELX,HRBARELY) arrays over the index range I=1,NN; J=1,NNV; L=1,NE.

NN - Number of horizontal nodes
NE - Number of horizontal 2-D triangular elements
NNV - Number of vertical nodes

Keep in mind that **Z and J are positive upward**:

Bottom: Z(I,J)=-HDOWN(I); J=1
Surface: Z(I,J)= 0 ; J=NNV

A variety of options are available to the user through the use of fixed subroutines:

1. Subroutine UNIGRID5 sets uniform spacing in the vertical which depends on nodal bathymetric depth and NNV.

Z - Uniform spacing which depends on nodal bathymetric depth and NNV
AK - Equal to the default value AK0
ENZ - Equal to default value EN0
RX,RY,HRBARELX,HRBARELY - equal to zero (no baroclinic forcing)

2. Subroutine SINEGRID5 sets a sinusiodal spacing in the vertical, decreasing spacing at the surface and bottom of the water column in an effort to better resolve ekman dynamics. DZBL sets the minimum vertical node spacing for which sinusoidal node placement is used. If the DZBL exceeds the bathymetric depth divided by NNV-1, uniform spacing is used; otherwise, sinusoidal vertical nodal placement is used.

Z - Sinusoidal spacing which depends on nodal bathymetric depth, DZBL, and NNV
AK - Equal to the default value AK0
ENZ - Equal to default value EN0
RX,RY,HRBARELX,HRBARELY - equal to zero (no baroclinic forcing)

3. Subroutine RHOGRID5 reads sigma mesh baroclinic forcing from a file conforming to the **.rho** Standard (See Appendix B).

Z - Read from **.rho** file
AK - Equal to the default value AK0
ENZ - Equal to default value EN0
RX,RY,HRBARELX,HRBARELY - read from **.rho** file

4. Subroutine BAROCLINIC5 computes the baroclinic forcing from a level surface σ_t data file that conforms to the **.lst** Standard (See Appendix B). Z must be computed prior to calling this routine. This routine is designed to be used in conjunction with UNIGRID5 or SINEGRID5.

Subroutine OUTPUT5 - This routine is responsible for writing any and all computed output to the user's specifications. The OUTPUT5 shell contains complete specifications and array dimensioning for the following REAL (X,Y,Z) and COMPLEX (ZETA,UBAR,VBAR,U,V,W) arrays:

X(I),Y(I) - Cartesian nodal coordinates
Z(I,J) - Vertical coordinate of 3-D nodes
ZETA(I) - Free surface elevation at nodes
UBAR(I) - Vertically averaged velocity in x-direction
VBAR(I) - Vertically averaged velocity in y-direction
U(I,J),V(I,J),W(I,J) - x,y,and z components of velocity

Index range: I=1,NN, J=1,NNV

NN - Number of horizontal nodes
NNV - Number of vertical nodes

A standard routine DUMP5 is available within FUNDY5; it writes all computed results to a user-specified file, in ASCII format. DUMP5 takes time and produces a large file; use only when needed. Specifications for DUMP5 are contained in the OUTPUT5 shell.

There are three FORTRAN 77 Functions included with the Fixed Subroutines that are useful in converting complex results to amplitude and phase format for writing Standard Output Files (See Appendix B for File Standards). A description of this conversion is contained in the General Notes section of this Manual.

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

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

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

The following FORTRAN 77 Subroutine [adapted from TEMP.FOR (IOS)] may be used to compute the 8 real-valued tidal ellipse parameters defined in Foreman (1978) for given complex velocity amplitudes U,V.

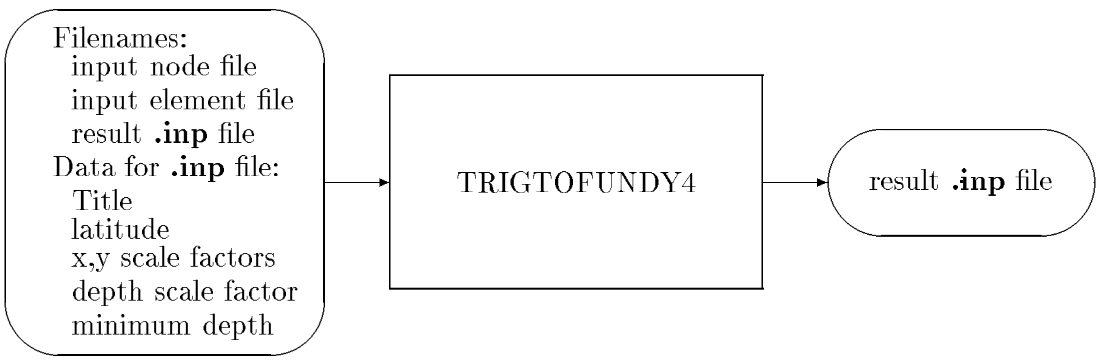
Subroutine ELLIPSE (U,V,amaj,amin,ainc,g,apl,amn,gpl,gmin)

