A PRIMITIVE EQUATION, 3-DIMENSIONAL MODEL OF THE OCEAN

Michael D. Cox

GFDL Ocean Group Technical Report No. 1

August 30, 1984

Geophysical Fluid Dynamics Laboratory/NOAA

Princeton University

Princeton, NJ 08542

TABLE OF CONTENTS

A. Introduction

More than two decades have passed since the first primitive equation, 3-dimensional numerical ocean model was coded for use in studying the most basic aspects of large-scale, baroclinic ocean circulation (Bryan and Cox, 1967). A description of the physics and numerics involved was published by Bryan (1969). In this model the prediction of currents is carried out using the Navier-Stokes equations with three basic assumptions. The Boussinesq approximation is adopted, in which density differences are neglected except in the buoyancy term. The hydrostatic assumption is made in which local acceleration and other terms of equal order are eliminated from the equation of vertical motion. And, lastly, closure is attained by adopting the turbulent viscosity hypothesis in which stresses exerted by scales of motion too small to be resolved by the grid are represented as an enhanced molecular mixing. The temperature and salinity are calculated using conservation equations, again utilizing a turbulent mixing hypothesis for closure. The equations are linked by a simplified equation of state.

Several techniques are used for the purpose of computational efficiency. High speed, external gravity waves are eliminated by the "rigid-lid" approximation, and a LaPlacian equation is solved for the non-divergent, vertically averaged flow. The next most serious timestep limitation, the half pendulum day constraint associated with inertia-gravity waves, is overcome by a semi-implicit treatment of the Coriolis term.

Considerable improvement was made to the structure of the FORTRAN code of this model by Semtner (1974) who, at the same time, added various features to the mathematical formulation, chief of which was the use of "hole relaxation" (Takano, 1974) in the solution of the external mode for a model with islands. This version of the model has been adopted by many investigators and has seen considerable use for a number of years in the ocean modelling work at GFDL. During this time, as vector processing machines became more demanding of suitable FORTRAN structure, significant changes have been made to the code for efficiency purposes. It has also been generalized in several ways, among which is the incorporation of variable grid spacing in the horizontal, and an arbitrary number of tracer prognostic variables. The relaxation code for the solution of the external mode has been redesigned, and a better technique for establishing the initial guess has reduced the scans-to-convergence considerably.

While many alterations have been made to the original Semtner code, anyone who is familiar with that code will note that the basic structure remains the same. It is the goal here to provide improvements while retaining as much continuity with the former version as possible.

The present code was optimized primarily for the CYBER 205 s tem. This machine yields a great payoff in execution speed for la vector lengths, and the basic design philosophy has been to prod long vectors to the greatest degree possible. At the same time, r sonably good efficiency should be realized on virtually any machi and this consideration has been made in the design as well.

Two objectives have been sought in designing the code for outside of GFDL. First, it has been made universal to some degree, the use of optional code lines. Secondly, it has been given the p sibility of future expansion and improvement by incorporating l sequence numbers. A separate, FORTRAN coded updating utility is p vided to carry out these operations (see section H).

In the interests of the greater ocean modelling community, it our wish that users of the code will be willing to share alterati and improvements which they have developed and find helpful. The companying updating system has been designed with that possibility mind. The staff at GFDL will act as disseminators of such informat if we are provided with such update sets. An example which will available at GFDL soon is an update set to provide 4th order, bih monic mixing. Periodic communications will be mailed to all kn users of the code describing available update sets.

Bryan, K., and M. D. Cox, 1967: A numerical investigation of the oceanic general circulation. Tellus, 19, 54-80.

Bryan, K., 1969: A numerical method for the study of the circulation of the World Ocean. J. Computat. Phys., 4, 347-376.

Semtner, A. J., 1974: An oceanic general circulation model with bottom topography. UCLA Dept. of Meteorology Tech. Rep. No. 9, 99 pp.

Takano, K., 1974: A general circulation model for the World Ocean. UCLA Dept. of Meteorology Tech. Rep. No. 8.

B. Continuous Equations of the Model

The basic equations of the model as described above are written here in continuous form. Let

$$
\begin{aligned}
m &= \sec\phi \\
n &= \sin\phi \\
f &= 2\Omega\sin\phi \\
u &= a\dot{\lambda}/m \\
v &= a\dot{\phi}
\end{aligned}
\qquad (1)
$$

where ϕ is latitude, λ is longitude and a is the radius of the earth. An advective operator,

$$\Gamma(\mu) = ma^{-1}[(u\mu)_\lambda + (v\mu m^{-1})_\phi] + (w\mu)_z \qquad (2)$$

is adopted in which μ is any scalar quantity. The equations of motion are then

$$u_t + \Gamma(u) - fv = -ma^{-1}(p/\rho_o)_\lambda + F^u \qquad (3)$$

$$v_t + \Gamma(v) + fu = -a^{-1}(p/\rho_o)_\phi + F^v \qquad (4)$$

where ρ_o is taken to be unity. The local pressure, p, is given by the hydrostatic relation,

$$p(z) = p^s + \int_z^0 g\rho dz \qquad (5)$$

where p^s is the pressure at the surface of the ocean. The continuity equation is

$$\Gamma(1) = 0. \qquad (6)$$

The conservation equation,

$$T_t+\Gamma(T) = F^T \tag{7}$$

applies to any "tracer" type of quantity carried in the model. These include the active tracers, potential temperature and salinity (active in the sense that they appear in the equation of state), and any passive tracers such as Carbon 14 or Tritium. The equation of state is

$$\rho = \rho(\theta,S,z) \tag{8}$$

where θ is potential temperature, S is salinity and the depth dependence arises from compression effects. In the present model, (8) is represented by a polynomial fit to the Knudsen formula (Bryan and Cox, 1972, see section J). Let

$$\nabla^2\mu = m^2\mu_{\lambda\lambda}+m(\mu_\phi/m)_\phi. \tag{9}$$

Then the effects of turbulent mixing are

$$F^u = A_{MV}u_{zz}+A_{MH}a^{-2}[\nabla^2u+(1-m^2n^2)u-2nm^2v_\lambda] \tag{10}$$

$$F^v = A_{MV}v_{zz}+A_{MH}a^{-2}[\nabla^2v+(1-m^2n^2)v+2nm^2u_\lambda] \tag{11}$$

$$F^T = [(A_{TV}/\delta)T_z]_z+A_{TH}a^{-2}\nabla^2T \tag{12}$$

where A_{ab} is the mixing coefficient corresponding to

a:
- M momentum
- T tracer

b:
- V vertical
- H horizontal.

Vertical mixing is known to be a rather complex function of vertical stability in nature. Since this process is still not well understood, we have adopted a simple, uniform mixing under statically stable conditions, and an infinite mixing under statically unstable conditions. If ρ''_z is the local vertical density gradient (absent of compression effects), then

$$\delta = \begin{cases} 1 & \rho''_z < 0 \\ 0 & \rho''_z > 0. \end{cases} \qquad (13)$$

At lateral walls, the boundary conditions are

$$u, v, T_n = 0 \qquad (14)$$

where the n subscript indicates a local derivative with respect to the coordinate normal to the wall. At the surface,

$$\begin{aligned} \rho_o A_{MV}(u_z, v_z) &= \tau^\lambda, \tau^\phi \\ A_{TV}(T_z) &= \eta \qquad\qquad z = 0 \\ w &= 0. \end{aligned} \qquad (15)$$

The "rigid-lid" assumption of zero vertical motion at the surface filters out high speed external gravity waves which would otherwise seriously limit the length of the time step of the numerical integration. The quantities τ^λ, τ^ϕ are the zonal and meridional components of the surface stress, and η is a flux through the surface, of the particular tracer involved. At the bottom,

$$\begin{aligned} \rho_o A_{MV}(u_z, v_z) &= \tau_B{}^\lambda, \tau_B{}^\phi \\ T_z &= 0 \qquad\qquad z = -H \\ w &= -mua^{-1}H_\lambda - va^{-1}H_\phi. \end{aligned} \qquad (16)$$

where $\tau_B{}^\lambda$ and $\tau_B{}^\phi$ are bottom stresses.

Combining (3) and (4) with (5),

$$u_t = u_t^{\prime} - ma^{-1}p_\lambda^s; \quad v_t = v_t^{\prime} - a^{-1}p_\phi^s \tag{17}$$

where

$$u_t^{\prime} = -\Gamma(u)+fv-mga^{-1}\int_z^0 \rho_\lambda dz' + F^u \tag{18}$$

$$v_t^{\prime} = -\Gamma(v)-fu- \quad a^{-1}\int_z^0 \rho_\phi dz' + F^v. \tag{19}$$

Let us define

$$u = \hat{u}+\overline{u}; \quad v = \hat{v}+\overline{v} \tag{20}$$

where

$$\overline{\mu} = H^{-1}\int_H^0 \mu dz \tag{21}$$

Then, since p^s is not a function of depth,

$$\hat{u}_t = u_t^{\prime}-\overline{u_t^{\prime}}; \quad \hat{v}_t = v_t^{\prime}-\overline{v_t^{\prime}}. \tag{22}$$

Since all terms on the right of (18) and (19) are known, (22) may be solved for the internal modes of momentum. Under the rigid-lid boundary condition, the external mode of momentum may be represented by a volume transport stream function, ψ,

$$\overline{u} = -(aH)^{-1}\psi_\phi; \quad \overline{v} = m(aH)^{-1}\psi_\lambda. \qquad (\ldots)$$

This is shown by integrating (6) vertically, substituting (23) and noting that the boundary conditions (15) and (16) on w are satisfied. A prognostic equation for Ψ may be obtained by averaging (17) vertically, and eliminating terms in p^* by applying the curl operator,

$$\mathrm{curl}_z(\bar{v}_t,\bar{u}_t) = ma^{-1}[\bar{v}_{t\lambda}-(\bar{u}/m)_{t\phi}]. \qquad (24)$$

Substituting (23),

$$[m\Psi_{t\lambda}/(Ha^2)]_\lambda+[\Psi_{t\phi}/(mHa^2)]_\phi = a^{-1}[\overline{v_t'}_\lambda-(\overline{u_t'}/m)_\phi]. \qquad (25)$$

The boundary condition on Ψ at lateral walls, corresponding to (14) is

$$\Psi_\phi = \Psi_\lambda = 0. \qquad (26)$$

This condition is satisfied by setting Ψ constant over each unconnected land mass comprising the ocean boundary. In the case of an enclosed basin with no islands, Ψ may be arbitrarily set to zero over the boundary forming land mass. If, in addition, islands are present, the associated constant for each island reflects the net flow around the island and must therefore be predicted by the governing equations. The method used is "hole relaxation" in which the line integral of the quantity ∇p^*, taken around the island, is required to vanish. Averaging (17) vertically, integrating around the coast of the island and setting the contribution due to p^* to zero, the predictive equation,

$$\oint[(m\Psi_{t\lambda}/H)d\phi-(\Psi_{t\phi}/(mH))d\lambda] = a\oint[\overline{v_t'}d\phi+(\overline{u_t'/m})d\lambda] \qquad (27)$$

is obtained. Applying the Stokes theorem yields a more useful form,

$$a^{-2}\int_A[(m\Psi_{t\lambda}/H)_\lambda+(\Psi_{t\phi}/(mH))_\phi]dA = a^{-1}\int_A[\overline{v_t'}_\lambda-(\overline{u_t'/m})_\phi]dA. \qquad (28)$$

Note that (28) is simply an area integral of (25), taken over the island.

C. Finite Difference Formulation

The initial value problem described in Section II by prognostic equations (7), (22), (25), associated diagnostic equations (6), (8), (23), and boundary conditions (14), (15), (16), (26), (28) may be solved numerically using finite difference techniques. Rewriting the continuous equations in finite difference form may be done in any of several different ways. However, it is of critical importance that certain integral constraints be maintained during the solution of the initial value problem, and these constraints dictate the particular finite difference formulation which is used. Arakawa (1966) did much of the early work along these lines and Bryan (1969) generalized it using the arguments summarized below.

Let the basin under consideration be divided into cells with interfaces lying along common planes of constant latitude, longitude, and depth. The first constraint which must be satisfied is that of mass conservation within each cell. If each cell interface is designated by the index b, with the area of interface b equal to A_b and the velocity normal to interface b into the cell equal to V_b, then

$$\sum_{b=1}^{6} V_b A_b = 0. \tag{29}$$

Secondly, the basin wide integral, I' of any conserved quantity q, must remain unchanged by the advective process. This assures conservation of momentum, heat, salt and other tracer quantities. If q_b is the value of q on interface b and there are a total of N cells in the (closed) basin, then

$$dI'/dt = -\sum_{n=1}^{N} \sum_{b=1}^{6} q_b V_b A_b = 0. \tag{30}$$

That the above integral vanishes in general can be seen by considering that each term involving interior interfaces appears twice once with a positive sign and once with a negative sign. The remaining, uncancelled terms at the boundaries are zero since V_b is zero there.

The third constraint is that the volume integral, of the square of q is unchanged by advection. This assures conservation of kinetic energy and the variance of temperature, salinity and other tracer quantities. If Q_n is the average of q within the cell, then

$$dI''/dt = -2\sum_{n=1}^{N} \sum_{b=1}^{6} q_b Q_n V_b A_b = 0. \tag{31}$$

The above integral is not zero for all definitions of q_b, although it has been shown (Arakawa, 1966) that it can be made to vanish by defining

$$q_b = (Q_b+Q_n)/2 \tag{32}$$

where Q_b is the average of q in the cell sharing interface b. The integral can be rewritten

$$dI''/dt = -\sum_{n=1}^{N}\left[Q_n^2\sum_{b=1}^{6}V_bA_b+\sum_{b=1}^{6}Q_nQ_bV_bA_b\right]. \tag{33}$$

The first term on the right vanishes by (29) and the second term vanishes by the same cancelling process which occurs in (30).

The fourth constraint which must be met by the finite difference equations is that kinetic energy gained (lost) through the pressure term of the momentum equations must be balanced by an equal loss (gain) of potential energy through the advection terms of the conservation equation for density. Note that, in actuality, there is no conservation equation for density itself, but only for the temperature and salinity separately. Therefore, when the equation of state, (8) is nonlinear, the following balance does not strictly hold. Multiplying the pressure terms by u and v, the density advection by gz, and integrating, we get, in the continuous form,

$$\int_V\left[-ma^{-1}u(p/\rho_o)_\lambda-va^{-1}(p/\rho_o)_\phi\right]dV = -\int_V gz\Gamma(\rho)dV. \tag{34}$$

Finally, since an insulating boundary condition exists for the tracer quantities, T, at all boundaries of the basin except the surface, the constraint,

$$\int_V F^T dV = \int_A \eta dA \tag{35}$$

must be met, where the integral on the right is taken over the surface area of the basin.

Let the cells described earlier be indexed such that the eastward position is given by i, the northward position by j and the downward position by k. The dimensions of each cell will be denoted by $\Delta_i\lambda$, $\Delta_j\Phi$ and $\Delta_k z$. The arrangement of variables within the cells corresponds to the "B-grid" configuration of Arakawa and Lamb (1977). Horizontally, the tracer quantities, T are situated in the centers of the cells with the horizontal velocity components placed at the corners as shown in Fig. 1a. The two-dimensional quantity, ψ is also positioned at the horizontal center of the cells. The vertical arrangement is illustrated in Fig. 1b. T, u and v are located halfway through the vertical dimension of the cell. Two sets of vertical velocities are calculated. The quantity w^T is used for computing T and w^V is used for computing u and v. In each case w is calculated at the horizontal interface of the cell, in vertical line with its associated prognostic variable, T, u, v.

It will be convenient to define the following finite difference operators:

$$\delta_\lambda(\mu_i) = (\mu_{i+1/2} - \mu_{i-1/2})/\Delta_i\lambda$$

$$\overline{\mu_i}^\lambda = (\mu_{i+1/2} + \mu_{i-1/2})/2$$

$$\text{Max}_\lambda(\mu_i) = \text{maximum of } (\mu_{i+1/2}, \mu_{i-1/2})$$

$$\text{Min}_\lambda(\mu_i) = \text{minimum of } (\mu_{i+1/2}, \mu_{i-1/2}).$$

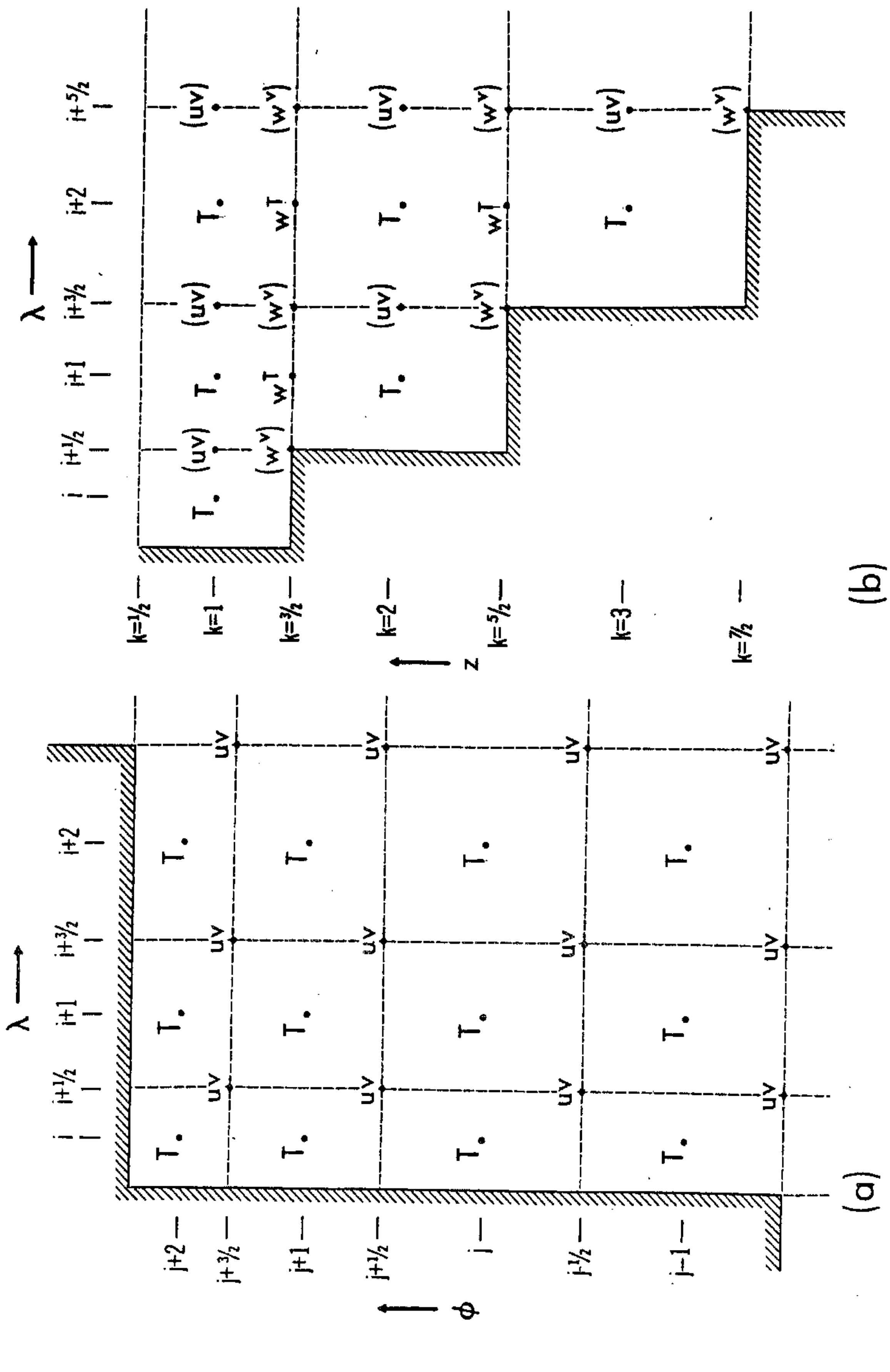
In the following discussion, the indices i,j will be used in some equations to denote the position of T and in other equations to denote the position of u,v. Whether i and j are full integers (T) or half integers (u,v) will be implied by the variable they index. Further, if indices are omitted, the values i,j,k are understood.