ADDITIONAL PROGRAMS

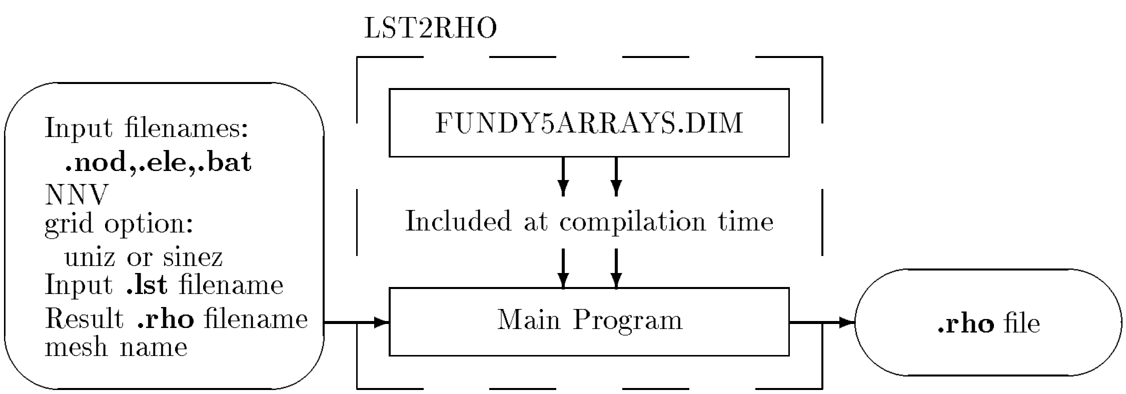
Three FORTRAN 77 programs are provided in addition to FUNDY5.

CONVCODES - This FORTRAN 77 program is designed to perform several mesh data file format conversions to create a formatted input file for FUNDY4 or FUNDY5, as discussed in Wolff (1993) and the CREATING AN INPUT FILE section of this report.

TRIGTOFUNDY4 - This FORTRAN 77 program assembles and writes a formatted input file for FUNDY4 or FUNDY5, from node and element files whose formats are compatible with output files of the TRIGRID mesh generation package (Henry 1988). Its use is discussed in the CREATING AN INPUT FILE section of this report.



LST2RHO - This FORTRAN 77 program converts a **.lst** Standard file to a **.rho** Standard file by calling FUNDY5 Fixed Subroutines. As discussed in the USER SUBROUTINES/Subroutine VERTGRID section of this document, standard subroutines exist to allow baroclinic forcing to be specified using either a **.lst** Standard file or a **.rho** Standard file. This program is included since there is a significant trade-off between using these two types of files. The **.lst** file has the advantage of being much smaller. However, Subroutine BAROCLINIC requires a large amount of memory at run time. The combined run time memory requirements of Subroutine BAROCLINIC and FUNDY5 may be a limiting factor for some computers.



EXAMPLE APPLICATION

As an example, the application of FUNDY5 to the Georges Bank/Gulf of Maine Region is presented. The 2-D triangular mesh (gb1) used in Lynch and Naimie (1993) appears in Figure 3, marked with node type codes appropriate for a M_2 tidal simulation (Figure 3a.) and for a steady diagnostic simulation (Figure 3b.).

Application files for the M_2 tidal simulation are available in the **fundy5/examples/gb1m2** subdirectory.

Data files necessary to run FUNDY5

gb1m2.inp	**.inp** Standard input file
gb1m2.USER.f	FUNDY5 USER Subroutines

FUNDY5 run result files

gb1m2.echo	FUNDY5 echo file
m2vbar.v2c	vertically averaged tidal velocity **.v2c** Standard file
m2zeta.s2c	free surface tidal elevation **.s2c** Standard file

Data files that can be used to produce the input file gb1m2.inp using CONVCODES:

gb1.nod	**.nod** Standard node file
gb1.ele	**.ele** Standard element file
gb1.bat	**.bat** Standard bathymetry file
gb1m2.ncd	**.ncd** Standard node code file

Application files for the steady diagnostic simulation are available in the **fundy5/examples/gb1zero** subdirectory:

Data files necessary to run FUNDY5

gb1zero.inp	**.inp** Standard input file
gb1zero.USER.f	FUNDY5 USER Subroutines

FUNDY5 run result files

gb1zero.echo	FUNDY5 echo file
resvbar.v2r	vertically averaged Eulerian velocity **.v2r** Standard file
resvbot.v2r	bottom Eulerian velocity **.v2r** Standard file
resvtop.v2r	surface Eulerian velocity **.v2r** Standard file
reszeta.s2r	free surface tidal elevation **.s2r** Standard file

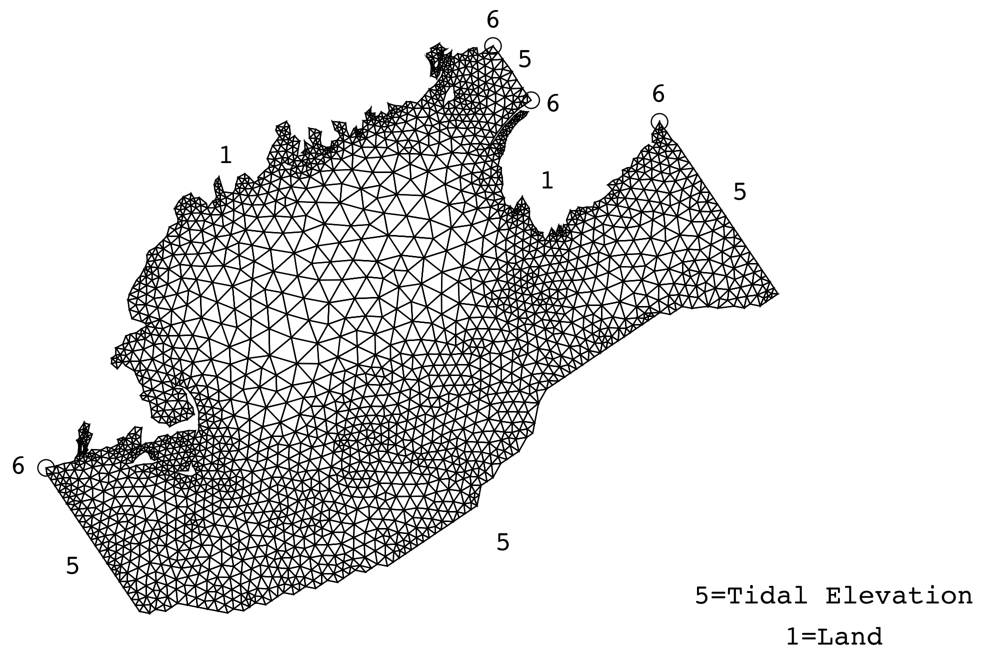


Figure 3a. Boundary Condition node type codes for M2 tidal simulation.

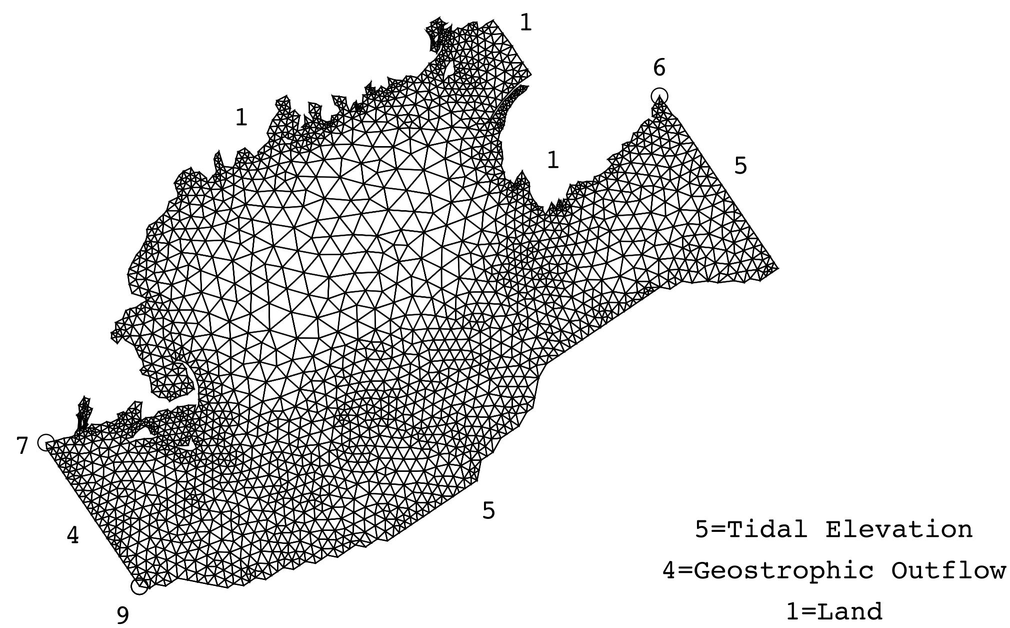


Figure 3b. Boundary Condition node type codes for steady diagnostic simulation.

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 input and output arrays use this convention, except the level surface sigmat file (**.lst** format). 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 that their physical and/or relative spacing is arbitrary and at the user's discretion. Therefore, equal values of J do 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: Z=-h, J=1

Surface: Z=0,J=NNV

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

4. FUNDY5 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 $Aexp(i\omega t)$ is assumed for all hydrodynamic variables. i is the $\sqrt{-1}$; ω is the radian frequency; t is the time in seconds; A is the complex amplitude. A may be reexpressed in terms of the real-valued amplitude a and the phase**lag** in degrees: $A \equiv aexp(\frac{-i\pi\phi}{180})$. Equivalently, a is the complex absolute value of A and $tan(\phi) = -Im(A)/Re(A)$.

USER NOTES

USER NOTES

ACKNOWLEDGMENTS

We thank David Greenberg for providing the finite element mesh from which the gb1 mesh was derived, John Loder for providing the level surface density and wind data, Elizabeth Wolff for creating CONVCODES, and other FUNDY4 users who provided insightful feedback regarding its performance. Charles Hannah's input during the final stages of FUNDY5 development were especially noteworthy. This work was supported by the National Science Foundation,Grant OCE-9012612 and 9016921.

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.