Based upon the previous description of the placement of variables within the cells, it is possible to define an alternate cell where the tracer quantities appear at the corners. The horizontal dimensions of such a cell would be

$$\Delta_i\lambda = \overline{\Delta_i\lambda}^\lambda; \quad \Delta_j\Phi = \overline{\Delta_j\Phi}^\Phi. \tag{36}$$

where, as stated above, the index of the term on the left is implied to be a half integer and that on the right, a full integer. Further, cells can be defined in which T, u and v appear at vertical interfaces. Their vertical dimension is

$$\Delta_k z = \overline{\Delta_k z}^z \tag{37}$$


(a)
(b)
λ
φ
z
uv
(uv)
(w^v)
w^T
T
k=1/2
k=1
k=3/2
k=2
k=5/2
k=3
k=7/2
i
i+1/2
i+1
i+3/2
i+2
i+5/2
j+2
j+3/2
j+1
j+1/2
j
j-1/2
j-1

where, again, the index on the left is implied to be a half integer. These cells are used later in forming volume integrals of quantities appearing at the horizontal or vertical velocity points.

From Fig. 1b it may be seen that the total depth of the basin H^V, defined at the corner points of the original cells is

$$H^V = \mathrm{Min}_\lambda[\mathrm{Min}_\phi(H^T)] \qquad (38)$$

or the minimum of the depths H^T, of the four surrounding vertical columns of cells. We can now define a finite difference equivalent of the vertical averaging operator (21),

$$\bar{\mu}^* = H^{-1}\sum_{k=1}^{K} \mu\Delta z. \qquad (39)$$

Rewriting (20), we get

$$u = \hat{u}+\bar{u}^*; \quad v = \hat{v}+\bar{v}^* \qquad (40)$$

where (23) is rewritten

$$\bar{u}^* = -(aH)^{-1}\delta_\phi\bar{\psi}^\lambda; \quad \bar{v}^* = m(aH)^{-1}\delta_\lambda\bar{\psi}^\phi. \qquad (41)$$

We will first develop the finite difference equations for the internal modes, followed by the equations for ψ and then the equations for the tracers, T.

Using centered differencing in time, (22) is written

$$\delta_t\overline{(\hat{u})}^t = \delta_t\overline{u'}^t - \delta_t\overline{\overline{u'}}^{*t} \; ; \; \delta_t\overline{(\hat{v})}^t = \delta_t\overline{v'}^t - \delta_t\overline{\overline{v'}}^{*t}. \qquad (42)$$

The Coriolis term is handled semi-implicitly in time so that the time-step need not be limited to a value small enough to resolve inertial oscillations. Equations (18) and (19) may be written

$$\delta_t \overline{u'}^{t} - f\alpha v'^{N+1} = G^u \qquad (43)$$

$$\delta_t \overline{v'}^{t} + f\alpha u'^{N+1} = G^v \qquad (44)$$

where N+1 indicates the time step being predicted and

$$G^u = -\Gamma^*(u) + f(1-\alpha)v^{N-1} - mga^{-1}\overline{\sum_{k'=1/2}^{k-1/2} \delta_\lambda \overline{\rho_{k'}}^{z} \overline{\Delta_{k'} z}^{z}}^{\phi} + F^{u*} \qquad (45)$$

$$G^v = -\Gamma^*(v) - f(1-\alpha)u^{N-1} - ga^{-1}\overline{\sum_{k'=1/2}^{k-1/2} \delta_\phi \overline{\rho_{k'}}^{z} \overline{\Delta_{k'} z}^{z}}^{\lambda} + F^{v*}. \qquad (46)$$

It may be shown that, for a value of $\alpha > .5$, the constraint of resolving inertial oscillations in time is removed. This is particularly useful in coarse resolution studies where long timesteps are otherwise possible.

Before defining the advective operator Γ^*, we construct two auxiliary quantities

$$u^* = \overline{\hat{u}}^{\lambda} - (\delta_\phi \Psi)/[aMax_\lambda(H)] \qquad (47)$$

$$(v/m)^* = \overline{\hat{v}/m}^{\phi} - (\delta_\lambda \Psi)/[aMax_\phi(H)]. \qquad (48)$$

Then,

$$\Gamma^*(\mu) = ma^{-1}\left[\delta_\lambda(u^*\overline{\mu}^{\lambda}) + \delta_\phi((v/m)^*\overline{\mu}^{\phi})\right] + \delta_z(w^v\overline{\mu}^{z}) \qquad (49)$$

and w^v is defined by

$$\Gamma^*(1) = 0 \qquad (50)$$

along with the boundary condition

$$w^{*}_{k=1/2} = 0. \qquad (51)$$

Constraints (29) and (30) are satisfied automatically by (50) and constraint (31) is satisfied by expressing the advected quantity, μ at the various interfaces, as the average of the neighboring values as in (32). The use of the special quantities (47) and (48) is needed so that the vertical velocity at the bottom of the basin is of the form of boundary condition (16).

The frictional terms are lagged one timestep for purposes of numerical stability (Richtmyer and Morton, 1967),

$$F^{u*} = A_{MV}\delta_z\delta_z u^{N-1} + A_{MH}a^{-2}[\nabla^{2*}u^{N-1} + (1-m^2n^2)u^{N-1} - 2nm^2\delta_\lambda v^{N-1}] \qquad (52)$$

$$F^{v*} = A_{MV}\delta_z\delta_z v^{N-1} + A_{MH}a^{-2}[\nabla^{2*}v^{N-1} + (1-m^2n^2)v^{N-1} + 2nm^2\delta_\lambda u^{N-1}] \qquad (53)$$

where

$$\nabla^{2*}(\mu) = m^2\delta_\lambda\delta_\lambda\mu + m\delta_\phi(m^{-1}\delta_\phi\mu). \qquad (54)$$

The finite difference form of (25) for the stream function, ψ is also complicated somewhat by the semi-implicit treatment of the Coriolis term. Bringing the implicit component to the left side,

$$\delta_\lambda[(m/(\overline{H}^{\phi}a^2))\delta_\lambda\delta_t\overline{\psi}^{t}] + \delta_\phi[(1/(m\overline{H}^{\lambda}a^2))\delta_\phi\delta_t\overline{\psi}^{t}]$$
$$+\delta_\phi\overline{[(\alpha f/(Ha^2))\delta_\lambda\overline{\psi^{N+1}}^{\phi}]\Delta\lambda}^{\lambda}/\Delta\lambda - \delta_\lambda\overline{[(\alpha f/(Ha^2))\delta_\phi\overline{\psi^{N+1}}^{\lambda}]\Delta\phi}^{\phi}/\Delta\phi$$
$$= a^{-1}[\delta_\lambda\overline{(\overline{G^{v}}^{*})\Delta\phi}^{\phi}/\Delta\phi - \delta_\phi\overline{(\overline{G^{u}}^{*}/m)\Delta\lambda}^{\lambda}/\Delta\lambda]. \qquad (55)$$

Boundary condition (26) is satisfied by setting ψ constant along the two rows of cells straddling the basin boundary. In the case of a singly connected basin, this value may be arbitrarily set at zero. For an island, the value must be calculated using a finite difference form of (28). This simply amounts to taking an area weighted sum of (55) over all cells for which ψ will take the island value.

The solution of the elliptic problem above is achieved by the method of successive over-relaxation. A guess is made for all cells variant in ψ, including the island constants, a residual is computed based on (55) and the stated boundary conditions, and a new guess is established from the residual. This process is iterated until the change in ψ between guesses is smaller than an arbitrary constant. Convergence is assured as long as the matrix of coefficients of ψ is diagonally dominant. Large gradients of bottom depth may destroy such dominance through the implicit Coriolis term of (55). However, this is not a problem for most bottom configurations used.

The predictive equation for the tracer quantities T may be written in finite difference form,

$$\delta_t \overline{T}^t = -\Gamma^{**}(T) + F^{T*} \qquad (56)$$

where

$$\Gamma^{**}(T) = ma^{-1}\left[\delta_\lambda(\overline{u\Delta\phi}^\phi \overline{T}^\lambda)/\Delta\phi + \delta_\phi(\overline{(v/m)\Delta\lambda}^\lambda \overline{T}^\phi)/\Delta\lambda\right] + \delta_z(w^T \overline{T}^z) \qquad (57)$$

and w^T is defined by

$$\Gamma^{**}(1) = 0 \qquad (58)$$

along with the boundary condition

$$w^T_{k=1/2} = 0. \qquad (59)$$

Constraints (29), (30) and (31) are satisfied by the same arguments used for the advective operator on momentum. The additional weighting by $\Delta\phi$ and $\Delta\lambda$ under the bar operators is needed to satisfy constraint (34) (see section D).

The diffusive operator may be written

$$F^{T*} = A_{TV}\delta_z\delta_z T + (A_{TH}/a^2)\nabla^{2*}T. \qquad (60)$$

Convective mixing, indicated by $\delta=0$ in (12), is accomplished in the model by testing the vertical static stability of each column of cells at the end of each timestep, volume averaging the T values for all cells which are found to be statically unstable, and resetting each cell to the average. This process simulates a vertical mixing with infinite mixing coefficient.

Arakawa, A., 1966: Computational design for long-term numerical integration of the equations of fluid motion: two dimensional incompressible flow. Part 1. J. Comput. Phys., 1, 119-143.

Richtmyer, R. D., and K. W. Morton, 1967: Difference Methods for Initial Value Problems, 2nd Ed., Interscience, New York.

Bryan, K., 1969: see section A.

Arakawa, A., and V. R. Lamb, 1977: Computational design of the basic dynamical processes of the UCLA general circulation model. Methods in Computational Physics, Vol. 17, Academic Press, 174-265.

D. Integral constraints

The importance of maintaining certain constraints on the volume integrals of kinetic and potential energy has been mentioned earlier. It is the purpose of this section to show formally that such constraints are satisfied by the particular set of finite difference equations presented here. The approach follows closely that of Bryan (1969).

If we denote the total kinetic energy (K.E.) by K, then

$$K = \overline{K} + \hat{K}, \qquad (61)$$

where

$$\overline{K} = (\overline{u}^2 + \overline{v}^2)/2 \qquad (62)$$

is the K.E. of the external mode and

$$\hat{K} = (\hat{u}^2 + \hat{v}^2)/2 \qquad (63)$$

is the K.E. of the internal modes. Let { } designate the volume integral over a closed ocean basin. The left side of (25), when multiplied by $m\psi/H$, integrated over the entire volume and rearranged becomes

$$\{(m/a^2H)[(m\psi\psi_{t\lambda}/H)_\lambda + (\psi\psi_{t\phi}/mH)_\phi]\}$$
$$-\{m^2(\psi_\lambda^2)_t/(a^2H^2) + (\psi_\phi^2)_t/(a^2H^2)\}/2.$$

The first two terms vanish due to boundary condition (26) on ψ, and the last two are the time derivative of the right side of (62) with (23) substituted. Therefore,

$$\{\overline{K}_t\} = -\{(m\psi/aH)[\overline{v'}_{t\lambda} - (\overline{u'_t}/m)_\phi]\} \qquad (64)$$

by (25) with u' and v' given by (18) and (19). Furthermore, the individual rates of change of K.E. due to a particular term on the right of (18) and (19) may be obtained by substituting that term for u' and v' in (64).

Multiplying (22) by $\hat{u},\hat{v}$ and integrating,

$$\{\hat{K}_t\} = \{\hat{u}u'_t+\hat{v}v'_t\}-\{\hat{u}\overline{u'_t}+\hat{v}\overline{v'_t}\}. \tag{65}$$

The second term on the right vanishes by definition of $\hat{u},\hat{v}$. Again, the contributions of the individual terms in (18) and (19) are obtained by substituting them in (65) for u',v'.

To obtain an expression for the rate of change of total K.E., it will be necessary to combine the finite difference equivalents of (64) and (65). For this purpose it will be helpful to establish several identities involving the finite difference operators. It is easy to verify that

$$\eta\Delta\lambda\delta_\lambda\mu+\overline{\mu\Delta\lambda\delta_\lambda\eta}^\lambda = \Delta\lambda\delta_\lambda(\overline{\eta}^\lambda\mu) \tag{66}$$

and

$$\overline{\overline{\eta}^\lambda\mu}^\lambda-\eta\overline{\mu}^\lambda= \Delta\lambda\delta_\lambda(\mu\Delta\lambda\delta_\lambda\eta)/4. \tag{67}$$

Combining (66) and (67) and substituting $\overline{\gamma}^\phi$ for μ,

$$\eta\Delta\lambda\delta_\lambda\overline{\gamma}^\phi+\overline{\gamma\overline{\Delta\lambda\delta_\lambda\eta}^\phi}^{\lambda\phi} = \Delta\lambda\delta_\lambda(\overline{\eta}^\lambda\overline{\gamma}^\phi)+\Delta\phi\delta_\phi[\overline{\gamma\Delta\phi\delta_\phi(\Delta\lambda\delta_\lambda\eta)}^\lambda]/4. \tag{68}$$

The finite difference equivalent of the right side of (64), corresponding to (55) is

$$\overline{I} = -\sum_{i}\sum_{j}\sum_{k}(ma\psi/H)\left[\delta_\lambda\overline{(\overline{G^v}^*)\Delta\phi}^\phi/\Delta\phi-\delta_\phi\overline{(\overline{G^u}^*/m)\Delta\lambda}^\lambda/\Delta\lambda\right]\Delta z\Delta\phi\Delta\lambda/m. \tag{69}$$

Since the integrand is constant in k, the vertical summation cancels with H. Using (68),

$$\bar{I} = a^2\sum_i\sum_j\Big[\overline{(\overline{G^v}^{*} m\delta_\lambda\Psi^\phi - \overline{G^u}^{*}\delta_\phi\Psi^\lambda)\Delta\phi\Delta\lambda/m}^{\lambda\phi}$$

$$-\Delta\lambda\delta_\lambda(\Psi^\lambda\overline{\Delta\phi\overline{G^v}^{*}}^{\phi}) + \Delta\phi\delta_\phi(\Psi^\phi\overline{\Delta\lambda\overline{G^u}^{*}/m}^{\lambda})$$

$$-\Delta\phi\delta_\phi[\overline{\overline{\overline{G^v}}^{*}\Delta\phi\Delta\phi\delta_\phi(\Delta\lambda\delta_\lambda\Psi)}^{\lambda}]/4$$

$$+\Delta\lambda\delta_\lambda[\overline{(\overline{G^u}^{*}/m)\Delta\lambda\Delta\lambda\delta_\lambda(\Delta\phi\delta_\phi\Psi)}^{\phi}]/4\Big]. \qquad (70)$$

The last four terms on the right vanish by boundary condition (26) on Ψ. Substituting (41),

$$\bar{I} = a^2\sum_i\sum_j\overline{H(\bar{u}^{*}\overline{G^u}^{*} + \bar{v}^{*}\overline{G^v}^{*})\Delta\phi\Delta\lambda/m}^{\lambda\phi}. \qquad (71)$$

The finite difference equivalent of the right side of (65), corresponding to (43) and (44) is

$$\hat{I} = a^2\sum_i\sum_j\sum_k(\hat{u}G^u + \hat{v}G^v)\Delta z\Delta\phi\Delta\lambda/m\ . \qquad (72)$$

Note that, since the integrand is defined at u,v points, this integral is taken on the alternate cells, with horizontal dimensions described by (36). However, it is easy to show that, for a closed basin, an equivalent expression is

$$\hat{I} = a^2\sum_i\sum_j\sum_k\overline{(\hat{u}G^u + \hat{v}G^v)\Delta z\Delta\phi\Delta\lambda/m}^{\lambda\phi}. \qquad (73)$$

Combining (71) and (73) we get an expression for the rate of change of total K.E.,

$$I = a^2\sum_i\sum_j\sum_k\overline{(uG^u + vG^v)\Delta z\Delta\phi\Delta\lambda/m}^{\lambda\phi} \qquad (74)$$

where total u and v are defined by (40). Again, the overbar is optional, making no difference in the value of the integral.

As stated earlier, the rates of change of K.E. due to individual terms of (45) and (46) may be evaluated by substituting them into (74) for G^u and G^v. For advection,

$$I_1 = a^2\sum_{i}\sum_{j}\sum_{k}[-u\Gamma^*(u)-v\Gamma^*(v)]\Delta z\Delta\phi\Delta\lambda/m. \tag{75}$$

This integral may be rewritten

$$I_1 = -\sum_{i}\sum_{j}\sum_{k}\left[u\sum_{b=1}^{6}V_b A_b u_b + v\sum_{b=1}^{6}V_b A_b v_b\right] \tag{76}$$

which is equivalent in form to (31). I_1 has been shown to vanish when u_b, v_b are defined as in (32). This condition is met by the advective operator (49). It is therefore established that no change in total K.E. occurs through advection.

Constraint (34) states that total (kinetic plus potential) energy must be conserved through the pressure term of (45) and (46) and the advection term of (56). Let

$$\gamma = \sum_{k'=1/2}^{k-1/2}\overline{\rho}^{z}_{k'}\,\overline{\Delta_{k'}z}^{z} \;;\; \delta_z\gamma = \overline{\rho}^{z}. \tag{77}$$

Then the rate of change of K.E. by the pressure term is

$$I_2 = ga\sum_{i}\sum_{j}\sum_{k}\overline{\left[-um\delta_\lambda\overline{\gamma}^{\phi}-v\delta_\phi\overline{\gamma}^{\lambda}\right]\Delta z\Delta\phi\Delta\lambda/m}^{\lambda\phi}. \tag{78}$$

Applying (68),

$$I_2 = ga\sum_i\sum_j\sum_k\Big[\gamma\Delta\lambda\delta_\lambda(\overline{u\Delta\phi}^\phi) - \Delta\lambda\delta_\lambda(\overline{\gamma}^\lambda\overline{u\Delta\phi}^\phi)$$

$$-\Delta\phi\delta_\phi\overline{[u\Delta\phi\Delta\phi\delta_\phi(\Delta\lambda\delta_\lambda\gamma)]}^\lambda/4$$

$$+\gamma\Delta\phi\delta_\phi(\overline{(v/m)\Delta\lambda}^\lambda) - \Delta\phi\delta_\phi(\overline{\gamma}^\phi\overline{(v/m)\Delta\lambda}^\lambda)$$

$$-\Delta\lambda\delta_\lambda\overline{[(v/m)\Delta\lambda\Delta\lambda\delta_\lambda(\Delta\phi\delta_\phi\gamma)]}^\phi/4\Big]\Delta z. \qquad (79)$$

For a closed basin, all terms on the right vanish except for the first and fourth. Combining them and substituting (58),

$$I_2 = ga^2\sum_i\sum_j\sum_k -\gamma\delta_z(w^T)\Delta z\Delta\phi\Delta\lambda/m. \qquad (80)$$

Using (66) and (77),

$$I_2 = ga^2\sum_i\sum_j\sum_k\Big[\overline{w^T\Delta z\overline{\rho}^z}^z - \Delta z\delta_z(\overline{\gamma}^z w^T)\Big]\Delta\phi\Delta\lambda/m. \qquad (81)$$

The second term vanishes in the summation, and applying (66) once more with $\eta=z$,

$$I_2 = ga^2\sum_i\sum_j\sum_k\Big[-z\Delta z\delta_z(w^T\overline{\rho}^z) + \Delta z\delta_z(\overline{z}^z w^T\overline{\rho}^z)\Big]\Delta\phi\Delta\lambda/m. \qquad (82)$$

where, once again, the last term vanishes in the summation. The remainder is equivalent to

$$I_2 = -ga^2\sum_i\sum_j\sum_k z\Gamma^{**}(\rho)\Delta z\Delta\phi\Delta\lambda/m \qquad (83)$$

since the two horizontal terms of (57) vanish in the summation. The expression above is the net loss of potential energy through advection of density, and constraint (34) is verified.

E. Program flow

The FORTRAN code of the model consists of the main program OCEAN, and seven subroutines. Their functions are described below.

- OCEAN : Performs all operations which need be done only once at the beginning of each run of the model, calls STEP once per timestep, and attends to operations which must be done at the end of each run.

- STEP : Called once per timestep by OCEAN, it initializes various quantities, bootstraps the row-by-row computation of prognostic variables, manages the I/O for the latter, and performs various analysis procedures on the progressing solution.

- CLINIC: Called once per row by STEP, it computes the internal mode component of the u and v velocities (Eqs. 43,44) as well as the vorticity driving function for use by RELAX later in determining the external mode (right side of Eq. 55).

- TRACER: Called once per row by STEP, it computes temperature, salinity, and any tracers which are carried in the model (Eq. 56).

- RELAX: Called once at the end of each timestep by STEP, it takes the vorticity driving function computed in CLINIC and, using sequential overralaxation, solves the LaPlacian equation for the external mode of velocity in terms of a mass transport stream function (Eq. 55).

- STATE: Called by CLINIC and TRACER once per row, and STEP in the bootstrap procedure, it computes normalized densities by using a 3rd order polynomial fit to the Knudsen formula (Eq. 8).

- MATRIX: Called by STEP on specified timesteps, it is a general 2-dimensional array printing routine.

- ODAM: (Ocean Direct Access Manager) Used primarily by STEP, it is a set of routines which is responsible for handling the transfer of data between memory and disc (virtual disc residing in memory in the core contained mode).

In summary, for one timestep, OCEAN calls STEP, which establishes the proper data in memory and calls CLINIC and TRACER row-by-row from south to north through the basin. Upon completing the final row, STEP calls RELAX and returns control to OCEAN, which may call STEP for another timestep.

Two additional subroutines, FINDEX and FILTER are used when Fourier filtering is needed to overcome the timestep limitation arising from convergence of meridians at high latitudes.

F. Disc I/O system

Next to the equations themselves, the component of the code which accounts for the greatest complexity is the I/O system. The purpose of this system is twofold. First, having a complete record of all prognostic variables on permanent disc at the end of each timestep allows restarting an experiment from an earlier run, or from a machine malfunction. Secondly, during a run, it is generally impossible to fit the three dimensional arrays in memory entirely. The I/O system is designed to feed data to and from memory in a row-by-row manner. Data for one row, including all east-west and vertical grid points is termed a "slab". At any one time, only the slabs necessary for the computation are present. While the computation for one row proceeds, the I/O system is feeding the slab just computed, back to disc and fetching the slab which will be needed to compute the next row. If the disc transfers are done fast enough, no wait for data will be needed at the beginning of each row and the system is said to be completely "buffered".

Three disc units are needed for the process when using centered differencing in time; one for the N timestep data (read), one for the N-1 timestep data (read), and one for the newly computed N+1 timestep data (written). Since the N level on one timestep becomes the N-1 level for the next, it is convenient to permute the disc units to minimize data transfers. Thus, on timestep 1, N-1 and N data are read from units 13 and 14, and the N+1 data is written to 15. On timestep 2, units 14 and 15 are read and the N+1 data is written to 13. On timestep 3, 15 and 13 are read with the N+1 data going to 14, etc.

Possibly the most abstruse feature of the model is the manner in which the "slab incidental data" is handled. Just as the slab system reduces the row dimension of the 3-dimensional prognostic variables to 3, it can also be used to reduce the row dimension of 2-dimensional variables which would otherwise add considerably to the memory requirements. Also, if these veriables are constant in time, there is no need to keep multiple time records of them. Consider two arrays, A and B, for which there is data, invariant in time, at each row and column horizontally across the grid. They may be carried in the model as "slab incidental data", thereby reducing their row dimension to 3. Furthermore, let A reside in the slab corresponding to the N-1 time level of the primary slab data, and B reside in the N level slab. The diagram below illustrates the 6 timestep cycle which the permuting disc units execute, with N-1, N and N+1 denoting the primary slab data, and A,B denoting slab incidental data.

TIMESTEP: UNIT	1	2	3	4	5	6	7
13	N-1,A	N+1,B	N,B	N-1,B	N+1,A	N,A	N-1,A
14	N,B	N-1,B	N+1,A	N,A	N-1,A	N+1,B	N,B
15	N+1,A	N,A	N-1,A	N+1,B	N,B	N-1,B	N+1,A

Note that on even timesteps, the slab incidental data enters memory in the wrong slabs and must be switched between slabs N and N-1. Also, storing of A and B into the N+1, slab must be alternated on successive timesteps. In the base code, FKMU and WSY are "A" type arrays, and FKMT and WSX are "B" type arrays.

Disc unit 12 is used primarily for the storage of 2-dimensional, horizontal fields. It is divided into 7+ blocks in the following manner:

Blocks 1-3:	permuting blocks for the stream function at N-1, N, and N+1 time levels
Block 4:	reciprocal of total depth
Blocks 5-6:	permuting blocks for former relaxation solutions
Block 7+ :	start and end indices (additional blocks are added as necessary when filtering indices fill block 7)

Finally, unit 11 contains the timestep counter, total elapsed time, and the area and volume of the basin.

Data is fed to and from these units by means of the entry points of subroutine ODAM. It, in turn, must use a direct access I/O package provided locally. If FORTRAN direct access I/O is available, with a facility to buffer the operations (such as a FIND statement), the QDAM calls in ODAM of the base code may be replaced by the appropriate FORTRAN statements. Otherwise, a specially written set of I/O utilities such as QDAM must be supplied. If, instead, the core-contained option is invoked, no such package is needed.

G. List of variables

The variables used in the routines OCEAN, STEP, CLINIC, TRACER, RELAX, STATE and ODAM are listed below in alphabetical order. The "base" names are often used in altered form, designated by a prefix or suffix described below:

Prefix

C2 : 2 times the base value

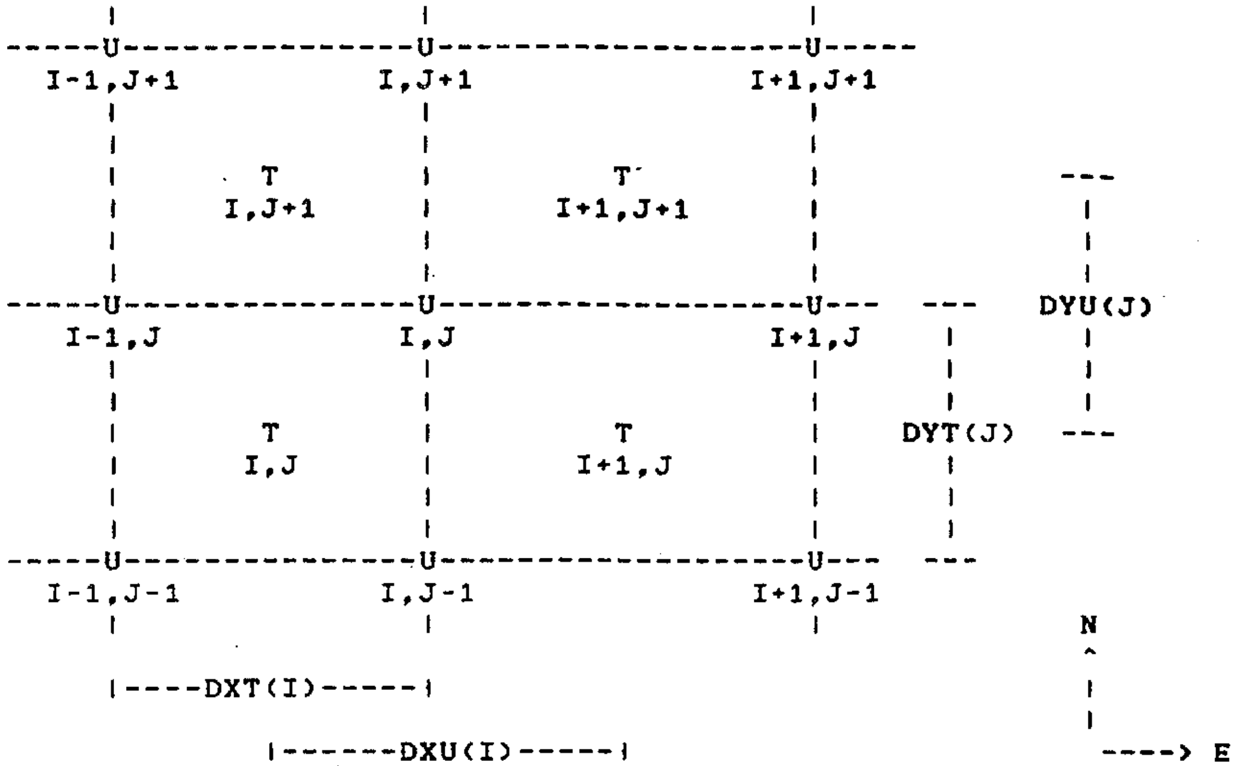
Suffixes

A : base value at timestep after present
B : base value at timestep before present
F : a forced full word version of the base value; no difference from base value in full word mode
M : base value at row J-1
P : base value at row J+1
Q : base value with a dimension added for purposes of vectorization
R : reciprocal of base value
2R : reciprocal of 2 times the base value
4R : reciprocal of 4 times the base value

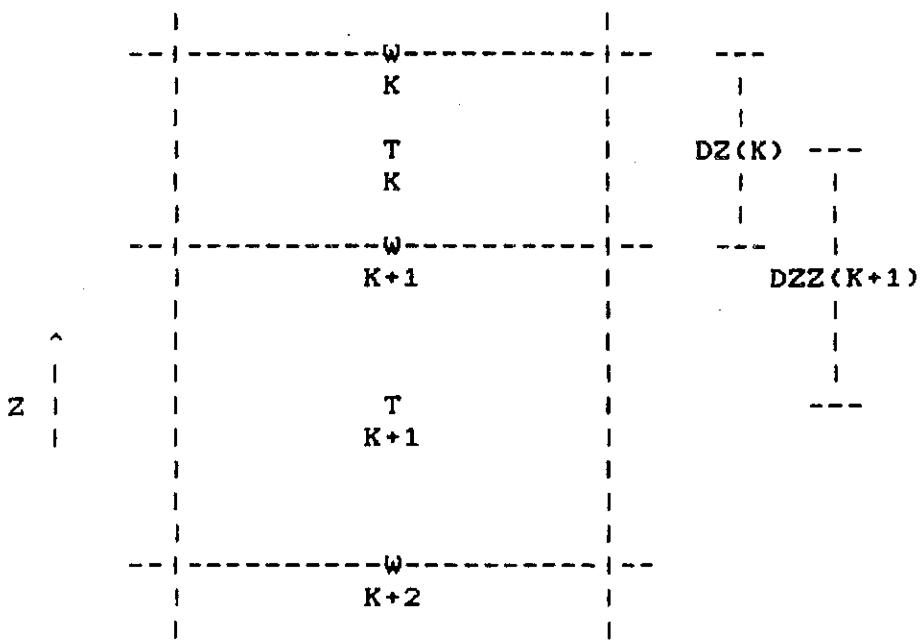
When these prefixes and suffixes are used in the following list, the base name which they supplement will be underlined.

A	Origination/destination array
ABT	Array of characters describing topography
ACOR	=0: treat Coriolis term explicitly >0: treat Coriolis term implicitly with forward component weighted by ACOR, past component by 1.-ACOR
ACORF	
AH	Coefficient of horizontal mixing of T
AHF	
AKNTRL	An unused series of words in the string saved to unit 11; available for storage of values which need to be saved on the restart file
AM	Coefficient of horizontal mixing of U,V
AMF	
AREA	Area of the surface of the model basin
BBTJ	Coefficient used in horizontal mixing of T
BBUJ	Coefficient used in horizontal mixing of U,V
BCON	Slab incidental data on N+1 slab
BIG	Virtual disc, holding disc data in core contained mode
BLK	A single blank character
BOXVOL	Volume of one grid box
BUOY	Energy transfer through buoyancy effects

C	In STATE: Coefficients of equation of state In ODAM: Equivalent to BIG, but dimensioned short to avoid 65K vector length limitation
CCTJ	Coefficient used in horizontal mixing of T
CCUJ	Coefficient used in horizontal mixing of U,V
CFE	Coefficient of eastern point in LaPlacian star
CFN	Coefficient of northern point in LaPlacian star
CFS	Coefficient of southern point in LaPlacian star
CFW	Coefficient of western point in LaPlacian star
CFWADR	Vector descriptor
CFWLOC	Index designator
CIQ	Knudsen formula coefficients with alternating reference levels
COF	Normalization array in computation of island flow
COFIS	Integral of COF
CONTRL	NAMELIST data file of program flow controllers
CPF	Normalization factor used in constructing LaPlacian star
CQ	Knudsen formula coefficients
CRIT	Criterion for convergence of relaxation
CRITF	
CRTP	Altered value of CRIT for computational efficiency
CS	Cosine of U,V point latitudes
CSR	
CST	Cosine of T point latitudes
CSTR	
C2DTSF	
C2DTTS	
C2DTUV	
C2DZ	
C2DZQ	
DAYSYR	Number of days per year
DDTJ	Coefficient used in horizontal mixing of T
DDUJ	Coefficient used in horizontal mixing of U,V
DETMR	Reciprocal of determinant of the matrix arising from simultaneous equations of the semi-implicit treatment of the Coriolis term
DIAG1	Temporary storage of diagonal differences for computational efficiency
DIAG2	" " " "
DOT	A single "." character
DPDX	Zonal derivative of hydrostatic pressure
DPDY	Meridional derivative of hydrostatic pressure
DTABS	Volume average of absolute change of temperature
DTSF	Length of timestep on stream function
DTSFF	
DTTS	Length of timestep on T
DTTSF	
DTUV	Length of timestep on internal mode of U,V
DTUVF	
DXT	Zonal grid spacing across T boxes (between U,V pts) (see figure on page G3)



HORIZONTAL GRID SPACING AND INDEXING



VERTICAL GRID SPACING AND INDEXING

DXTQ	
DXTR	
DXT2R	
DXT4R	
DXT4RQ	
DXU	Zonal grid spacing across U,V boxes (between T ts) (see figure on page G3)
DXUQ	
DXUR	
DXU2R	
DXU2RQ	
DXU4R	
DYT	Meridional grid spacing across T boxes (between U,V pts) (see figure on page G3)
DYTR	
DYT2R	
DYT4R	
DYU	Meridional grid spacing across U,V boxes (between T pts) (see figure on page G3)
DYUR	
DYU2R	
DYU4R	
DZ	Vertical grid spacing across U,V,T boxes (between w pts) (see figure on page G3)
DZZ	Vertical grid spacing across w boxes (between U,V,T pts) (see figure on page G3)
DZZQ	
DZZ2R	
DZZ2RQ	
DZ2R	
DZ2RQ	
EB	=.true. : Euler backward timestep for time mixing =.false. : forward timestep for time mixing
EDDY	NAMELIST data file of eddy mixing coefficients
EEH	Upper vertical mixing coefficient of T
EEHQ	
EEM	Upper vertical mixing coefficient of U,V
EEMQ	
EKTOT	Total kinetic energy normalized by volume
ENGEXT	Accumulators of rates of change of KE of external mode
ENGINT	Accumulators of rates of change of KE of internal mode
ENGTMP	Temporary accumulator of change of KE of external mode
FFH	Lower vertical mixing coefficient of T
FFHQ	
FFM	Lower vertical mixing coefficient of U,V
FFMQ	
FINS	Floating point array to read in start and end indices
FKMP	Number of vertical levels of ocean at T points
FKMQ	Number of vertical levels of ocean at U,V points
FKMT	Number of vertical levels of ocean at T points

FKMTM	
FKMTP	
FKMU	Number of vertical levels of ocean at U,V points
FKMUM	
FKMUP	
FKMZ	Number of vertical levels of ocean at interior (non-land neighboring) T points
FKPH	Coefficient of vertical mixing of T
FKPHF	
FKPM	Coefficient of vertical mixing of U,V
FKPMF	
FM	Masking array for T points; 0 over land, 1 over ocean
FMM	
FMP	
FUW	Advective coefficient for west face of U,V box in CLINIC; T box in TRACER
FVN	Advective coefficient for north face of U,V box in CLINIC; T box in TRACER
FVST	Advective coefficient for south face of T box
FVSU	Advective coefficient for south face of U,V box
FX	Temporary value, constant in subsequent DO loops
FXA	Temporary value, constant in subsequent DO loops
FXB	Temporary value, constant in subsequent DO loops
FXC	Temporary value, constant in subsequent DO loops
GGUJ	Coefficient used in horizontal mixing of U,V
GM	Masking array for U,V points; 0 for land, 1 for ocean
GRAV	Acceleration due to gravity
H	Total depth at U,V points
HHUJ	Coefficient used in horizontal mixing of U,V
HR	Reciprocal of total depth at U,V points
I	Zonal grid point index (see figure on page G3)
IBK	DO loop index
IBOX	NAMELIST data file of island box corner point indices
IDX	=0: can't use Fourier coefficients from previous call >0: can use Fourier coefficients from previous call
IE	Ending index for I DO loop
IEAVE	Ending index for previous I DO loop
IEIS	Ending I index of island box
IEND	Index designator
IEND5L	Index designator
IEPT	Index designator
IEPU	Index designator
IETF	Array of ending indices for filtering T
IEUF	Array of ending indices for filtering U,V
IEZ	Array of ending indices for vorticity
IEZF	Array of ending indices for filtering vorticity
II	DO loop index
IKM	Index designator
IM	Length of strip to be filtered