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

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

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. (1990). *Three-dimensional diagnostic model for baroclinic, wind-driven and tidal circulation in shallow seas; FUNDY4 User's Manual.* Dartmouth College Numerical Methods Laboratory Report, No. NML-90-2, Hanover, New Hampshire, U.S.A., 23pp.

Lynch, D.R. and F.E. Werner (1991). Three-dimensional hydrodynamics on finite elements. Part II: Nonlinear time-stepping model. *Int. J. for Numerical Methods in Fluids* **12**, 507-533.

Lynch, D.R., F.E. Werner, D.A. Greenberg, J.W. Loder (1992). Diagnostic model for baroclinic, wind-driven and tidal circulation in shallow seas. *Continental Shelf Research* **12**, 1, 37-64.

Lynch, D.R. and C.E.Naimie (1993). The M2 tide and its residual on the Outer Banks of the Gulf of Maine. *J. Phys. Oceanogr.* (in press).

Wolff, E.M. (1993) *CONVCODES Users' Manual.* Dartmouth College Numerical Methods Laboratory Report, No. NML-93-(in preparation), Hanover, New Hampshire, U.S.A.

APPENDIX A

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

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

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

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

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, with components u and v
$W(x,y,z)$ 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 $(\partial/\partial x, \partial/\partial y)$
$h\mathbf{\Psi}(x,y)$ 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

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

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:[5]

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

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

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

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

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

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 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 $\nu^{\pm}(-h)$:

[5]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.

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

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.[6]

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

$$\begin{aligned} k\mathbf{V}(-h) = &\left(\frac{\tau^{+} + \tau^{-}}{2}\right)h\bar{\mathbf{V}} - j\left(\frac{\tau^{+} - \tau^{-}}{2}\right)\hat{\mathbf{z}} \times h\bar{\mathbf{V}} \\ &-\left(\frac{\alpha^{+} + \alpha^{-}}{2}\right)h\mathbf{\Psi} + j\left(\frac{\alpha^{+} - \alpha^{-}}{2}\right)\hat{\mathbf{z}} \times h\mathbf{\Psi} \end{aligned}$$

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

$$-\left(\beta^+ + \beta^-\right)h\hat{\mathbf{x}} + j\left(\beta^+ - \beta^-\right)h\hat{\mathbf{y}} \tag{29}$$

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', β',Ψ' and $\bar{\mathbf{R}}'$ all contain contributions from the bottom stress:

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

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

$$\beta' = \hat{\mathbf{x}}(\beta^+ + \beta^-) - j\hat{\mathbf{y}}(\beta^+ - \beta^-) \tag{33}$$

$$\mathbf{\Psi}' = \mathbf{\Psi}\left[1 + \left(\frac{\alpha^+ + \alpha^-}{2}\right)\right] - j\left(\frac{\alpha^+ - \alpha^-}{2}\right)\hat{\mathbf{z}} \times \mathbf{\Psi} \tag{34}$$

$$\bar{\mathbf{R}}' = \bar{\mathbf{R}} + \beta' \tag{35}$$

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', β',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:

$$\bar{\mathbf{V}} = \left(\frac{(j\omega + \tau')(\mathbf{G} + \mathbf{\Psi}' + \bar{\mathbf{R}}') - \mathbf{f}' \times (\mathbf{G} + \mathbf{\Psi}' + \bar{\mathbf{R}}')}{(j\omega + \tau')^2 + f'^2}\right) \tag{36}$$

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

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

$$-\oint h\bar{\mathbf{V}} \cdot \hat{\mathbf{n}}\phi_i\, ds + \left\langle\left[\frac{(j\omega + \tau')h(\mathbf{\Psi}' + \bar{\mathbf{R}}') - \mathbf{f}' \times h(\mathbf{\Psi}' + \bar{\mathbf{R}}')}{(j\omega + \tau')^2 + f'^2}\right] \cdot \nabla\phi_i\right\rangle \tag{37}$$

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 was 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', $\mathbf{\Psi}'$ 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:

$$\zeta(x,y) = \sum_j \zeta_j \phi_j(x,y) \tag{38}$$

we obtain from (37) the matrix equation

$$\begin{aligned}
[A]\{\zeta\} &= \{B\} - \{F\} \\
A_{ij} &= \langle j\omega\phi_j\phi_i\rangle + \left\langle \left[\frac{(j\omega+\tau')gh\nabla\phi_j - \mathbf{f}'\times gh\nabla\phi_j}{(j\omega+\tau')^2+f'^2}\right]\cdot\nabla\phi_i \right\rangle \\
B_i &= \left\langle \left[\frac{(j\omega+\tau')h(\mathbf{\Psi}'+\bar{\mathbf{R}}') - \mathbf{f}'\times h(\mathbf{\Psi}'+\bar{\mathbf{R}}')}{(j\omega+\tau')^2+f'^2}\right]\cdot\nabla\phi_i \right\rangle \\
F_i &= \oint h\bar{\mathbf{V}}\cdot\hat{\mathbf{n}}\phi_i\, ds
\end{aligned} \tag{39}$$

In the present implementation all inner products are evaluated numerically, with quadrature points at the nodes of the triangles. Quantities in (39) are expanded in a nodal basis, except for the $\bar{\mathbf{R}}$ part of $\bar{\mathbf{R}}'$, which is treated as an element based quantity. 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:

$$h\bar{\mathbf{V}}\cdot\hat{\mathbf{n}} = \frac{h}{f}(\mathbf{G}+\mathbf{\Psi}'+\bar{\mathbf{R}}')\cdot\hat{\mathbf{t}} \tag{40}$$

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 $(\mathbf{\Psi}'+\bar{\mathbf{R}}')\cdot\hat{\mathbf{t}}$ 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 $[A]$:

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

$$B'_{ij} = B_{ij} - \int \frac{h}{f} (\mathbf{\Psi}' + \bar{\mathbf{R}}') \cdot \hat{\mathbf{t}} \, \phi_i \, ds \tag{41}$$

with $\int ds$ indicating integration over the Type III boundary only. The integrals in (41) are integrated exactly over the boundary segment. Quantities in (41) are expanded in a nodal basis, except for the $\bar{\mathbf{R}} \cdot \hat{\mathbf{t}}$ part of $\bar{\mathbf{R}}' \cdot \hat{\mathbf{t}}$, which is treated as an element based quantity.

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

$$\sum_j \langle \phi_i \phi_j \rangle \mathbf{G_j} = -\langle g \nabla \zeta \phi_i \rangle \tag{42}$$

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 Lynch and Werner (1991) (herein, LW91). 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 $\Delta\sigma$.

In LW91 the continuity equation was used directly to solve for the vertical velocity. As a first-order equation, it admits only one boundary condition. LW91 applied a kinematic condition at the bottom, which resulted in a biased accumulation of error at the free surface. Herein we solve instead the differentiated form, as discussed in Lynch and Naimie (1993)

$$\frac{\partial^2 W}{\partial z^2} = -\frac{\partial}{\partial z} (\nabla \cdot \mathbf{V}) \tag{43}$$

A Galerkin implementation of 42 is used, with Dirichlet conditions enforced at the bottom and top.

APPENDIX B

NUMERICAL METHODS LABORATORY
DATA FILE STANDARDS FOR THE GULF OF MAINE PROJECT

NUMERICAL METHODS LABORATORY

DATA FILE STANDARDS FOR THE GULF OF MAINE PROJECT

Christopher E. Naimie

March 29, 1993[7]

Units

All standard data files will be written in MKS units, unless explicitly declared otherwise.

When spherical coordinates are used, the right-handed $(\lambda, \phi, z) = (longitude, latitude, z)$ coordinate system will be used, with (λ, ϕ) defined in fractional degrees. Longitude is relative to Greenwich, positive east, and is therefore *negative* in North America; latitude is relative to the equator, positive north.

(x, y) coordinates will be derived from (λ, ϕ) via Mercator projection, centered at the Boston tide gage $(\lambda, \phi) = (-71.03^o, 42.35^o)$. The earth's radius for this purpose is $R = 6.3675 \times 10^6\, m$.

Tidal phase ϕ will be reported as phaselag, in degrees, relative to Greenwich.

Tidal frequencies ω will be reported in radians/sec (i.e. MKS). For example, the M2 frequency based on 12.42 hour period is $\omega = 1.4053 \times 10^{-4} sec^{-1}$.

Frequency - Time Domain

In reconstructing time series, the formula is:

$$f(t) = a\, cos(\omega t - \frac{\pi}{180}\phi)$$

with a the real-valued amplitude. Equivalently, in complex exponential form,

$$f(t) = Re(A\, e^{j\omega t})$$

with the complex amplitude $A = a\, e^{-(\frac{j\pi}{180}\phi)}$ and $j = \sqrt{-1}$

File Format/Naming

All files will be machine-readable in ASCII (*) format, unless otherwise indicated.

NML standard data files for the Gulf of Maine project are designated by descriptive three-character trailing sequences that are preceded by a period (.).

[7] This is an update of NML memo (DRL,11/11/91)

Cartesian Geometry Files

Several standard files will be used to define only geometry. These will all share a common meshname, with a three-character trailing sequence identifying the type of file. For example, the file "**gom1.nod**" would indicate a node file for the gom1 mesh.

meshname.README: Self-explanatory.

meshname.nod: The **.nod** file contains (x,y) pairs for all nodes in a given mesh. There is no header; the file contains one line of the following form for each node I in the mesh.

I   X(I)   Y(I)

meshname.ele: The **.ele** file contains triangle incidence lists for all elements in a given mesh. There is no header; the file contains one line of the following form for each element L in the mesh. The incidence list is required to be in counterclockwise order.

L   IN(1,L)   IN(2,L)   IN(3,L)

meshname.bat: The **.bat** file contains bathymetry data for all nodes in a given mesh. Bathymetric depth is positive. There is no header; the file contains one line of the following form for each node I in the mesh.