IMT	Total number of T grid boxes zonally
IMTKM	IMT*KM
IMTM1	IMT-1
IMTM2	IMT-2
IMTP1	IMT+1
IMU	Total number of U,V grid boxes zonally
IMUM1	IMU-1
IMUM2	IMU-2
IND	Indicates which levels to compare for static stability
INTBV	Bit interior indicator; =1 on interior (non-land neighboring) points
IPRT	Number of columns to print beginning at western boundary
IREDO	Same function as IDX
IS	Starting index for I DO loop
ISAVE	Starting index for previous I DO loop
ISIS	Starting I index of island box
ISLE	DO loop index indicating island being computed
ISMASK	Grid point type indicator; =0 over interior (non-land neighboring) points =1 over perimeter (land neighboring) points =2 over land points
ISP	Index designator
ISTF	Array of starting indices for filtering T
ISTOP	Final column to be printed
ISTORE	Index designator
ISTRT	First column to be printed
ISUF	Array of starting indices for filtering U,V
ISZ	Array of starting indices for vorticity
ISZF	Array of starting indices for filtering vorticity
ITT	Timestep counter; total number of timesteps completed
J	Meridional grid point index (see figure on page G3)
JE	Ending index for J DO loop
JEIS	Ending J index of island box
JFRST	First J row of T to be filtered
JFTO	Filtering is done on T with a low-pass cut-off set to make the zonal dimension of the box filtered effectively the same as that of the boxes on row JFTO
JFT1	Last J row of T in southern hemisphere to be filtered
JFT2	First J row of T in northern hemisphere to be filtered
JFUO	Same function as JFTO but for U,V points
JFU1	Last J row of U,V in southern hemisphere to be filtered
JFU2	First J row of U,V in northern hemisphere to be filtered
JJ	Index designator
JMT	Total number of T grid boxes meridionally
JMTM1	JMT-1
JMTM2	JMT-2
JMTP1	JMT+1
JREV	DO loop index in reverse order
JS	Starting index for J DO loop

JSIS	Starting J index of island box
JSKPT	Number of rows of T not filtered in mid & low latitudes +1
JSKPU	Number of rows of U,V not filtered in mid & low lats. +1
K	Vertical grid point index (see figure on page G3)
KAR	KAR(K)=K used to enable vectorization
KFLDS	Disc unit number (12) for 2-D horizontal fields and start and end indices
KM	Total number of vertical levels
KMM1	KM-1
KMP1	KM+1
KMP2	KM+2
KMT	Number of vertical levels of ocean at T points
KMTP	
KMU	Number of vertical levels of ocean at U,V points
KMUP	
KONTRL	Disc unit number (11) for timestep counter etc.
KPR	Temporary array for printing topography map
KREF	Reference level indicator
KS	Indicates which levels to compare for static stability
KZ	Temporary indicator of number of levels of ocean points
L	Index designator
LABS	Disc unit numbers (13-15) for slabs
LBC	Number of arrays of slab incidental data
LEN	Block length indicator
LENV	Vector length indicator
LENV1	Vector length indicator
LENV2	Vector length indicator
LL	Index designator
LN	Vector length indicator
LO	Restart tape unit number
LSEG	Max. number of sets of start and end indices for vorticity
LSEGF	Max. number of sets of start and end indices for filtering
LSEGP	LSEG+1
LU	Disc unit number
LUPTD	Permuting disc unit number for saving previous relaxation solution
LUPTDB	Permuting disc block number for saving relaxation solution of 2 timesteps previous; if LUPTD=5, then LUPTDB=6; when next solution is obtained, LUPTDB is set to 5, LUPTD to 6 and the new solution is written to LUPTD.
M	Index indicating the tracer being computed
MIX	Mixing timestep indicator; =0 : no mixing timestep =1 : mixing timestep
MM	Index indicating the tracer being computed
MSB	Unused
MSCAN	Relaxation scan counter
MTEST	Print on-off switch
MXP	=1 on 2nd pass of Euler backward timestep =0 otherwise

MXSCAN	Maximum number of relaxation scans permitted
N	Index designator
NA	Unused integer read in by NAMELIST; available for use (used in core contained mode to enable restart tape write)
NB	Unused integer read in by NAMELIST; available for use
NBLK	Number of words per block on disc unit
NBUF	Number of buffers set aside for disc I/O use
NC	Unused integer read in by NAMELIST; available for use
NCON	Number of passes to make on convection; since mixing is done only 2 levels at a time, total homogenization of unstable levels does not occur in one pass
NDICES	Total number of start and end indices
NDISK	Permutes with NDISKB and NDISKA on 13,14,15 indicating the disc units for the slabs at various time levels; see section F
NDISKA	
NDISKB	
NDISKX	Temporary disc unit to which data is written on 2nd pass of Euler backward timestep
NDW	Unused integer read in by NAMELIST; suggest using it to indicate writeout of data for analysis purposes
NE	Ending DO loop limit
NERGY	=1 to execute energy/printout code =0 otherwise
NFIRST	=1 to start a run from scratch =0 to restart from data supplied on units 11-15
NFRST	Disc unit address of the 1st word to be transferred
NISLE	Number of islands in the model basin
NJTBFT	Number of J rows to be filtered on T
NJTBFU	Number of J rows to be filtered on U,V
NLAST	Final timestep to compute on this run of the model
NKFLDS	Number of 2-D fields needed on disc unit 12; this is 6 plus the number of (equivalent) 2-D fields needed to contain the start and end indices, normally one.
NMIX	Number of timesteps between mixing timesteps; mixing is done to suppress the computational mode associated with leap-frog timestepping; see EB
NNERGY	No. of timesteps between execution of energy/printout code
NS	Starting Do loop limit
NSKP	Index designator
NSLAB	Number of words in one slab
NSSIF	Total number of start and end indices for filtering
NSWICH	Number of words in slab incidental data within each slab which is non-prognostic and must be switched; see sec. F
NT	Number of tracer type variables carried in the model; this is generally 2 (T and S) + no. of passive tracers
NTMIN2	Maximum of NT or 2
NTOT	Total length of a disc unit
NTSI	Number of timesteps between print of single line of information containing timestep number, KE, etc.

NWDS	IMT*JMT In ODAM: Number of words to transfer in a disc operation
NWRITE	Unused integer read in by NAMELIST; available for use (used in core contained mode as number of timesteps between backup restart write)
NWRS	Number of words to transfer in a disc operation
OMEGA	Rate of rotation of the coordinate system
P	Mass transport stream function
PAD	Unused memory space to prevent overwriting on I/O
PARMS	NAMELIST data file of various parameters of the model
PB	
PHI	Latitude in radians of the U,V points
PHIT	Latitude in radians of the T points
PI	3.1415927
PLICEX	Energy change due to implicit effects on external mode
PLICIN	Energy change due to implicit effects on internal mode
PTD	Change of stream function across a timestep
PTDB	Change of stream function across previous timestep
RADIAN	57.29578 degrees
RADIUS	Radius of the earth
RES	Residual of relaxation
RESADR	Vector descriptor
RESIS	Residual of relaxation of island
RESLOC	Index designator
RESMAX	Maximum of the residuals at all grid points
RHO	Density with a space and time invariant constant subtracted; the constant varies with level and is approximately equal to the mean density for the level
RHON	RHO for the row to the north
RHOS	RHO for the row to the south
SCL	Scaling factor for printout
SFU	External mode component of U (used elsewhere for temporary storage)
SFUB	
SFV	External mode component of V (used elsewhere for temporary storage)
SFVB	
SINE	Sine of U,V point latitudes
SO	Normalizing salinities for KNUDSN coefficients
SOQ	
SOIQ	Normalizing salinities with alternating reference level
SOR	Coefficient of over-relaxation; generally between 1.5-1.8
SORF	
SPCOS	Coefficients used to turn vectors before filtering and restore them afterward, to compensate for spher. geom.
SPSIN	See SPCOS
SQ	Normalized salinities
SUMDY	Summation of DYT
SWLDEG	Latitude in degrees of the southern wall (PHI(1)*RADIAN)
SX	Salinities

T	Tracer type of variables (temperature, salinity, tracers)
TA	
TB	
TBM	
TBP	
TBRN	Zonal summation of tracer, defined to the north
TBRS	Zonal summation of tracer, defined to the south
TBRZ	Zonal/vertical average of tracer
TBSLAB	Array of entire N-1 slab
TDIF	Diffusion computation array
TEMPA	Utility array used as temporary storage
TEMPB	Utility array used as temporary storage
TEST1	Product of reciprocals of four surrounding depths
TEST2	Sum of reciprocals of four surrounding depths
TFULL	Array of temporary storage forced full precision
TINIT	Initial values of tracers
TINITF	
TM	
TMT	Meridional mass transport
TNG	Tangent of U,V point latitudes
TO	Normalizing temperatures for KNUDSN coefficients
TOQ	
TOIQ	Normalizing temperatures with alternating reference levels
TOTDX	Total zonal span of ocean boxes
TOTDZ	Total vertical span of ocean boxes
TP	
TQ	Normalized temperatures
TSLAB	Array of entire N slab
TSPROF	NAMELIST data file for initial tracers
TSTEPS	NAMELIST data file for timestep lengths
TTDAY	Current day-of-the-year number
TTDTOT	Array of integrals on tracers
TTN	Northward transport of tracers
TTSEC	Current total elapsed time in seconds
TTYEAR	Number of years of integration completed
TVAR	Change of variance of tracers
TX	Temperatures
TO	Parameter used in timing code
T1	Parameter used in timing code
T2	Parameter used in timing code
T3	Parameter used in timing code
U	Zonal component of velocity
UA	
UB	
UBM	
UBP	
UCLIN	Array of internal mode component of U at row J+1
UDIF	Vertical mixing computation array for U
UENG	Individual forcing terms on U
UM	

UOVER	Positioned in COMMON to be equivalent of UDIF with K index equal 0; used to set surface boundary conditions
UP	
USAV	Array of internal mode component of U at row J
UUNDER	Positioned in COMMON to be equivalent of UDIF with K index equal KM+1
V	Meridional component of velocity
VA	
VB	
VBM	
VBP	
VBR	Zonal summation of meridional velocity
VBRZ	Zonal/vertical summation of meridional velocity
VCLIN	Array of internal mode component of V at row J+1
VDIF	Vertical mixing computation array for V
VENG	Individual forcing terms on V
VM	
VOLUME	Total volume of the model basin
VOVER	Positioned in COMMON to be equivalent of VDIF with K index equal 0; used to set surface boundary conditions
VP	
VSAV	Array of internal mode component of V at row J
VUNDER	Positioned in COMMON to be equivalent of VDIF with K index equal KM+1
W	Vertical velocity computed in vertical line with U,V in CLINIC computed in vertical line with T in TRACER
WSX	Zonal component of surface wind stress
WSXM	
WSXP	
WSY	Meridional component of surface wind stress
WSYM	
WSYP	
ZDZ	Vertical position of bottom of levels
ZDZZ	Vertical position of center of levels
ZTD	Change of vorticity across one timestep
ZUN	Time change of vertically averaged zonal forcing
ZUNENG	Vertical average of U forcing
ZVS	Time change of vertically averaged meridional forcing
ZVSENG	Vertical average of V forcing

H. Updating and code options

There is currently no standardization within the computer industry concerning specific procedures used in code updating utilities. It is, therefore, impossible to provide the model code with update directives that are usable by all updating systems. For this reason, a separate, stand alone updating utility, "UPDOC" (UPdate OCean) has been provided with which the "base code" which is supplied here, can be altered to produce the specific model desired. By necessity, this is a "batch", as opposed to "interactive" type of updating system. It requires the construction of a separate set of updates which are then used to alter the base code. Obviously, the update sets themselves may be constructed and altered interactively. Users who desire to alter their code directly in interactive mode must be aware that their code will quickly become divorced from the base code supplied here, and future interactions with other users, such as use of externally supplied updates, will be quite difficult. (Note.. such a direct interactive mode may be feasible if a COMPARE utility is available locally.)

UPDOC uses information in columns 82-100 of the FORTRAN lines of the base code. A sequence number appears in columns 82-90, and code option characters appear in columns 92-100. The function of the latter will be described first.

Code can be treated optionally in a FORTRAN program either at execution time by means of "IF" statements, or in a pre-compilation, update step. The latter has the advantage of shortening and simplifying the executable code and is used here. The options provided in the base code are designated by single letters and are listed below with their respective functions:

C : include comments

F : include Fourier filtering at high latitudes

H : run in half-word mode

I : islands allowed in the topography

K : run core contained -- no disc I/O necessary

O : cyclic conditions are set east-west

Q : code which is specific to the CYBER 205

S : symmetry conditions are set at the northern boundary

T : timing analysis is done for each subroutine (a routine must be available to interrogate the system clock - replace GETIME & Q5TIME)

W : WHERE statements are used in place of IF

A single record of input ("option record") is read by UPDOC from unit 10, to control the execution/deletion of optional lines from the base code. It begins in column 1 and its format is:

```
OPT=X,X,X,...
```

where the X's correspond to the option letters above. A line from the base code is executed if and only if <u>all</u> of the option letters designated on it appear in the option record. There are also reverse logic options designated on some of the lines of the base code, indicated by a "-" preceding them. For these cases, the line is executed if and only if the option letter does <u>not</u> appear in the option record to UPDOC.

The sequence number in columns 82-90 is used by UPDOC to alter the base code beyond the preset options described above. The high 4 digits in these numbers will remain unchanged for all time. The lower 5 are used for updating. The base code is altered in the following manner by records 2 and beyond in unit 10:

replace a line in the base code : Include the new line in unit 10 with the same sequence number as the replaced line in columns 82-90.

insert a line in the base code : Include the new line in unit 10 with a sequence number between those of its intended neighbors.

delete line(s) from the base code : Include a record of the following format beginning in column 1:

```
-XXXXXXXXX,YYYYYYYYY
```

causing lines with sequence numbers XXXXXXXXX through YYYYYYYYY to be deleted. The comma and second number may be omitted, in which case only line XXXXXXXXX is deleted. Full, 9 digit numbers must be provided (include leading zeroes).

The update command records must be ordered such that the sequence numbers of the lines processed are in ascending order. In the case of deletions, the first number on the deletion line must be greater than the previous sequence number processed.

It is possible under UPDOC, to perform separate, or modular update stages. This is convenient if the user has constructed a separate update module for say, a special mixing parameterization, which is to be applied along with other updates to form the final model. Such modules must be separated in unit 10 by a line with the character "/" in column 1. The remainder of the line is free-form and may contain an identifying comment. Note that the ascending order constraint on update sequence numbers is released between modules, but remains in effect within each module.

It is convenient to categorize updates in the following manner:

local : updates which are used to construct a specific model

global : updates which are more general in nature and are of use in many different models

permanent : updates which improve the code for all possible applications

In constructing local updates, only the lower three digits of the sequence numbers should be altered. Digits 4 and 5 can then be used for global and permanent updates, thereby limiting the possible interference when using multiple update sets.

The final process within UPDOC is the distribution of the COMDECKS, common PARAMETER and COMMON statements, throughout the code where they are designated with *CALL commands in the base code.

After constructing the input file (unit 10), the base code is placed on unit 20, and UPDOC is executed, rewriting unit 20 with the resulting model code. This file can then be compiled and executed.

Permanent updates should be made only through the GFDL staff. As stated earlier, users are encouraged to make homemade global updates available to other users through GFDL. They should be written as generally as possible and well checked out. All known users will be notified periodically of the latest permanent and global updates available.

I. Memory requirements

The following symbols will be used in the memory formulas below:

- IMT : total number of grid points east-west
- JMT : total number of grid points north-south
- KM : total number of grid points vertically
- LBC : number of arrays kept as slab incidental data
- NKFLDS : number of IMTxJMT fields stored on unit 12 (normally 7)
- NISLE : number of islands in the model
- I : 1 if NISLE>0
0 if NISLE=0
- H : 1 if in fullword mode
1/2 if in halfword mode
- C : 0 if in disc I/O mode
1 if in core contained mode

The common block, WORKSP, is the single largest block of memory used by the model for most applications. Since the relaxation solution for the external mode is done after all of the row-by-row computations of CLINIC and TRACER are completed, the same work space may be used for both. Therefore, WORKSP appears in an alternate version in RELAX. Depending on the grid shape, the RELAX version (R) may be smaller or larger than the standard version (S) found in the other routines. The total space taken by each of the two is:

MEM(S)=(78+8*NT)*IMT*KM+(37+7*LBC)*IMT

MEM(R)=6*IMT*JMT+IMT+I*2*IMT*JMT+NISLE

The requirement for WORKSP is the maximum of these two. The requirement for the next largest common block, FIELDS, is 4*IMT*JMT. Data in singly dimensioned arrays takes extremely little memory and will not be considered here. If the model is run in core contained mode, an additional block must be provided for the virtual disc, "BIG":

MEM(BIG)=20+NKFLDS*IMT*JMT+2*((NT+2)*IMT*KM+LBC*IMT)*JMT

If the model is run in halfword mode, these figures are all multiplied by 1/2. The total data memory requirement is then

MEM(TOTAL)=[Maximum of (MEM(S),MEM(R))+4*IMT*JMT+C*MEM(BIG)]*H.

In addition, memory will be used by both the object code of the model, and I/O buffers (in the disc I/O mode).

J. Polynomial approximation to the Knudsen formula.

A technique is described in Bryan and Cox (1972) whereby, for each level of the model individually, a 9-term, 3rd order polynomial in temperature and salinity may be constructed to serve as the equation of state, which closely approximates the Knudsen formula for the density of seawater. A considerable decrease in computational effort is realized over using the formula directly. To use this technique, a set of nine coefficients must be provided for each level. The program "KNUDSN" generates FORTRAN "DATA" statements on unit 10 containing these coefficients, with the proper sequence numbers included, so they can be merged directly into the unit 10 input file UPDOC. On a listing of "KNUDSN", near the top are two comment lines with asterisks extending to the right, directing the entry of, first, the number of levels (LEVELS=), corresponding to KM in the base code, and secondly, for each level, the thickness in centimeters corresponding to DZ in the base code. The bounds of T and S over which the polynomial is to be fitted is given by the array TS for each 250 meter depth span of the ocean. It is currently set at a reasonably general distribution but may be altered if unusual T or S values are expected, such as in a paleo-oceanographic model.

The values of LEVELS and DZ currently entered correspond to those of sample models 2 and 3 (see section L). "KNUDSN" may be run in its present form to obtain a check with the coefficients used there. Note, however, that somewhat different values may be obtained, particularly for the higher order terms, due to different precision and/or object code sequence produced by different machines.