I   H(I)

meshname.lnd: The **.lnd** "land node" file is used for display purposes only. Together with the **.lel** "land element" file, it describes a mesh of triangles covering the land areas adjacent to the relevant hydrodynamic mesh and conforming to it along the land-water interface. The **.lnd** file contains (x,y) pairs for all nodes in a given land mesh. There is no header. The format is identical to the **.nod** format.

meshname.lel: The **.lel** "land element" file contains triangle incidence lists for all elements in a given land mesh. There is no header. The format is identical to the **.ele** format.

meshname.nei: The **.nei** file conforms to TRIGRID specifications. There are three header lines:

line 1: the total number of nodes
line 2: the maximum number of neighbors for any node (MX below)
line 3: the range of the (x,y) data in the file: xmax, ymax, xmin, ymin
Following these lines, there are NN lines of the form:

I X(I) Y(I) NCD(I) H(I) [NGH(I,J), J=1,MX]

where: X(I),Y(I) are the cartesian coordinates of node I
NCD(I) is the node type code of node I
H(I) is the bathymetric depth at node I
NGH(I,J) are the nodes in the mesh that are "neighbors" to node I
I indexes from 1 to NN, where NN is the number of horizontal nodes

In the FUNDY4 Users' Manual (Lynch, 1990), unique node type codes from 0 to 11 are defined, reflecting various physical boundary conditions. For the purposes of describing geometry, only three codes are relevant:

0: interior
1: exterior boundary
2: island boundary

All publicly-available .nei files in Dartmouth directories will be restricted to these values of NCD(I), preserving their universality as geometry files. (See the description of **.bel** files below.) The program CONVCODES is available for converting between file types.

meshname.gr2: A special geometry file with various node and element connectivities pre-computed, for use with the DROG3D simulator (Blanton (1992)). These are derived from the above files by the program CONNECT2D. They contain only geometry data.

Spherical Geometry Files

Several additional standard files contain the equivalent geometry in (λ, ϕ) coordinates. These are indicated by the trailing characters "**ll**". For example, "**gom1ll.nod**" indicates a node file in spherical coordinates for the gom1 mesh.

meshnamell.nod: spherical equivalent of meshname.nod

meshnamell.lnd: spherical equivalent of meshname.lnd

meshnamell.nei: spherical equivalent of meshname.nei

meshnamell.gr2: spherical equivalent of meshname.gr2

Boundary Condition Files

These files indicate physical/dynamic conditions to be enforced on a given mesh. There will be several such files for a given geometry, each representing different combinations of physical forcing at the boundaries.

.bel: A boundary element description of the boundary conditions. There is a two-line header:

line 1: the geometric meshname
line 2: arbitrary text identifying the contents of the file
Following these lines, there are NBE lines of the form:

L IN(1,L) IN(2,L) IBC(left) IBC(right)

where: element L is a line segment beginning at node IN(1,L) and ending at node IN(2,L)
IBC(1) indicates the nature of the boundary on the left of the element
IBC(2) indicates the nature of the boundary on the right of the element
IN(I,L) is a conventional incidence list for 1-D, linear boundary elements
L indexes from 1 to NBE, where NBE is the number of boundary elements

The node numbers refer to the nodes in the triangular mesh file **meshname.nod**, identified in the first header line. The boundary codes are as given in the FUNDY4 Users' Manual (Lynch, 1990):

0: interior
1: land
2: island
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

Only codes 0-5 are required to describe boundary elements; the additional corner codes are needed only for boundary node classification; they never appear in **.bel** files.

Input Files

.inp: An input file for FUNDY5. This file contains both geometric and boundary condition information, sufficient to obtain a hydrodynamic solution with FUNDY5. It is written by the program TRIGTOFUNDY4 or CONVCODES, as described in the FUNDY5 Users' Manual.

.din: An input file for DROG3D. This file contains sufficient information to open a velocity output **.vel** file, which in turn contains the geometry meshname, initial positions for one or more drogues, and time-integration parameters for drogue trajectories. It is described in the DROG3D Users' Manual (Blanton (1992)). **Note: time integration parameters in this file are to be given in hours**, an important exception to the general MKS standard.

FUNDY5 Density Data Files

.lst: A 3-D file which contains level surface σ_t density data. There are 4 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: NLEV, the number of vertical levels for which σ_t data exists.
line 4: The height of each level surface, from the bottom to the top.

Z(1) Z(2) Z(3) . . . . . Z(NLEV)

Following these lines, there are NN*NLEV lines of the form:

σ_t(I,L)

where: σ_t is the density at horizontal node I, vertical level L (sigmat units)
the inner loop is over 1-D vertical levels; L=1,NLEV
the outer loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the 2-D horizontal mesh

.rho: A 3-D file which contains the baroclinic forcing data required by FUNDY5.

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: NNV, the number of vertical sigma mesh nodes under each horizontal 2-D node
Following these lines, there are NN*NNV lines of the form:

I Z(I,J) RHOX(I,J) RHOY(I,J)

where: $\text{RHOX(I,J)} = -\frac{g}{\rho_{ref}} \int_{Z(I,J)}^{0} \frac{\partial \rho}{\partial x} dz$

$\text{RHOY(I,J)} = -\frac{g}{\rho_{ref}} \int_{Z(I,J)}^{0} \frac{\partial \rho}{\partial y} dz$

the inner loop is over 1-D vertical nodes; J=1,NNV
the outer loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the 2-D horizontal mesh

Following line: elemental values of hrbar: k,hrbarelx,hrbarely
Following NE lines:

K HRBARELX(K) HRBARELY(K)

where: $\text{HRBARELX(K)} = -\frac{g}{\rho_{ref}} \int_{Z(K,1)}^{0} [\int_{Z(K,J)}^{0} \frac{\partial \rho}{\partial x} dz] dz$

$\text{HRBARELY(K)} = -\frac{g}{\rho_{ref}} \int_{Z(K,1)}^{0} [\int_{Z(K,J)}^{0} \frac{\partial \rho}{\partial y} dz] dz$

the loop is over 2-D horizontal elements; K=1,NE
NE = the number of elements in the 2-D horizontal mesh

Result Files Conforming to DROG3D Standards

.vel: A 3-D complex velocity file describing a velocity output in harmonic form. This file conforms to DROG3D input standards (See Blanton (1992)). **Note that phaselag in .vel files is in radians.** There are 4 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: NNV, the number of vertical sigma mesh nodes under each horizontal 2-D node
line 4: the number of harmonic constituents contained in the file
Following these lines there are 1 or more blocks containing 3-D velocity amplitude and phase results for a particular harmonic constituent plus its frequency:

Following line: frequency (radians/second)
Following NN*NNV lines:

I Z(I,J) Uamp(I,J) Upha(I,J) Vamp(I,J) Vpha(I,J) Wamp(I,J) Wpha(I,J)

where: Z(I,J) is the vertical coordinate at horizontal node I, vertical node J (MKS)
(Uamp,Vamp,Wamp) are amplitudes of the (x,y,z) velocities (MKS)
(Upha,Vpha,Wpha) are the phaselags of the (x,y,z) velocities (radians)
the inner loop is over 1-D vertical nodes; J=1,NNV
the outer loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the horizontal 2-D mesh

.pth: An output file from DROG3D, describing the trajectories of one or more drogues. There are several header lines:

line 1: the geometric meshname where the relevant **.nod, .ele, .bat, .gr2, etc.** files can be found. For example, "gom1".

lines 2ff: an echo of the **.din** file which controlled the execution of DROG3D and the writing of the present **.pth** file. This echo begins with the **.vel** filename used in the computation of the trajectories.

Following this echo, a separate line with the characters "XXXX" indicates the end of the header section. The balance of the **.pth** file contains the trajectory information as described in Blanton (1992).

Other Standard Result Files

.v2r: A 2-D, real-valued horizontal velocity field, sampled for example at the surface or the bottom of a 3-D mesh. There are 2 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
Following these lines, there are NN lines of the form:

I U(I) V(I)

where: (U,V) are the (x,y) components of horizontal velocity (MKS)
the loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the horizontal 2-D mesh

.v2c: A 2-D, complex-valued horizontal velocity field, sampled for example at the surface or the bottom of a 3-D mesh. There are 3 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: frequency (radians/second)
Following these lines, there are NN lines of the form:

I Uamp(I) Upha(I) Vamp(I) Vpha(I)

where: (Uamp,Vamp) are the amplitudes of the (x,y) velocities (MKS)
(Upha,Vpha) are the phaselags of the (x,y) velocities (degrees)
the loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the 2-D horizontal mesh

.s2r: A 2-D, real-valued scalar field (e.g. tidal amplitude). There are 2 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
Following these lines, there are NN lines of the form:

I S(I)

where: S(I) is the scalar value at node I (MKS)
the loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the horizontal 2-D mesh

.s2c: A 2-D, complex-valued scalar field (e.g. tidal amplitude and phase). There are 3 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: frequency (radians/second)
Following these lines, there are NN lines of the form:

I Samp(I) Spha(I)

where: Samp(I) is the amplitude at node I (MKS)
Spha(I) is the phaselag at node I (degrees)
the loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the 2-D horizontal mesh

.s3r: A 3-D, real-valued scalar field (e.g. density). There are 3 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: NNV, the number of vertical sigma mesh nodes under each horizontal 2-D node
Following these lines, there are NN*NNV lines of the form:

I Z(I,J) S(I,J)

where: Z(I,J) is the vertical co-ordinate at horizontal node I, vertical node J (MKS)
S(I,J) is the scalar value at horizontal node I, vertical node J (MKS)
the inner loop is over 1-D vertical nodes; J=1,NNV
the outer loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the 2-D horizontal mesh

.s3c: A 3-D, complex-valued scalar field (e.g. pressure amplitude and phase). There are 4 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: NNV, the number of vertical sigma mesh nodes under each horizontal 2-D node
line 4: frequency (radians/second)
Following these lines, there are NN*NNV lines of the form:

I Z(I,J) Samp(I,J) Spha(I,J)

where: Z(I,J) is the vertical coordinate at horizontal node I, vertical node J (MKS)
Samp(I,J) is the amplitude at horizontal node I, vertical node J (MKS)
Spha(I,J) is the phaselag at horizontal node I, vertical node J (degrees)
the inner loop is over 1-D vertical nodes; J=1,NNV
the outer loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the 2-D horizontal mesh