Bryan, K., and M. D. Cox, 1972: An approximate equation of state for numerical models of ocean circulation. J. Phys. Oceanogr., 2, 510-514.

```
*COMDECK PARAM
      PARAMETER (INT=??,JNT=??,KM=??,NT=??,LSEG=??,NISLE=??,LBC=??
     *,LSEGF=??,JFRST=??,JFTO=??,JFT1=??,JFT2=??,JFUO=??,JFU1=??,JFU2=??
C
C  FILTER T TO YIELD EQUIV DX AT J=JFTO FROM J=JFRST TO JFT1 AND
C         J=JFT2 TO JNTM1
C  FILTER U TO YIELD EQUIV DX AT J=JFUO FROM J=JFRST TO JFU1 AND
C         J=JFU2 TO JNTM2
C
     *,INU=INT-1
     *,INU=INT
     *,INTP1=INT+1,INTM1=INT-1,INTM2=INT-2,INUM1=INU-1,INUM2=INU-2
     *,JNTP1=JNT+1,JNTM1=JNT-1,JNTM2=JNT-2,JSCAN=JNTM2
     *                                                    +1
     *,KMP1=KM+1,KMP2=KM+2,KMM1=KM-1
     *,NSLAB=INT*((NT+2)*KM+LBC),NWDS=INT*JNT,NSWICH=LBC*INT
     *,JSKPT=JFT2-JFT1,JSKPU=JFU2-JFU1
     *,NJTBFT=(JFT1-JFRST+1)+(JNTM1-JFT2+1)
     *,NJTBFU=(JFU1-JFRST+1)+(JNTM1-JFU2+1)
     *,NSSIF=2*(NJTBFT+NJTBFU)*LSEGF*KM+2*NJTBFU*LSEGF
     *,NDICES=2*LSEG*JNT+4*NISLE
     *                              +NSSIF
     *,INTKM=INT*KM,NTMIN2=NT+1/NT,NKFLDS=7+(NDICES/NWDS))
      IMPLICIT HALF PRECISION(A-H,O-Z)
*COMDECK FULLWD
      REAL TTSEC,AREA,VOLUME,AKNTRL,PAD,ENGINT,ENGEXT,TTDTOT,BUOY,
     *     PLICIN,PLICEX,EKTOT,DTABS,TVAR
      LOGICAL EB
      COMMON /FULLWD/ ITT,TTSEC,AREA,VOLUME,AKNTRL(6),PAD(10),
     * ENGINT(8),ENGEXT(8),TTDTOT(6,NT),BUOY,PLICIN,PLICEX,EKTOT,
     * DTABS(NT),TVAR(NT),NFIRST,NLAST,NMIX,NNERGY,NWRITE,NA,NB,NC,
     * MXSCAN,NDISKB,NDISK,NDISKA,KONTRL,KFLDS,LABS(3),MIX,MXP,NERGY,
     * NSCAN,NSB,NDW,NTSI,KAR(KM),
     * KMT(INT),KMTP(INT),KMU(INT),KMUP(INT),EB,
     * ISZ(JNT,LSEG),IEZ(JNT,LSEG)
     *,ISIS(NISLE),IEIS(NISLE),JSIS(NISLE),JEIS(NISLE)
     *,ISTF(NJTBFT,LSEGF,KM), IETF(NJTBFT,LSEGF,KM),
     * ISUF(NJTBFU,LSEGF,KM), IEUF(NJTBFU,LSEGF,KM),
     * ISZF(NJTBFU,LSEGF),    IEZF(NJTBFU,LSEGF),
     * SPSIN(INT), SPCOS(INT)
*COMDECK SCALAR
      COMMON /SCALAR/ DTTS,DTUV,DTSF,C2DTTS,C2DTUV,C2DTSF,AH,AM,FKPH,
     * FKPM,ACOR,CRIT,SOR,OMEGA,RADIUS,GRAV,RADIAN,PI,SWLDEG
*COMDECK ONEDIM
      COMMON /ONEDIM/
     * DXT  (INT),DXTR (INT),DXT2R(INT),DXU  (INT),DXUR (INT),DXU2R(INT)
     *,DXU4R(INT),DXT4R(INT),SFU  (INT),SFUB (INT),SFV  (INT),SFVB (INT)
     *,ZUN  (INT),ZUS  (INT),ZVN  (INT),ZVS  (INT)
     *,DYT  (JNT),DYTR (JNT),DYT2R(JNT),DYU  (JNT),DYUR (JNT),DYU2R(JNT)
     *,DYU4R(JNT),DYT4R(JNT),CS   (JNT),CSR  (JNT),CST  (JNT),CSTR (JNT)
     *,PHI  (JNT),PHIT (JNT),SINE (JNT),TNG  (JNT)
     *,C2DZ ( KM),DZ   ( KM),DZ2R ( KM),EEH  ( KM),EEM  ( KM),FFH  ( KM)
     *,FFM  ( KM),ZDZ  ( KM)
     *,DZZ (KMP1),DZZ2R(KMP1),ZDZZ(KMP1),TINIT(KM,NT)
*COMDECK FIELDS
      COMMON /FIELDS/ P(INT,JNT),PB(INT,JNT),ZTD(INT,JNT),HR(INT,JNT)
*COMDECK WRKSPA
      COMMON /WORKSP/
     *TA (INT,KM,NT),UA (INT,KM),VA (INT,KM),BCON(INT,LBC),
     *TBP(INT,KM,NT),UBP(INT,KM),VBP(INT,KM),
     *  FKMUP(INT),WSYP(INT),
     *TP (INT,KM,NT),UP (INT,KM),VP (INT,KM),
     *  FKMTP(INT),WSXP(INT),
     *TB (INT,KM,NT),UB (INT,KM),VB (INT,KM),
     *  FKMU (INT),WSY (INT),
     *T  (INT,KM,NT),U  (INT,KM),V  (INT,KM),
     *  FKMT (INT),WSX (INT),
     *TBM(INT,KM,NT),UBM(INT,KM),VBM(INT,KM),
     *  FKMUM(INT),WSYM(INT),
     *TM (INT,KM,NT),UM (INT,KM),VM (INT,KM),
     *  FKMTM(INT),WSXM(INT)
      COMMON /WORKSP/
     * UCLIN(INT,KM),VCLIN(INT,KM),USAV (INT,KM),VSAV (INT,KM),
     * RHON (INT,KM),RHOS (INT,KM),FUW  (INT,KM),FVN  (INT,KM),
     * FVSU (INT,KM),FVST (INT,KM),
     * FMM  (INT,KM),FM   (INT,KM),FMP (INT,KM),
     *                GM   (INT,KM),
     *    UOVER(INT),UDIF (INT,KM),UUNDER(INT),
     *    VOVER(INT),VDIF (INT,KM),VUNDER(INT),
     * W(INT,KMP1),TEMPA(INT,KMP1),TEMPB(INT,KMP1),
     * TDIF(INT,KMP2,NTMIN2),
     * ZUNENG(INT,8),ZUSENG(INT,8),ZVNENG(INT,8),ZVSENG(INT,8)
      COMMON /WORKSP/
     * DXTQ  (INT,KM),DXUQ  (INT,KM),DXT4RQ(INT,KM),DXU2RQ(INT,KM),
```

```
     * DZ2RQ (IMT,KM),DZZQ  (IMT,KM),DZZ2RQ(IMT,KM),C2DZQ (IMT,KM),
     * EEHQ  (IMT,KM),EEMQ  (IMT,KM),FFHQ  (IMT,KM),FFMQ  (IMT,KM),
     * CQ (IMT,KM,9  ),TOQ (IMT,KM  ),SOQ (IMT,KM  ),
     * CIQ(IMT,KM,9,2),TOIQ(IMT,KM,2),SOIQ(IMT,KM,2)
*COMDECK WRKSPB
      COMMON /WORKSP/
     * ISMASK(IMT,JMT),COF(IMT,JMT),COFIS(NISLE),
     * CFN(IMT,JMT),CFS(IMT,JMT),CFE(IMT,JMT),CFW(IMT,JMT),
     * PTD(IMT,JMT),RES(IMT,JMT),CPF(IMT)
*COMDECK BITVEC
      BIT KEVENBV(IMT,KM),KODDBV(IMT,KM),KALTBV(IMT,KM,2)
      EQUIVALENCE (KODDBV,KALTBV(1,1,1)),(KEVENBV,KALTBV(1,1,2))
      COMMON /BITVEC/ KALTBV
*COMDECK TIME
      REAL TO,T1,T2,T3,TIME
      COMMON /TIME/ TIME(10)
```

```
*DECK OCEAN
      PROGRAM OCEAN(INPUT,TAPE5=INPUT,OUTPUT=TAPE6)
C
C*****************************************************************************
C                                                                 ***********
C  OCEAN IS THE PRIMARY CALLING ROUTINE.  IT PERFORMS ALL          ***********
C        OPERATIONS WHICH NEED BE DONE ONLY ONCE AT THE           ***********
C        BEGINNING OF EACH RUN OF THE MODEL, CALLS STEP           ***********
C        ONCE PER TIMESTEP, AND ATTENDS TO OPERATIONS             ***********
C        WHICH MUST BE DONE ONLY AT THE END OF EACH RUN.          ***********
C                                                                 ***********
C        THE CALLING SEQUENCE OF THE CODE IS AS FOLLOWS:          ***********
C                                                                 ***********
C                                   ---> CLINIC - STATE           ***********
C        ODAM     ODAM            /                               ***********
C          \        \            /                                ***********
C    -->  OCEAN --> STEP -- - ---> TRACER - STATEC                ***********
C                   /          \                                  ***********
C               MATRIX          \                                 ***********
C                                   ---> RELAX - ODAM             ***********
C                                                                 ***********
C*****************************************************************************
C
C-----------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C-----------------------------------------------------------------------
C
*CALL PARAM
*CALL FULLWD
*CALL SCALAR
*CALL ONEDIM
*CALL FIELDS
*CALL WRKSPA
*CALL BITVEC
*CALL TIME
C
C-----------------------------------------------------------------------
C  DEFINE AND EQUIVALENCE LOCAL DATA; DEFINE NAMELIST INPUT
C-----------------------------------------------------------------------
C
      DIMENSION FKMP(IMT,JMT),FKMQ(IMT,JMT),FKMZ(IMT,JMT),FINS(3)
      CHARACTER*1 DOT,BLK,ABT
      DIMENSION ABT(IMT)
      DIMENSION KPR(IMT)
      EQUIVALENCE (P,FKMP,FINS),(PB,FKMZ),(ZTD,FKMQ)
      REAL SUMDY
      REAL AMF,AHF,FKPMF,FKPHF,DTTSF,DTUVF,DTSFF,ACORF,SORF,CRITF,
     * TINITF(KM,NT)
      NAMELIST /EDDY/ AMF,AHF,FKPMF,FKPHF /TSTEPS/ DTTSF,DTUVF,DTSFF
     * /PARMS/ ACORF,MXSCAN,SORF,CRITF
     */CONTRL/ NFIRST,NLAST,NNERGY,NMIX,NWRITE,NDW,NTSI,NA,NB,NC
      NAMELIST /TSPROF/ TINITF
      NAMELIST /IBOX/ ISIS,IEIS,JSIS,JEIS
      DATA DOT/'.'/,BLK/' '/
C
C-----------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C-----------------------------------------------------------------------
C
C*****************************************************************************
C  BEGIN INTRODUCTORY SECTION WHICH IS NEEDED FOR EACH RUN   *******************
C       (INCLUDING RESTARTS)                                 *******************
C*****************************************************************************
C
C
      DO 10 LTIME=1,10
 10     TIME(LTIME)=0.
      CALL GETIME(T3,T1)
C
C-----------------------------------------------------------------------
C  SET THE TYPE OF MIXING TIMESTEP.  IF EB=.TRUE., AN EULER
C  BACKWARD STEP IS DONE; IF EB=.FALSE., A FORWARD STEP IS DONE.
C-----------------------------------------------------------------------
C
      EB=.TRUE.
C
C-----------------------------------------------------------------------
C  IN HALFWORD MODE, NWDS AND NSLAB MUST BE EVEN FOR I/O REASONS
C-----------------------------------------------------------------------
C
      IF(MOD(NWDS,2).NE.0 .OR. MOD(NSLAB,2).NE.0) THEN
        PRINT 9831
        STOP 9831
      ENDIF
```

```
 9831 FORMAT(1X,'NWDS & NSLAB MUST BE EVEN FOR QDAM IN HALFWORD MODE')
C
C-----------------------------------------------------------------------
C  INITIALIZE VARIOUS QUANTITIES
C-----------------------------------------------------------------------
C
      NWRITE=1000000
      NDW=1000000
      NTSI=1
      NA=0
      NB=0
      NC=0
      KONTRL=11
      KFLDS=12
      LABS(1)=13
      LABS(2)=14
      LABS(3)=15
      PI=3.1415927
      OMEGA=3.1415927/43082.
      RADIUS=6370.E5
      RADIAN=57.29578
      GRAV=980.6
C
C-----------------------------------------------------------------------
C  SET THE LATITUDE (IN DEGREES) OF THE SOUTHERN WALL
C-----------------------------------------------------------------------
C
      SWLDEG=??
C
C-----------------------------------------------------------------------
C  SET Y DIMENSION OF BOXES IN DEGREES AND CONVERT TO CENTIMETERS
C-----------------------------------------------------------------------
C
      DO 52 J=2,JMTM1
        DYT(J)=??
        DYT(J)=DYT(J)*RADIUS/RADIAN
 52   CONTINUE
      DYT(1)=DYT(2)
      DYT(JMT)=DYT(JMTM1)
C
C-----------------------------------------------------------------------
C  SET X DIMENSION OF BOXES IN DEGREES AND CONVERT TO CENTIMETERS
C-----------------------------------------------------------------------
C
      DO 57 I=2,IMTM1
        DXT(I)=??
        DXT(I)=DXT(I)*RADIUS/RADIAN
 57   CONTINUE
      DXT(1)=DXT(2)
      DXT(IMT)=DXT(IMTM1)
C
C  SET CYCLIC CONDITIONS ON DXT
C
      DXT(1)=DXT(IMTM1)
      DXT(IMT)=DXT(2)
C
C-----------------------------------------------------------------------
C  SET Z DIMENSION OF BOXES (IN CENTIMETERS)
C-----------------------------------------------------------------------
C
      DZ(1)=??
C
C-----------------------------------------------------------------------
C  READ IN RUN PARAMETERS
C-----------------------------------------------------------------------
C
      READ (5,CONTRL)
      WRITE(6,CONTRL)
      READ (5,EDDY)
      WRITE(6,EDDY)
      READ (5,TSTEPS)
      WRITE(6,TSTEPS)
      READ (5,PARMS)
      WRITE(6,PARMS)
      AM=AMF
      AH=AHF
      FKPM=FKPMF
      FKPH=FKPHF
      DTTS=DTTSF
      DTUV=DTUVF
      DTSF=DTSFF
      ACOR=ACORF
      SOR=SORF
      CRIT=CRITF
```

```
C
C-----------------------------------------------------------------------
C  COMPUTE AUXILIARY ARRAYS BASED UPON THE SPACING SPECIFIED ABOVE
C-----------------------------------------------------------------------
C
      DO 100 K=1,KM
        C2DZ(K)=2.0*DZ(K)
        DZ2R(K)=1.0/C2DZ(K)
  100 CONTINUE
      DZZ(1)=0.5*DZ(1)
      ZDZ(1)=DZ(1)
      DO 110 K=2,KM
        DZZ(K)=0.5*(DZ(K-1)+DZ(K))
        ZDZ(K)=ZDZ(K-1)+DZ(K)
  110 CONTINUE
      DZZ(KM+1)=0.5*DZ(KM)
      DZZ2R(KMP1)=0.5/DZZ(KMP1)
      ZDZZ(1)=DZZ(1)
      DO 120 K=1,KM
        DZZ2R(K)=.5/DZZ(K)
        ZDZZ(K+1)=ZDZZ(K)+DZZ(K+1)
        EEH(K)=FKPH/(DZ(K)*DZZ(K))
        FFH(K)=FKPH/(DZ(K)*DZZ(K+1))
        EEM(K)=FKPM/(DZ(K)*DZZ(K))
        FFM(K)=FKPM/(DZ(K)*DZZ(K+1))
        KAR(K)=K
  120 CONTINUE
      PHI(1)=SWLDEG/RADIAN
      PHIT(1)=PHI(1)-.5*DYT(1)/RADIUS
      SUMDY=PHI(1)
      DYU(JMT)=DYT(JMT)
      DO 130 J=1,JMT
        IF(J.NE.JMT) DYU(J)=.5*(DYT(J)+DYT(J+1))
        DYTR(J)=1./DYT(J)
        DYT2R(J)=.5/DYT(J)
        DYT4R(J)=.25/DYT(J)
        DYUR(J)=1./DYU(J)
        DYU2R(J)=.5/DYU(J)
        DYU4R(J)=.25/DYU(J)
        IF(J.NE.JMT) SUMDY=SUMDY+DYT(J+1)/RADIUS
        IF(J.NE.JMT) PHI(J+1)=SUMDY
        IF(J.NE.1) PHIT(J)=.5*(PHI(J-1)+PHI(J))
        CST(J)=COS(PHIT(J))
        CS (J)=COS(PHI  (J))
        SINE(J)=SIN(PHI(J))
        CSTR(J)=1.0/CST(J)
        CSR(J)=1.0/CS(J)
        TNG(J)=SINE(J)/CS(J)
  130 CONTINUE
      DXU(IMT)=DXT(IMT)
      DXU(IMT)=.5*(DXT(2)+DXT(3))
      DO 140  I=1,IMT
        IF(I.NE.IMT) DXU(I)=.5*(DXT(I)+DXT(I+1))
        DXTR(I)=1./DXT(I)
        DXT2R(I)=.5/DXT(I)
        DXT4R(I)=.25/DXT(I)
        DXUR(I)=1./DXU(I)
        DXU2R(I)=.5/DXU(I)
        DXU4R(I)=.25/DXU(I)
  140 CONTINUE
C
C-----------------------------------------------------------------------
C  CREATE A BIT ARRAY FOR LATER WHERE STATEMENTS
C  (..NOTE.. "KEVENBV" AND "KODDBV" ARE EQUIVALENCED WITH "KALTBV")
C-----------------------------------------------------------------------
C
      LENV=2*IMT*KM
      KALTBV(1,1,1;LENV)=B'0'
      KALTBV(1,1,1;LENV)=.NOT.KALTBV(1,1,1;LENV)
      DO 145 K=1,KM
        IF (K-2*(K/2).NE.0) THEN
          KEVENBV(1,K;IMT)=B'0'
        ELSE
          KODDBV(1,K;IMT)=B'0'
        ENDIF
  145 CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE SIN AND COS VALUES FOR VECTOR CORRECTION BEFORE FILTER
C-----------------------------------------------------------------------
C
      FX=1.0E-10
      FXA=DXT(1)/RADIUS
      DO 670 I=2,IMUM1
```

```
      FXB=FXA*FLOAT(I-2)
      SPSIN(I)=SIN(FXB)
      SPCOS(I)=COS(FXB)
      IF(ABS(SPSIN(I)).LT.FX)SPSIN(I)=0.0
      IF(ABS(SPCOS(I)).LT.FX)SPCOS(I)=0.0
  670 CONTINUE
      SPSIN(1)=0.0
      SPCOS(1)=0.0
      SPSIN(IMU)=0.0
      SPCOS(IMU)=0.0
C
C-------------------------------------------------------------------
C  PRINT GRID GEOMETRY ARRAYS
C-------------------------------------------------------------------
C
      PRINT 9701
 9701 FORMAT(50HO GRID BOX THICKNESS  'DZ'                          )
      PRINT 970, DZ
      PRINT 9702
 9702 FORMAT(50HO GRID POINT SEPARATION  'DZZ'                      )
      PRINT 970, DZZ
      PRINT 9703
 9703 FORMAT(50HO DEPTH OF BOX BOTTOM  'ZDZ'                        )
      PRINT 970, ZDZ
      PRINT 9704
 9704 FORMAT(50HO DEPTH OF GRID POINT  'ZDZZ'                       )
      PRINT 970, ZDZZ
      PRINT 9705
 9705 FORMAT(50HO LATITUDE OF T,S POINTS (RADIANS)  'PHIT'          )
      PRINT 970, PHIT
      PRINT 9706
 9706 FORMAT(50HO LATITUDE OF U,V POINTS (RADIANS)  'PHI'           )
      PRINT 970, PHI
      PRINT 9707
 9707 FORMAT(50HO COSINE OF T,S LATITUDE  'CST'                     )
      PRINT 970, CST
      PRINT 9708
 9708 FORMAT(50HO COSINE OF U,V LATITUDE  'CS'                      )
      PRINT 970, CS
      PRINT 9709
 9709 FORMAT(50HO SINE OF U,V LATITUDE  'SINE'                      )
      PRINT 970, SINE
 970  FORMAT(1X,10E13.5)
C
C-------------------------------------------------------------------
C  OPEN THE DISK DATASETS
C-------------------------------------------------------------------
C
      CALL OSTART(KONTRL,20,20,1)
      CALL OSTART(KFLDS,NKFLDS*NWDS,NWDS,1)
      NBUF=2
      CALL OSTART(LABS(1),JMT*NSLAB,NSLAB,NBUF)
      CALL OSTART(LABS(2),JMT*NSLAB,NSLAB,NBUF)
      CALL OSTART(LABS(3),JMT*NSLAB,NSLAB,NBUF)
      IF(NFIRST.EQ.0) CALL ORD(21)
C
C************************************************************************
C  END INTRODUCTORY SECTION  ********************************************
C************************************************************************
C
C************************************************************************
C  BEGIN SECTION WHICH IS EXECUTED ONLY WHEN STARTING  ******************
C       A RUN FROM SCRATCH                             ******************
C************************************************************************
C
      IF (NFIRST.EQ.0) GO TO 160
C
C-------------------------------------------------------------------
C  SET MAXIMUM LEVEL INDICATORS FOR TOPOGRAPHY
C-------------------------------------------------------------------
C
      DO 690 J=1,JMT
      DO 690 I=1,IMT
        FKMP(I,J)=0
        FKMQ(I,J)=0
        FKMZ(I,J)=0
 690  CONTINUE
C
C  1ST, SET NUMBER OF VERTICAL LEVELS FOR T POINTS
C
      DO 700 J=2,JMTM1
      DO 700 I=2,IMTM1
        FKMP(I,J)=??
 700  CONTINUE
```

```
C
C  SET CYCLIC BOUNDARY CONDITIONS
C
      DO 728 J=1,JMT
        FKMP(  1,J)=FKMP(IMTM1,J)
        FKMP(IMT,J)=FKMP(    2,J)
 728  CONTINUE
C
C  2ND, COMPUTE NUMBER OF VERTICAL LEVELS AT EACH U,V POINT
C
      DO 730 J=1,JMTM1
      DO 730 I=1,IMTM1
       FKMQ(I,J)=MIN(FKMP(I,J),FKMP(I+1,J),FKMP(I,J+1),FKMP(I+1,J+1))
 730  CONTINUE
C
C  SET CYCLIC CONDITIONS
C
      DO 732 J=1,JMT
        FKMQ(IMT,J)=FKMQ(2,J)
 732  CONTINUE
C
C  3RD, COMPUTE AN ARRAY TO INDICATE "INTERIOR" GRID BOXES
C
      DO 740 J=2,JMTM1
      DO 740 I=2,IMU
        FKMZ(I,J)=MIN(FKMQ(I-1,J-1),FKMQ(I,J-1),FKMQ(I-1,J),FKMQ(I,J))
 740  CONTINUE
C
C  SET CYCLIC CONDITIONS
C
      DO 742 J=1,JMT
        FKMZ(1,J)=FKMZ(IMTM1,J)
 742  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE START & END INDICES FOR STREAM FUNCTION CALCULATIONS
C-----------------------------------------------------------------------
C
      DO 750 N=1,NDICES
        ISZ(N,1)=0
 750  CONTINUE
      LSEGP=LSEG+1
      DO 780 J=3,JSCAN
        L=1
      DO 780 I=2,IMUM1
        IF(FKMZ(I-1,J).EQ.0.  .AND. FKMZ(I,J).NE.0.) ISZ(J,L)=I
        IF(I.EQ.2 .AND. FKMZ(IMT,J).NE.0.) ISZ(J,L)=2
        IF(FKMZ(I,J).NE.0.  .AND. FKMZ(I+1,J).EQ.0.) IEZ(J,L)=I
        IF(I.EQ.IMUM1 .AND. FKMZ(I+1,J).NE.0.) IEZ(J,L)=I
        IF(FKMZ(I,J).NE.0.  .AND. FKMZ(I+1,J).EQ.0.) L=L+1
        IF(L.GT.LSEGP) STOP 780
 780  CONTINUE
C
C-----------------------------------------------------------------------
C  FIND AND PRINT START & END INDICES FOR FILTERING
C-----------------------------------------------------------------------
C
      PRINT 833
      IF (LSEGF.GT.11) PRINT 834
      PRINT 835
      CALL FINDEX(FKMP,NJTBFT,KM,JFT1,JFT2,IMT,ISTF,IETF)
      PRINT 836
      CALL FINDEX(FKMQ,NJTBFU,KM,JFU1,JFU2,IMU,ISUF,IEUF)
      PRINT 837
      CALL FINDEX(FKMZ,NJTBFU, 1,JFU1,JFU2,IMT,ISZF,IEZF)
  833 FORMAT (1H1,'START AND END INDICES FOR FOURIER FILTERING:'/)
  834 FORMAT (1X,'ONLY 11 SETS OF INDICES FIT ACCROSS THE PAGE.',
     *        '  OTHERS WILL NOT BE PRINTED.'/)
  835 FORMAT (///1X,'FILTERING INDICES FOR T,S:')
  836 FORMAT (///1X,'FILTERING INDICES FOR U,V:')
  837 FORMAT (///1X,'FILTERING INDICES FOR STREAM FUNCTION:')
C
C-----------------------------------------------------------------------
C  COMPUTE FIELD OF RECIPROCAL DEPTH
C-----------------------------------------------------------------------
C
      DO 790 J=1,JMT
      DO 790 I=1,IMT
        HR(I,J)=0.0
        IF(FKMQ(I,J).NE.0) HR(I,J)=1./ZDZ(INT(FKMQ(I,J)))
 790  CONTINUE
C
C  SET SYMMETRY CONDITIONS
C
```

```
      DO 792 I=1,IMT
        HR(I,JMT)=HR(I,JMTM2)
 792  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE THE SURFACE AREA AND VOLUME OF THE OCEAN
C-----------------------------------------------------------------------
C
      AREA=0.0
      VOLUME=0.0
      DO 800 J=2,JMTM1
      DO 800 I=2,IMTM1
        IF(FKMP(I,J).GT.0) THEN
          AREA=AREA+CST(J)*DXT(I)*DYT(J)
          VOLUME=VOLUME+CST(J)*DXT(I)*DYT(J)*ZDZ(INT(FKMP(I,J)))
        ENDIF
 800  CONTINUE
C
C-----------------------------------------------------------------------
C  PRINT TOPOGRAPHY MAP
C   (..NOTE.. THE NUMBER OF LEVELS ARE PRINTED IN HEX:
C             A DOT SUPERIMPOSED ===> ADD AN ADDITIONAL 16)
C-----------------------------------------------------------------------
C
      PRINT 950
 950  FORMAT(50H1 NUMBER OF LEVELS AT T,S POINTS AND U,V POINTS    )
      DO 810 IBK=1,IMT,65
        PRINT 960
 960    FORMAT(/)
        ISP=IBK
        IEPT=IBK+64
        IEPU=IBK+64
        IF(IEPT.GT.IMT) IEPT=IMT
        IF(IEPU.GT.IMU) IEPU=IMU
      DO 810 JREV=1,JMT
        J=JMT-JREV+1
        IF(J.NE.JMT) THEN
          DO 968 I=1,IMT
            KPR(I)=FKMQ(I,J)
 968      CONTINUE
          PRINT 972, (KPR(I),I=ISP,IEPU)
 972      FORMAT(2X,65(1X,Z1))
          DO 969 I=1,IMT
            ABT(I)=BLK
 969      CONTINUE
          DO 952 I=ISP,IEPU
            IF(KPR(I).GT.15)ABT(I)=DOT
 952      CONTINUE
          PRINT 971,(ABT(I),I=ISP,IEPU)
 971      FORMAT(2H+ ,65(1X,A1))
        ENDIF
        DO 953 I=1,IMT
          KPR(I)=FKMP(I,J)
 953    CONTINUE
        PRINT 982, (KPR(I),I=ISP,IEPT)
 982    FORMAT(1X,65(1X,Z1))
        DO 979 I=1,IMT
          ABT(I)=BLK
 979    CONTINUE
        DO 954 I=ISP,IEPT
          IF(KPR(I).GT.15)ABT(I)=DOT
 954    CONTINUE
        PRINT 981,(ABT(I),I=ISP,IEPT)
 981    FORMAT(1H+,65(1X,A1))
 810  CONTINUE
C
C-----------------------------------------------------------------------
C  PRINT AREA AND VOLUME OF THE OCEAN, AS WELL AS START & END
C  INDICES FOR THE STREAM FUNCTION CALCULATION
C-----------------------------------------------------------------------
C
      PRINT 940, AREA,VOLUME
 940  FORMAT(//,15H SURFACE AREA =,1PE13.6,5X,9H VOLUME =,1PE13.6)
      PRINT 9502
 9502 FORMAT(43H1 START AND END INDICES FOR STREAM FUNCTION)
      DO 830 JREV=1,JMT
        J=JMT-JREV+1
        PRINT 930,J,(ISZ(J,L),IEZ(J,L),L=1,LSEG)
 930    FORMAT(' J=',I3,5X,5(2I5,10X))
 830  CONTINUE
C
C-----------------------------------------------------------------------
C  READ IN INITIAL TRACER VALUES
C  AND ISLAND BOX CORNER POINT INDICES
```

```
C-----------------------------------------------------------------------
C
      READ (5,TSPROF)
      WRITE(6,TSPROF)
      DO 832 M=1,NT
      DO 832 K=1,KM
        TINIT(K,M)=TINITF(K,M)
 832  CONTINUE
      READ (5,IBOX)
      WRITE(6,IBOX)
C
C-----------------------------------------------------------------------
C  INITIALIZE SLAB DATA ON DISK
C-----------------------------------------------------------------------
C
      DO 880 J=1,JMT
        DO 840 I=1,IMT
C
C  SET WIND STRESS TO SPECIFIED DISTRIBUTION
C
          WSX(I)=??
          WSY(I)=??
C
C  SET MAXIMUM LEVEL INDICATORS TO VALUES COMPUTED ABOVE
C
          FKMT(I)=FKMP(I,J)
          FKMU(I)=FKMQ(I,J)
 840    CONTINUE
        DO 842 K=1,KM
        DO 842 I=1,IMT
C
C  SET INTERNAL MODE VELOCITIES TO ZERO
C
          UB(I,K)=0.0
          U (I,K)=0.0
          VB(I,K)=0.0
          V (I,K)=0.0
        DO 842 M=1,NT
          TB(I,K,M)=0.0
          T (I,K,M)=0.0
 842    CONTINUE
C
C  SET TRACERS OVER OCEAN POINTS TO SPECIFIED VALUES
C
        DO 870 I=1,IMT
          KZ=FKMP(I,J)
          IF(KZ.NE.0) THEN
            DO 860 K=1,KZ
            DO 860 M=1,NT
              TB(I,K,M)=TINIT(K,M)
              IF(M.EQ.2) TB(I,K,M)=TINIT(K,M)-0.035
              T(I,K,M)=TB(I,K,M)
 860        CONTINUE
          ENDIF
 870    CONTINUE
C
C  SEND THE INITIAL SLABS TO DISK
C
        CALL OPUT(LABS(1),NSLAB,(J-1)*NSLAB+1,TB)
        CALL OPUT(LABS(2),NSLAB,(J-1)*NSLAB+1,T )
        CALL OPUT(LABS(3),NSLAB,(J-1)*NSLAB+1,T )
 880  CONTINUE
C
C-----------------------------------------------------------------------
C   INITIALIZE REMAINDER OF DISK
C-----------------------------------------------------------------------
C
C  SET INITIAL STREAM FUNCTION TO ZERO
C
      DO 890 J=1,JMT
      DO 890 I=1,IMT
        PB(I,J)=0.0
        P (I,J)=0.0
 890  CONTINUE
C
C  SEND INITIALIZED DATA TO DISC
C
      CALL OPUT(KFLDS,NWDS,1,PB)
      CALL OPUT(KFLDS,NWDS,NWDS+1,P)
      CALL OPUT(KFLDS,NWDS,3*NWDS+1,HR)
      CALL OPUT(KFLDS,NWDS,4*NWDS+1,P)
      CALL OPUT(KFLDS,NWDS,5*NWDS+1,P)
C
C  CONVERT START AND END INDICES TO REAL (THIS IS DONE TO ACCOMODATE
```

```
C        THE OPTION OF RUNNING THE MODEL IN HALFWORD MODE; ALSO,
C        NOTE THAT "FINS" IS EQUIVALENCED WITH "P")
C
      DO 164 N=1,NDICES
        FINS(N)=ISZ(N,1)
 164  CONTINUE
C
C  SET TIMESTEP COUNTER AND TOTAL ELAPSED TIME TO ZERO
C
      ITT=0
      TTSEC=0.0
C
C  SEND REMAINDER OF INITIALIZED DATA TO DISC
C
      CALL OPUT(KONTRL,20,1,ITT)
      CALL OPUT(KFLDS,NDICES,6*NWDS+1,FINS)
C
C***********************************************************************
C  END SECTION TO START FROM SCRATCH  **********************************
C***********************************************************************
C
C***********************************************************************
C  BEGIN SECTION TO TIMESTEP THE MODEL  ********************************
C***********************************************************************
C
C-----------------------------------------------------------------------
C  READ DISK DATA INTO MEMORY FOR STARTUP
C-----------------------------------------------------------------------
C
C  READ IN TIMESTEP COUNTER, TOTAL ELAPSED TIME, AREA AND VOLUME
C
 160  CALL OGET(KONTRL,20,1,ITT)
C
C  READ IN START AND END INDICES AND CONVERT TO INTEGERS
C
      CALL OGET(KFLDS,NDICES,6*NWDS+1,FINS)
      DO 165 N=1,NDICES
        ISZ(N,1)=FINS(N)
 165  CONTINUE
C
C  COMPUTE PERMUTING DISC INDICATORS AND READ IN 2 LEVELS OF
C       STREAM FUNCTION AS WELL AS RECIPROCAL DEPTH.
C
      NDISK =MOD(ITT+0,3)+1
      NDISKA=MOD(ITT+1,3)+1
      CALL OGET(KFLDS,NWDS,(NDISK-1)*NWDS+1,PB)
      CALL OGET(KFLDS,NWDS,(NDISKA-1)*NWDS+1,P)
      CALL OGET(KFLDS,NWDS,3*NWDS+1,HR)
C
C-----------------------------------------------------------------------
C  INITIALIZE SEVERAL VARIABLES TO ZERO TO AVOID AN "UNINITIALIZED
C       VARIABLE" TYPE OF ERROR LATER WHERE, FOR PURPOSES OF VEC-
C       TORIZATION, THE COMPUTATION PROCEEDS ACROSS LAND POINTS
C-----------------------------------------------------------------------
C
      DO 168 I=1,IMT
        UUNDER(I)=0.0
        VUNDER(I)=0.0
        DO 166 J=1,JMT
          ZTD(I,J)=0.0
 166    CONTINUE
        DO 167 K=1,KMP1
          TEMPA(I,K)=0.0
          TEMPB(I,K)=0.0
 167    CONTINUE
      DO 168 M=1,NT
      DO 168 K=1,KMP2
        TDIF(I,K,M)=0.0
 168  CONTINUE
C
      CALL GETIME(TO,TIME(1))
C
C-----------------------------------------------------------------------
C  PROCEED WITH TIMESTEPPING UNTIL THE SPECIFIED NUMBER
C       OF STEPS HAVE BEEN TAKEN
C-----------------------------------------------------------------------
C
 200  CALL STEP
C
C-----------------------------------------------------------------------
C  SAVE RESTART DATA EVERY NWRITE TIMESTEPS, AND AT END OF RUN IF NA=1
C-----------------------------------------------------------------------
C
      IF(MOD(ITT,NWRITE).EQ.0) CALL OWRT(22)
```

```
      IF(ITT.EQ.NLAST .AND. NA.EQ.1) CALL OWRT(21)
      IF(ITT.LT.NLAST) GO TO 200
C
C=============================================================================
C  END TIMESTEPPING OF THE MODEL  =============================================
C=============================================================================
C
C-----------------------------------------------------------------------
C  COMPLETE THE TIMING ANALYSIS AND PRINT IT
C-----------------------------------------------------------------------
C
      CALL GETIME(T1,T2)
      T3=100./(T1-T3)
      TIME(2)=T1-T0-TIME(3)-TIME(4)-TIME(5)
      TIME(3)=TIME(3)-TIME(6)
      TIME(4)=TIME(4)-TIME(7)
      TIME(5)=TIME(5)-TIME(8)
      DO 3325 LTIME=1,10
 3325 TIME(LTIME)=TIME(LTIME)*T3
      PRINT 9601
 9601 FORMAT('0TIMING ANALYSIS',/,7X,'MAIN',6X,'STEP',4X,'CLINIC',1X,
     *'STATE(CL)',4X,'TRACER',1X,'STATE(TR)',5X,'RELAX',5X,'SCANS')
      PRINT 9602,TIME(1),TIME(2),TIME(3),TIME(6),TIME(4),TIME(7),
     *TIME(5),TIME(8)
 9602 FORMAT(1X,13(F9.2,'%'))
      PRINT 9603,TIME(9)
 9603 FORMAT('0% OF TOTAL TIME IN FILTER:',F10.2)
C
C-----------------------------------------------------------------------
C  CLOSE THE DISC DATASETS
C-----------------------------------------------------------------------
C
      CALL OCLOSE(KONTRL)
      CALL OCLOSE(KFLDS)
      CALL OCLOSE(LABS(1))
      CALL OCLOSE(LABS(2))
      CALL OCLOSE(LABS(3))
      STOP 13000
      END
```

```
*DECK STEP
      SUBROUTINE STEP
C
C*************************************************************************
C                                                              **********
C  STEP IS CALLED ONCE PER TIMESTEP.  IT INITIALIZES VARIOUS   **********
C       QUANTITIES, BOOTSTRAPS THE BASIC ROW BY ROW COMPUTATION **********
C       OF PROGNOSTIC VARIABLES, MANAGES THE I/O FOR THE LATTER, **********
C       AND PERFORMS VARIOUS ANALYSIS PROCEDURES ON THE PROGRESSING **********
C       SOLUTION.                                               **********
C                                                              **********
C*************************************************************************
C
C---------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C---------------------------------------------------------------------
C
*CALL PARAM
*CALL FULLWD
*CALL SCALAR
*CALL ONEDIM
*CALL FIELDS
*CALL WRKSPA
*CALL TIME
C
C---------------------------------------------------------------------
C  DEMENSION AND EQUIVALENCE LOCAL DATA
C---------------------------------------------------------------------
C
      DIMENSION TBSLAB(NSLAB),TSLAB(NSLAB),VBR(KM),TBRN(KM,NT),
     *          TBRS(KM,NT),TTN(8,JMT,NTMIN2),TMT(JMT,KM)
      EQUIVALENCE (TBSLAB,TB),(TSLAB,T)
      CHARACTER C8TIME*8,CTIME*10
C
C---------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C---------------------------------------------------------------------
C