.v3r: A 3-D, real-valued velocity field. There are 3 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: NNV, the number of vertical sigma mesh nodes under each horizontal 2-D node
Following these lines, there are NN*NNV lines of the form:

I Z(I,J) U(I,J) V(I,J) W(I,J)

where: Z(I,J) is the vertical coordinate at horizontal node I, vertical node J (MKS)
(U,V,W) are the real (x,y,z) vector components of the 3-D velocity
the inner loop is over 1-D vertical nodes; J=1,NNV
the outer loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the 2-D horizontal mesh

.v3c: A 3-D, complex-valued vector field. There are 4 header lines:

line 1: the geometric meshname
line 2: reserved for user's description of the file
line 3: NNV, the number of vertical sigma mesh nodes under each horizontal 2-D node
line 4: frequency (radians/second)
Following these lines, there are NN*NNV lines of the form:

I Z(I,J) Uamp(I,J) Upha(I,J) Vamp(I,J) Vpha(I,J) Wamp(I,J) Wpha(I,J)

where: Z(I,J) is the vertical coordinate at horizontal node I, vertical node J (MKS)
(Uamp,Vamp,Wamp) are (x,y,z) velocity amplitudes (MKS)
(Upha,Vpha,Wpha) are (x,y,z) phaselags (degrees)
the inner loop is over 1-D vertical nodes; J=1,NNV
the outer loop is over 2-D horizontal nodes; I=1,NN
NN = the number of nodes in the 2-D horizontal mesh

Note - The **.v3c** format has phase in degrees while the related **.vel** files are in radians.

References

Blanton, B. (1992). *DROG3D.f - User's Manual for 3-Dimensional Drogue Tracking on a Finite Element Grid with Linear Finite Elements.* Skidaway Institute of Oceanography,Savannah, Georgia, U.S.A., July 1992.

Lynch, D.R. (1990). *Three-Dimensional Diagnostic Model for Baroclinic, Wind-Driven and Tidal Circulation in Shallow Seas; FUNDY4 User's Manual.* Dartmouth College Numerical Methods Laboratory Report, No. NML-90-2, Hanover, New Hampshire, U.S.A., 23pp.

Lynch, D.R. (1991) Previous version of this memo dated 11/11/91.

Wolff, E.M. (1993) *CONVCODES Users' Manual.* Dartmouth College Numerical Methods Laboratory Report, No. NML-93-(in preparation), Hanover, New Hampshire, U.S.A.

APPENDIX C

DIFFERENCES BETWEEN FUNDY4 AND FUNDY5

DIFFERENCES BETWEEN FUNDY4 AND FUNDY5

Though a variety of changes were made to improve the code, the implementation of FUNDY5 is very similar to FUNDY4. The input file, Subroutine BC and Subroutine ATMOS have not changed with this revision; however, User Subroutines VERTGRID and OUTPUT must be replaced with the newer versions named VERTGRID5 and OUTPUT5. The changes made range from computational differences to enhance the solution and code restructuring to increase code efficiency to the inclusion of additional subroutines to reduce the requirement for variants to the software.

- Computational differences between FUNDY5 and FUNDY4:
  1. FUNDY5 uses elemental values of $h\overline{\mathbf{R}}$ in the computation of the RHS of the elevation diffusion equation - FUNDY5 uses nodal values.
  2. FUNDY5 uses $\frac{\partial}{\partial z}$(continuity equation) to compute the vertical velocities - FUNDY4 uses the continuity equation alone.
  3. FUNDY5 uses vertical averaging to compute the depth-averaged horizontal velocities - FUNDY4 computes them using Equation (36) of Appendix A.
  4. FUNDY5 uses the include file FUNDY5ARRAYS.DIM to set parameters - FUNDY4 requires them to be set within the main routine.
  5. FUNDY5 employs the Geostrophic Boundary Condition using elemental baroclinic forcing and exact integration - FUNDY4 employs this BC using nodal baroclinic forcing and nodal quadrature.
  6. FUNDY5 does not compute the 3-D pressure field - FUNDY4 does.
  7. FUNDY5 defaults required elevation Boundary Conditions to zero - FUNDY4 does not.
  8. FUNDY5 writes a revised echo file which includes a report of the agreement between the Boundary Condition specification in the **.inp** file and Subroutine BC - FUNDY4 does not report the results of these checks.
- In the transition from FUNDY4 to FUNDY5, the following changes were made to increase code efficiency:
  1. Unnecessary arrays were eliminated.
  2. The main program was modularized into subroutines.
  3. A transposed complex matrix solver [Subroutine CSOLVET] was implemented.
  4. A more efficient sparse matrix builder [Subroutine SPRSBLD1] was implemented.
- Additional subroutines have been added to FUNDY5 to reduce the requirement for variants of the software. These subroutines are designed to be called from Subroutine VERTGRID5. They are described in the USER SUBROUTINE section of this manual.
  1. Subroutine SINEGRID5
  2. Subroutine RHOGRID5
  3. Subroutine BAROCLINIC5