      CALL GETIME(TO,T1)
      CALL Q5TIME('MASTER=',C8TIME)
      PRINT 110,TO,C8TIME
 110  FORMAT (' ACC TIME(SEC)=',F10.2,5X,Z16)
C
C*************************************************************************
C  BEGIN SECTION FOR THE INITIALIZATION OF   *****************************
C  VARIOUS QUANTITIES ON EACH TIMESTEP       *****************************
C*************************************************************************
C
C---------------------------------------------------------------------
C  UPDATE TIMESTEP COUNTER AND TOTAL ELAPSED TIME
C---------------------------------------------------------------------
C
      ITT=ITT+1
      TTSEC=TTSEC+DTTS
C
C---------------------------------------------------------------------
C  UPDATE PERMUTING DISC I/O UNITS
C---------------------------------------------------------------------
C
      NDISKB= MOD(ITT+2,3)+1
      NDISK = MOD(ITT+0,3)+1
      NDISKA= MOD(ITT+1,3)+1
C
C---------------------------------------------------------------------
C  ADJUST VARIOUS QUANTITIES FOR MIXING TIMESTEP
C---------------------------------------------------------------------
C
      MIX=0
      MXP=0
      C2DTTS=2.0*DTTS
      C2DTUV=2.0*DTUV
      C2DTSF=2.0*DTSF
      IF(MOD(ITT,NMIX).EQ.1) THEN
        MIX=1
        C2DTTS=DTTS
        C2DTUV=DTUV
        C2DTSF=DTSF
        DO 170 J=1,JMT
        DO 170 I=1,IMT
          PB(I,J)=P(I,J)
 170    CONTINUE
      ENDIF
 182  CONTINUE
C
```

```
C-----------------------------------------------------------------------
C  ESTABLISH OVER DIMENSIONED ARRAYS FOR VECTORIZATION
C-----------------------------------------------------------------------
C
      DO 184 K=1,KM
      DO 184 I=1,IMT
        DXTQ  (I,K)=DXT  (I)
        DXT4RQ(I,K)=DXT4R(I)
        DXUQ  (I,K)=DXU  (I)
        DXU2RQ(I,K)=DXU2R(I)
        DZZQ  (I,K)=DZZ  (K)
        DZ2RQ (I,K)=DZ2R (K)
        DZZ2RQ(I,K)=DZZ2R(K)
        C2DZQ (I,K)=C2DZ (K)
        EEHQ  (I,K)=EEH  (K)
        EEMQ  (I,K)=EEM  (K)
        FFHQ  (I,K)=FFH  (K)
        FFMQ  (I,K)=FFM  (K)
 184  CONTINUE
C
C-----------------------------------------------------------------------
C  LOAD COEFFICIENT ARRAYS FOR SUBSEQUENT CALLS TO "STATE" AND "STATEC"
C-----------------------------------------------------------------------
C
      CALL STINIT
C
C-----------------------------------------------------------------------
C  QUEUE UP DISK READS FOR THIS TIMESTEP
C-----------------------------------------------------------------------
C
      DO 120 J=2,JMTM1
        CALL OFIND(LABS(NDISKB),NSLAB,(J-1)*NSLAB+1)
        CALL OFIND(LABS(NDISK),NSLAB,(J-1)*NSLAB+1)
 120  CONTINUE
C
C-----------------------------------------------------------------------
C  INITIALIZE VARIOUS QUANTITIES USED FOR ANALYSIS OF THE SOLUTION
C-----------------------------------------------------------------------
C
      EKTOT=0.0
      DO 130 M=1,NT
        DTABS(M)=0.0
        TVAR(M)=0.0
 130  CONTINUE
      NERGY=0
      IF(MOD(ITT,NNERGY).EQ.0) NERGY=1
      IF(NERGY.EQ.1 .AND. MXP.EQ.0) THEN
        BUOY=0.0
        DO 190 LL=1,8
          ENGINT(LL)=0.0
          ENGEXT(LL)=0.0
        DO 190 I=1,IMT
          ZUSENG(I,LL)=0.0
          ZVSENG(I,LL)=0.0
 190    CONTINUE
        DO 192 M=1,NT
        DO 192 LL=1,6
          TTDTOT(LL,M)=0.0
 192    CONTINUE
        DO 194 J=1,JMT
          DO 193 M=1,NTMIN2
          DO 193 LL=1,8
            TTN(LL,J,M)=0.0
 193      CONTINUE
        DO 194 K=1,KM
          TMT(J,K)=0.0
 194    CONTINUE
      ENDIF
C
C=======================================================================
C  END OF SECTION FOR INITIALIZATION  ==================================
C=======================================================================
C
C=======================================================================
C  BEGIN A BOOTSTRAP PROCEDURE TO PREPARE FOR THE  =====================
C  ROW-BY-ROW COMPUTATION OF PROGNOSTIC VARIABLES  =====================
C=======================================================================
C
C-----------------------------------------------------------------------
C  FETCH DATA FOR ROW 2 FROM THE DISC
C-----------------------------------------------------------------------
C
      CALL OGET(LABS(NDISKB),NSLAB,NSLAB+1,TBP)
      CALL OGET(LABS(NDISK ),NSLAB,NSLAB+1,TP )
```

```
C
C------------------------------------------------------------------------
C  SWITCH SLAB INCIDENTAL DATA INTO CORRECT SLAB AFTER READIN
C------------------------------------------------------------------------
C
      IF(MOD(ITT,2)+MXP.NE.1) THEN
        DO 220 N=1,NSWICH
          BCON(N,1)=FKMUP(N)
          FKMUP(N)=FKMTP(N)
          FKMTP(N)=BCON(N,1)
 220    CONTINUE
      ENDIF
C
C------------------------------------------------------------------------
C  CONVERT MAXIMUM LEVEL INDICATORS TO INTEGERS
C------------------------------------------------------------------------
C
      DO 222 I=1,IMT
        KMTP(I)=FKMTP(I)
        KMUP(I)=FKMUP(I)
 222  CONTINUE
C
C------------------------------------------------------------------------
C  MOVE TAU-1 DATA TO TAU LEVEL ON A MIXING TIMESTEP
C------------------------------------------------------------------------
C
      IF(MIX.EQ.1) THEN
        DO 224 M=1,NT
        DO 224 K=1,KM
        DO 224 I=1,IMT
          TBP(I,K,M)=TP(I,K,M)
 224    CONTINUE
        DO 226 K=1,KM
        DO 226 I=1,IMT
          UBP(I,K)=UP(I,K)
          VBP(I,K)=VP(I,K)
 226    CONTINUE
      ENDIF
C
C------------------------------------------------------------------------
C  INITIALIZE ARRAYS FOR FIRST CALLS TO CLINIC AND TRACER
C------------------------------------------------------------------------
C
      FX=0.0
      DO 240 N=1,NSLAB
        TBSLAB(N)=FX
        T SLAB(N)=FX
 240  CONTINUE
      DO 250 K=1,KM
      DO 250 I=1,IMT
        FVST(I,K)=FX
        RHOS(I,K)=FX
        FMM (I,K)=FX
        FM  (I,K)=FX
 250  CONTINUE
C
C------------------------------------------------------------------------
C  CONSTRUCT MASK ARRAY FOR ROW 2 TRACERS
C------------------------------------------------------------------------
C
      DO 254 K=1,KM
      DO 254 I=1,IMT
        IF(KMTP(I).GE.KAR(K)) THEN
          FMP(I,K)=1.0
        ELSE
          FMP(I,K)=0.0
        ENDIF
      FXA=0.0
      FXB=1.0
      WHERE (KMTP(1;IMT).GE.KAR(K))
        FMP(1,K;IMT)=FXB
      OTHERWISE
        FMP(1,K;IMT)=FXA
      END WHERE
 254  CONTINUE
C
C------------------------------------------------------------------------
C  SET VORTICITY COMPUTATION ARRAYS AT SOUTHERN WALL
C------------------------------------------------------------------------
C
      FX=0.0
      DO 258 I=1,IMT
        ZUS(I)=FX
        ZVS(I)=FX
```

```
 258  CONTINUE
C
C-----------------------------------------------------------------------
C  SAVE INTERNAL MODE VELOCITIES FOR ROW 2
C  AND COMPUTE ADVECTIVE COEFFICIENT FOR SOUTH FACE OF ROW 2 U,V BOXES
C-----------------------------------------------------------------------
C
      FX=DYU2R(2)*CSR(2)*CST(2)*0.5
      DO 260 K=1,KM
      DO 260 I=1,IMT
        UCLIN(I,K)=UP(I,K)
        VCLIN(I,K)=VP(I,K)
        FVSU(I,K)=(VP(I,K)+V(I,K))*FX
 260  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE EXTERNAL MODE VELOCITIES FOR ROW 2
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE FOR TAU-1 TIME LEVEL
C
      J=1
      DO 270 I=1,IMTM1
        DIAG1=PB(I+1,J+2)-PB(I  ,J+1)
        DIAG2=PB(I  ,J+2)-PB(I+1,J+1)
        SFUB(I)=-(DIAG1+DIAG2)*DYU2R(J+1)*HR(I,J+1)
        SFVB(I)= (DIAG1-DIAG2)*DXU2R(I  )*HR(I,J+1)*CSR(J+1)
 270  CONTINUE
C
C  2ND, COMPUTE FOR TAU TIME LEVEL
C
      DO 280 I=1,IMTM1
        DIAG1=P (I+1,J+2)-P (I  ,J+1)
        DIAG2=P (I  ,J+2)-P (I+1,J+1)
        SFU (I)=-(DIAG1+DIAG2)*DYU2R(J+1)*HR(I,J+1)
        SFV (I)= (DIAG1-DIAG2)*DXU2R(I  )*HR(I,J+1)*CSR(J+1)
 280  CONTINUE
C
C  3RD, SET CYCLIC BOUNDARY CONDITIONS
C
      SFUB(IMT)=SFUB(2)
      SFVB(IMT)=SFVB(2)
      SFU (IMT)=SFU (2)
      SFV (IMT)=SFV (2)
C
C-----------------------------------------------------------------------
C  ADD EXTERNAL MODE TO INTERNAL MODE FOR ROW 2  (OCEAN PTS. ONLY)
C-----------------------------------------------------------------------
C
      DO 300 K=1,KM
      DO 300 I=1,IMU
        IF(KMUP(I).GE.KAR(K)) THEN
          UBP(I,K)=UBP(I,K)+SFUB(I)
          VBP(I,K)=VBP(I,K)+SFVB(I)
          UP (I,K)=UP (I,K)+SFU (I)
          VP (I,K)=VP (I,K)+SFV (I)
        ENDIF
        WHERE (KMUP(1;IMU).GE.KAR(K))
          UBP(1,K;IMU)=UBP(1,K;IMU)+SFUB(1;IMU)
          VBP(1,K;IMU)=VBP(1,K;IMU)+SFVB(1;IMU)
          UP (1,K;IMU)=UP (1,K;IMU)+SFU (1;IMU)
          VP (1,K;IMU)=VP (1,K;IMU)+SFV (1;IMU)
        ENDWHERE
 300  CONTINUE
C
C-----------------------------------------------------------------------
C  ACCUMULATE KINETIC ENERGY FROM ROW 2 EVERY NTSI TIMESTEPS
C-----------------------------------------------------------------------
C
      IF(MOD(ITT,NTSI).EQ.0) THEN
        DO 305 K=1,KM
          FX=0.5*CS(J+1)*DYU(J+1)*DZ(K)
        DO 305 I=2,IMUM1
          EKTOT=EKTOT+(UP(I,K)*UP(I,K)+VP(I,K)*VP(I,K))*FX*DXU(I)
 305    CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  COMPUTE DENSITY OF ROW 2
C-----------------------------------------------------------------------
C
      CALL STATE(TP(1,1,1),TP(1,1,2),RHOS,TDIF(1,1,1),TDIF(1,1,2))
C
C  SET CYCLIC BOUNDARY CONDITIONS
```

```
C
      DO 310 K=1,KM
        RHOS(IMT,K)=RHOS(2,K)
 310  CONTINUE
C
C***********************************************************************
C  END OF BOOTSTRAP PROCEDURE  *****************************************
C***********************************************************************
C
C***********************************************************************
C  BEGIN ROW-BY-ROW COMPUTATION OF PROGNOSTIC VARIABLES  ***************
C***********************************************************************
C
      DO 380 J=2,JMTM1
C
C-----------------------------------------------------------------------
C  MOVE ALL SLAB DATA DOWN ONE ROW
C-----------------------------------------------------------------------
C
      DO 320 N=1,NSLAB
        TBM(N,1,1)=TB (N,1,1)
        TM (N,1,1)=T  (N,1,1)
        TB (N,1,1)=TBP(N,1,1)
        T  (N,1,1)=TP (N,1,1)
 320  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPLETE READIN OF J+1 SLAB (EXCEPT LAST ROW)
C-----------------------------------------------------------------------
C
      IF(J.NE.JMTM1) THEN
        CALL OGET(LABS(NDISKB),NSLAB,J*NSLAB+1,TBP)
        CALL OGET(LABS(NDISK ),NSLAB,J*NSLAB+1,TP )
      ENDIF
C
C-----------------------------------------------------------------------
C  INITIATE WRITEOUT OF NEWLY COMPUTED DATA FROM PREVIOUS ROW
C-----------------------------------------------------------------------
C
      IF(J.GT.2) CALL OPUT(LABS(NDISKA),NSLAB,(J-2)*NSLAB+1,TA)
C
C-----------------------------------------------------------------------
C  SWITCH SLAB INCIDENTAL DATA INTO CORRECT SLAB AFTER READIN
C-----------------------------------------------------------------------
C
      IF(MOD(ITT,2)+MXP.NE.1) THEN
        DO 332 N=1,NSWICH
          BCON(N,1)=FKMUP(N)
          FKMUP(N)=FKMTP(N)
          FKMTP(N)=BCON(N,1)
 332    CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  SHIFT MAXIMUM LEVEL INDICATORS DOWN ONE ROW AND SET J+1 VALUES
C-----------------------------------------------------------------------
C
      DO 334 I=1,IMT
        KMT (I)=KMTP (I)
        KMU (I)=KMUP (I)
        KMTP(I)=FKMTP(I)
        KMUP(I)=FKMUP(I)
 334  CONTINUE
C
C-----------------------------------------------------------------------
C  SET SYMMETRY BOUNDARY CONDITIONS ON LAST ROW
C-----------------------------------------------------------------------
C
      IF(J.EQ.JMTM1) THEN
        DO 335 I=1,IMT
          KMTP(I)=FKMT (I)
          KMUP(I)=FKMUM(I)
 335    CONTINUE
        DO 336 M=1,NT
        DO 336 K=1,KM
        DO 336 I=1,IMT
          TBP(I,K,M)=TB(I,K,M)
          T P(I,K,M)=T (I,K,M)
 336    CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  MOVE TAU-1 DATA TO TAU LEVEL ON A MIXING TIMESTEP
C-----------------------------------------------------------------------
```

```
C
      IF(MIX.EQ.1) THEN
        DO 337 M=1,NT
        DO 337 K=1,KM
        DO 337 I=1,IMT
          TBP(I,K,M)=TP(I,K,M)
 337    CONTINUE
        DO 338 K=1,KM
        DO 338 I=1,IMT
          UBP(I,K)=UP(I,K)
          VBP(I,K)=VP(I,K)
 338    CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  SHIFT MASKS DOWN ONE ROW AND COMPUTE NEW MASKS
C-----------------------------------------------------------------------
C
      DO 345 K=1,KM
      DO 345 I=1,IMT
        FMM(I,K)=FM (I,K)
        FM (I,K)=FMP(I,K)
 345  CONTINUE
      DO 354 K=1,KM
      DO 354 I=1,IMT
        IF(KMTP(I).GE.KAR(K)) THEN
           FMP(I,K)=1.0
        ELSE
           FMP(I,K)=0.0
        ENDIF
        IF(KMU(I).GE.KAR(K)) THEN
           GM(I,K)=1.0
        ELSE
           GM(I,K)=0.0
        ENDIF
        FXA=0.0
        FXB=1.0
        WHERE (KMTP(1;IMT).GE.KAR(K))
           FMP(1,K;IMT)=FXB
        OTHERWISE
           FMP(1,K;IMT)=FXA
        ENDWHERE
        WHERE (KMU(1;IMT).GE.KAR(K))
           GM(1,K;IMT)=FXB
        OTHERWISE
           GM(1,K;IMT)=FXA
        ENDWHERE
 354  CONTINUE
C
      CALL GETIME(TO,T2)
C
C-----------------------------------------------------------------------
C  CALL THE MAIN COMPUTATIONAL ROUTINES TO UPDATE THE ROW
C-----------------------------------------------------------------------
C
      IF(J.NE.JMTM1) CALL CLINIC(J)
C
      CALL CLINIC(J)
C
      CALL GETIME(T1,T2)
      TIME(3)=TIME(3)+T1-TO
C
      CALL TRACER(J)
C
      CALL GETIME(TO,T2)
      TIME(4)=TIME(4)+TO-T1
C
C-----------------------------------------------------------------------
C  PRINT THE PROGRESSING SOLUTION AT SPECIFIED ROWS ON ENERGY TSTEP
C-----------------------------------------------------------------------
C
      MTEST=MOD(J,??)
      IF(NERGY.EQ.0.OR.MXP.EQ.1) GO TO 339
      IF(MTEST.NE.0) GO TO 8090
      IPRT=??
C
C  DETERMINE INDEX OF FIRST T OCEAN POINT
C
      DO 430 I=1,IMT
        ISTRT=I
        IF(KMT(I).NE.0) GO TO 431
 430  CONTINUE
 431  CONTINUE
      ISTOP=ISTRT+IPRT-1
```

```
      IF(ISTOP.GT.IMT) ISTOP=IMT
      DO 8015 M=1,NT
        IF(M.EQ.1) PRINT 8001,J,ITT
        IF(M.EQ.2) PRINT 8002,J,ITT
        IF(M.EQ.3) PRINT 8003,J,ITT
        IF(M.EQ.4) PRINT 8004,J,ITT
 8001   FORMAT(20H TEMPERATURE FOR J =,I4,12H AT TIMESTEP,I7)
 8002   FORMAT(20H SALINITY    FOR J =,I4,12H AT TIMESTEP,I7)
 8003   FORMAT(20H TRACER 1    FOR J =,I4,12H AT TIMESTEP,I7)
 8004   FORMAT(20H TRACER 2    FOR J =,I4,12H AT TIMESTEP,I7)
        SCL=1.0
        IF(M.EQ.2) SCL=1.E-3
        CALL MATRIX(T(1,1,M),IMT,ISTRT,ISTOP,O,KM,SCL)
 8015 CONTINUE
      PRINT 8011,J,ITT
 8011 FORMAT(20H W VELOCITY  FOR J =,I4,12H AT TIMESTEP,I7)
C
C  SET CYCLIC BOUNDARY CONDITION ON W BEFORE PRINTING
C
      DO 433 K=1,KMP1
        W(1   ,K)=W(IMTM1,K)
        W(IMT,K)=W(2    ,K)
 433  CONTINUE
      SCL=1.E-3
      CALL MATRIX(W,IMT,ISTRT,ISTOP,O,KMP1,SCL)
C
C  DETERMINE INDEX OF FIRST U,V OCEAN POINT
C
      DO 440 I=1,IMTM1
        ISTRT=I
        IF(KMU(I+1).NE.O) GO TO 441
 440  CONTINUE
 441  CONTINUE
      ISTOP=ISTRT+IPRT-1
      IF(ISTOP.GT.IMT) ISTOP=IMT
      PRINT 8021, J,ITT
 8021 FORMAT(20H U VELOCITY  FOR J =,I4,12H AT TIMESTEP,I7)
      SCL=1.0
      CALL MATRIX(U,IMT,ISTRT,ISTOP,O,KM,SCL)
      PRINT 8022, J,ITT
 8022 FORMAT(20H V VELOCITY  FOR J =,I4,12H AT TIMESTEP,I7)
      CALL MATRIX(V,IMT,ISTRT,ISTOP,O,KM,SCL)
C
C-----------------------------------------------------------------------
C  COMPUTE THE NORTHWARD TRANSPORT OF EACH TRACER QUANTITY
C  AS WELL AS THE ZONALLY INTEGRATED MERIDIONAL MASS TRANSPORT
C-----------------------------------------------------------------------
C
 8090 IF(J.EQ.JMTM1) GO TO 8190
      DO 8092 K=1,KM
        VBR(K)=0.0
      DO 8092 M=1,NT
        TBRS(K,M)=TBRN(K,M)
        TBRN(K,M)=0.0
 8092 CONTINUE
      IF(J.GT.2) GO TO 8110
      DO 8094 M=1,NT
      DO 8094 K=1,KM
        TBRS(K,M)=0.0
 8094 CONTINUE
      DO 8102 K=1,KM
        TOTDX=0.0
        DO 8100 I=2,IMTM1
          TOTDX=TOTDX+DXT(I)*(FM(I,K))
        DO 8100 M=1,NT
          TBRS(K,M)=TBRS(K,M)+T(I,K,M)*FM(I,K)*DXT(I)
 8100   CONTINUE
        IF(TOTDX.NE.0.0) THEN
          DO 8101 M=1,NT
            TBRS(K,M)=TBRS(K,M)/TOTDX
 8101     CONTINUE
        ENDIF
 8102 CONTINUE
 8110 CCTJ=AH*DYUR(J)
      DO 8130 K=1,KM
        TOTDX=0.0
        DO 8120 I=2,IMTM1
          TOTDX=TOTDX+DXT(I)*(FMP(I,K))
          VBR(K)=VBR(K)+V(I,K)*DXU(I)*CS(J)
        DO 8120 M=1,NT
          TBRN(K,M)=TBRN(K,M)+TP(I,K,M)*FMP(I,K)*DXT(I)
 8120   CONTINUE
        IF(TOTDX.NE.0.0) THEN
          DO 8122 M=1,NT
```

```
            TBRN(K,M)=TBRN(K,M)/TOTDX
 8122     CONTINUE
        ENDIF
        IF(K.EQ.1) TMT(J,1)=VBR(1)*DZ(1)
        IF(K.GT.1) TMT(J,K)=TMT(J,K-1)+VBR(K)*DZ(K)
      DO 8130 M=1,NT
        TTN(1,J,M)=TTN(1,J,M)+VBR(K)*(TBRN(K,M)+TBRS(K,M))*0.5*DZ(K)
      DO 8130 I=2,IMTM1
        TTN(6,J,M)=TTN(6,J,M)+(V(I,K)*DXU(I)+V(I-1,K)*DXU(I-1))*
     *             (T(I,K,M)+TP(I,K,M))*CS(J)*0.25*DZ(K)
        TTN(7,J,M)=TTN(7,J,M)-CCTJ*FM(I,K)*FMP(I,K)*
     *             (TP(I,K,M)-T(I,K,M))*DXT(I)*CS(J)*DZ(K)
 8130 CONTINUE
      DO 8140 M=1,NT
      DO 8140 I=2,IMTM1
        TOTDZ=0.0
        VBRZ=0.0
        TBRZ=0.0
        IKM=I
        IF(KMU(I-1).GT.KMU(I)) IKM=I-1
        KZ=KMU(IKM)
        IF(KZ.EQ.0) GO TO 8140
        DO 8136 K=1,KZ
          VBRZ=VBRZ+(V(I,K)*DXU(I)+V(I-1,K)*DXU(I-1))*DZ(K)
          TBRZ=TBRZ+(T(I,K,M)+TP(I,K,M))*DZ(K)
          TOTDZ=TOTDZ+DZ(K)
 8136   CONTINUE
        TBRZ=TBRZ/TOTDZ
        TTN(3,J,M)=TTN(3,J,M)+VBRZ*TBRZ*CS(J)*0.25
        TTN(5,J,M)=TTN(5,J,M)-(WSX(I)*DXU(I)+WSX(I-1)*DXU(I-1))*
     *             (T(I,1,M)+TP(I,1,M)-TBRZ)*CS(J)/(8.0*OMEGA*SINE(J))
 8140 CONTINUE
      DO 8150 M=1,NT
        TTN(2,J,M)=TTN(6,J,M)-TTN(1,J,M)
        TTN(4,J,M)=TTN(6,J,M)-TTN(3,J,M)-TTN(5,J,M)
        TTN(8,J,M)=TTN(6,J,M)+TTN(7,J,M)
 8150 CONTINUE
 8190 CONTINUE
 339  CONTINUE
C
C-----------------------------------------------------------------------
C  PUT SLAB INCIDENTAL DATA INTO CORRECT SLAB FOR WRITEOUT
C-----------------------------------------------------------------------
C
      IF(MOD(ITT,2).EQ.0) THEN
        DO 340 N=1,NSWICH
          BCON(N,1)=FKMT(N)
 340    CONTINUE
      ELSE
        DO 360 N=1,NSWICH
          BCON(N,1)=FKMU(N)
 360    CONTINUE
      ENDIF
 380  CONTINUE
C
C=======================================================================
C  END ROW-BY-ROW COMPUTATION  ========================================
C=======================================================================
C
C-----------------------------------------------------------------------
C  PRINT ONE LINE OF TIMESTEP INFORMATION ON SPECIFIED TIMESTEPS
C-----------------------------------------------------------------------
C
      IF(EB.AND.MIX.EQ.1) GO TO 390
      IF(MOD(ITT,NTSI).EQ.0) THEN
        EKTOT=EKTOT/VOLUME
        DO 381 M=1,NT
          DTABS(M)=DTABS(M)/VOLUME
 381    CONTINUE
        DAYSYR=365.25
        TTYEAR=TTSEC/(3600.*24.*DAYSYR)
        TTDAY=TTSEC/(3600.*24.)
        TTDAY=MOD(TTDAY,DAYSYR)
        PRINT 910,ITT,TTYEAR,TTDAY,EKTOT,DTABS(1),DTABS(2),MSCAN
 910    FORMAT(4H TS=,I6,7H  YEAR=,F7.2,6H  DAY=,F5.1,9H  ENERGY=,
     *   1PE13.6,8H  DTEMP=,1PE13.6,8H  DSALT=,1PE13.6,8H  SCANS=,I3)
      ENDIF
C
C-----------------------------------------------------------------------
C  COMPLETE AND PRINT THE ON-LINE INTEGRALS ON ENERGY TIMESTEPS
C-----------------------------------------------------------------------
C
      IF(NERGY.EQ.0) GO TO 390
C
```

```
C  1ST, NORMALIZE PREVIOUSLY COMPUTED INTEGRALS BY VOLUME
C
      DO 382 LL=1,8
        ENGINT(LL)=ENGINT(LL)/VOLUME
        ENGEXT(LL)=ENGEXT(LL)/VOLUME
  382 CONTINUE
      DO 383 M=1,NT
        TVAR(M)=TVAR(M)/VOLUME
      DO 383 LL=1,6
        TTDTOT(LL,M)=TTDTOT(LL,M)/VOLUME
  383 CONTINUE
      BUOY=BUOY/VOLUME
C
C  2ND, COMPUTE RESIDUAL TERMS
C
      PLICIN=ENGINT(1)-ENGINT(2)-ENGINT(3)-ENGINT(4)
     *        -ENGINT(5)-ENGINT(6)-ENGINT(7)-ENGINT(8)
      PLICEX=ENGEXT(1)-ENGEXT(2)-ENGEXT(3)-ENGEXT(4)
     *        -ENGEXT(5)-ENGEXT(6)-ENGEXT(7)-ENGEXT(8)
      DO 384 M=1,NT
        TTDTOT(6,M)=TTDTOT(1,M)-TTDTOT(2,M)-TTDTOT(3,M)
     *              -TTDTOT(4,M)-TTDTOT(5,M)
  384 CONTINUE
C
C  3RD, PRINT THE INTEGRALS
C
      PRINT 9100
      PRINT 9101,ENGINT(1),ENGEXT(1),TTDTOT(1,1),TTDTOT(1,2)
      PRINT 9102,ENGINT(2),ENGEXT(2),TTDTOT(2,1),TTDTOT(2,2)
      PRINT 9103,ENGINT(3),ENGEXT(3),TTDTOT(3,1),TTDTOT(3,2)
      PRINT 9104,ENGINT(4),ENGEXT(4),TTDTOT(4,1),TTDTOT(4,2)
      PRINT 9105,ENGINT(5),ENGEXT(5),TTDTOT(5,1),TTDTOT(5,2)
      PRINT 9106,ENGINT(6),ENGEXT(6),TTDTOT(6,1),TTDTOT(6,2)
      PRINT 9109,PLICIN,PLICEX,TVAR(1),TVAR(2)
      PRINT 9107,ENGINT(7),ENGEXT(7)
      PRINT 9108,ENGINT(8),ENGEXT(8)
 9100 FORMAT( 1X,50HWORK BY:                       INTERNAL MODE  EXTERNAL MODE,
     *       10X,50H                               TEMPERATURE     SALINITY     )
 9101 FORMAT( 1X,20HTIME RATE OF CHANGE ,2(1PE15.6),
     *       10X,20HTIME RATE OF CHANGE ,2(1PE15.6))
 9102 FORMAT( 1X,20HHORIZONTAL ADVECTION,2(1PE15.6),
     *       10X,20HHORIZONTAL ADVECTION,2(1PE15.6))
 9103 FORMAT( 1X,20HVERTICAL ADVECTION  ,2(1PE15.6),
     *       10X,20HVERTICAL ADVECTION  ,2(1PE15.6))
 9104 FORMAT( 1X,20HHORIZONTAL FRICTION ,2(1PE15.6),
     *       10X,20HHORIZONTAL DIFFUSION,2(1PE15.6))
 9105 FORMAT( 1X,20HVERTICAL FRICTION   ,2(1PE15.6),
     *       10X,20HSURFACE FLUX        ,2(1PE15.6))
 9106 FORMAT( 1X,20HPRESSURE FORCES     ,2(1PE15.6),
     *       10X,20HTRUNCATION ERROR    ,2(1PE15.6))
 9107 FORMAT( 1X,20HWORK BY WIND        ,2(1PE15.6))
 9108 FORMAT( 1X,20HBOTTOM DRAG         ,2(1PE15.6))
 9109 FORMAT( 1X,20HIMPLICIT EFFECTS    ,2(1PE15.6),
     *       10X,20HCHANGE OF VARIANCE  ,2(1PE15.6))
      TVAR(1)=BUOY-ENGINT(6)-ENGEXT(6)
      DTABS(1)=ENGINT(2)+ENGINT(3)+ENGEXT(2)+ENGEXT(3)
      PRINT 9110,BUOY,TVAR(1),DTABS(1)
 9110 FORMAT(1X,25HWORK BY BUOYANCY FORCES  ,1PE15.6,5X,25HENERGY CONVER
     *SION ERROR  ,1PE15.6,5X,25HNONLINEAR EXCHANGE ERROR ,1PE15.6)
C
C-----------------------------------------------------------------------
C  PRINT THE NORTHWARD TRANSPORT OF HEAT AND SALT
C-----------------------------------------------------------------------
C
      PRINT 8195
 8195 FORMAT(/,' NORTHWARD TRANSPORT OF HEAT (X10**15 WATTS)',24X,'NORTH
     *WARD TRANSPORT OF SALT (X10**10 CM**3/SEC)',/,6X,'X MEAN  X EDDY
     *Z MEAN  Z EDDY   EKMAN TOT ADV  DIFFUS   TOTAL     X MEAN  X EDDY  Z
     * MEAN  Z EDDY   EKMAN TOT ADV  DIFFUS   TOTAL')
C
C  CONVERT HEAT TRANSPORT TO PEDAWATTS, SALT TRNSPT TO 10**10 CM**3/SEC
C
      DO 8198 J=1,JMT
      DO 8198 LL=1,8
        TTN(LL,J,1)=TTN(LL,J,1)*4.186E-15
        TTN(LL,J,2)=TTN(LL,J,2)*1.E-10
 8198 CONTINUE
      DO 8197 JJ=2,JMTM2
        J=JMT-JJ
        PRINT 8196,J,(TTN(I,J,1),I=1,8),(TTN(I,J,2),I=1,8)
 8196   FORMAT(I4,8F8.3,1X,8F8.3)
 8197 CONTINUE
      PRINT 8194
 8194 FORMAT(/,' MERIDIONAL MASS TRANSPORT')
```

```
      SCL=1.E12
      CALL MATRIX(TMT,JMT,1,JMT,0,KM,SCL)
 390  CONTINUE
C
C-----------------------------------------------------------------------
C  INITIATE WRITEOUT OF NEWLY COMPUTED DATA FROM THE FINAL ROW
C-----------------------------------------------------------------------
C
      CALL OPUT(LABS(NDISKA),NSLAB,(JMT-2)*NSLAB+1,TA)
C
      CALL GETIME(TO,T2)
C
C-----------------------------------------------------------------------
C  SOLVE FOR THE NEW STREAM FUNCTION
C-----------------------------------------------------------------------
C
      CALL RELAX
C
      CALL GETIME(T1,T2)
      TIME(5)=TIME(5)+T1-TO
C
C-----------------------------------------------------------------------
C  IF THIS IS THE END OF THE 1ST PASS OF AN EULER BACKWARD TIMESTEP,
C  SET THE INPUT DISC UNITS SO THAT THE PROPER LEVELS ARE FETCHED ON
C  THE NEXT PASS.  THE OUTPUT FOR THE 2ND PASS WILL BE PLACED ON THE
C  UNUSED UNIT ("NDISKB") AND TRANSFERRED TO THE PROPER UNIT ("NDISKA")
C  LATER.  RETURN TO THE TOP OF "STEP" TO DO THE 2ND PASS.
C-----------------------------------------------------------------------
C
      IF(MIX.EQ.1 .AND. EB) THEN
        MIX=0
        MXP=1
        NDISKX=NDISKB
        NDISKB=NDISK
        NDISK=NDISKA
        NDISKA=NDISKX
        GO TO 182
      ENDIF
C
C-----------------------------------------------------------------------
C  IF THIS IS THE END OF THE 2ND PASS OF AN EULER BACKWARD TIMESTEP,
C  TRANSFER THE DATA WRITTEN TEMPORARILY TO "NDISKX" TO ITS FINAL
C  DESTINATION (THE ORIGINAL "NDISKA").
C-----------------------------------------------------------------------
C
      IF(MXP.EQ.1) THEN
        NDISKA=NDISK
        NDISK=NDISKB
        DO 394 J=2,JMTM1
          CALL OGET(LABS(NDISKX),NSLAB,(J-1)*NSLAB+1,TA)
          CALL OPUT(LABS(NDISKA),NSLAB,(J-1)*NSLAB+1,TA)
 394    CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  FOR PURPOSES OF RECOVERING FROM THE DISC AFTER AN ABNORMAL STOP,
C  NORMALLY INACTIVE DISC UNITS ARE BROUGHT UP TO DATE HERE.
C-----------------------------------------------------------------------
C
      CALL OPUT(KFLDS,NWDS,(NDISKA-1)*NWDS+1,P)
      CALL OPUT(KONTRL,20,1,ITT)
C
C-----------------------------------------------------------------------
C  PRINT THE STREAM FUNCTION ON AN ENERGY TIMESTEP
C-----------------------------------------------------------------------
C
      IF(NERGY.EQ.1) THEN
        PRINT 8000,ITT
 8000   FORMAT(' STREAM FUNCTION IN SVERDRUPS, TS=',I6)
        SCL=1.E12
        CALL MATRIX(P,IMT,2,IMTM1,JMT,0,SCL)
      ENDIF
C
      RETURN
      END
```

```
*DECK CLINIC
      SUBROUTINE CLINIC(J)
C
C*****************************************************************************
C                                                                 ***********
C  CLINIC COMPUTES, FOR ONE ROW, THE INTERNAL MODE COMPONENT OF   ***********
C         THE U AND V VELOCITIES, AS WELL AS THE VORTICITY DRIVING ***********
C         FUNCTION FOR USE BY "RELAX" LATER IN DETERMINING THE     ***********
C         EXTERNAL MODES, WHERE:                                   ***********
C           J=THE ROW NUMBER                                       ***********
C                                                                 ***********
C*****************************************************************************
C
C-----------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C-----------------------------------------------------------------------
C
*CALL PARAM
*CALL FULLWD
*CALL SCALAR
*CALL ONEDIM
*CALL FIELDS
*CALL WRKSPA
C
C-----------------------------------------------------------------------
C  DEFINE AND EQUIVALENCE LOCAL DATA
C-----------------------------------------------------------------------
C
      DIMENSION DPDX(IMT,KM),DPDY(IMT,KM),UENG(IMT,KM),VENG(IMT,KM)
      EQUIVALENCE (TDIF(1,1,1),DPDX(1,1),UENG(1,1)),
     *            (TDIF(1,1,2),DPDY(1,1),VENG(1,1))
C
C-----------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C-----------------------------------------------------------------------
C
C*****************************************************************************
C  BEGIN INTRODUCTORY SECTION, PREPARING VARIOUS     *************************
C  ARRAYS FOR THE COMPUTATION OF THE INTERNAL MODES  *************************
C*****************************************************************************
C
C-----------------------------------------------------------------------
C  FIND ADVECTIVE COEFFICIENT 'FUW' FOR WEST  FACE OF U,V BOX
C                           & 'FVN' FOR NORTH FACE OF U,V BOX
C-----------------------------------------------------------------------
C
C  1ST, CALCULATE EXT. MODE U AT WEST  FACE OF U,V BOX
C                         & V AT NORTH FACE OF U,V BOX
C
      DO 100 I=1,IMT
        SFU(I)=-(P(I  ,J+1)-P(I,J  ))*DYUR(J)*MIN(HR(I-1,J  ),HR(I,J))
        SFV(I)= (P(I+1,J+1)-P(I,J+1))*DXUR(I)*MIN(HR(I  ,J+1),HR(I,J))
     *                                       *CSTR(J+1)
 100  CONTINUE
C
C  2ND, CALCULATE INT. MODE U AT WEST  FACE OF U,V BOX
C                         & V AT NORTH FACE OF U,V BOX
C
      FX=0.5
      DO 110 K=1,KM
      DO 110 I=1,IMT
        FUW(I,K)=(UCLIN(I,K)+UCLIN(I-1,K))*FX
        FVN(I,K)=(VP   (I,K)+VCLIN(I  ,K))*FX
 110  CONTINUE
C
C  3RD, COMBINE EXT. AND INT. MODES AND ADD GRID WGT. FACTOR
C
      FX=DYU2R(J)*CSR(J)*CST(J+1)
      DO 115 K=1,KM
      DO 115 I=1,IMT
        FUW(I,K)=(FUW(I,K)+SFU(I))*CSR(J)
        FVN(I,K)=(FVN(I,K)+SFV(I))*FX
 115  CONTINUE
C
C-----------------------------------------------------------------------
C  SAVE INTERNAL MODE VELOCITIES
C-----------------------------------------------------------------------
C
      DO 140 K=1,KM
      DO 140 I=1,IMT
        USAV(I,K)=UCLIN(I,K)
        VSAV(I,K)=VCLIN(I,K)
        UCLIN(I,K)=UP(I,K)
        VCLIN(I,K)=VP(I,K)
```

```
 140  CONTINUE
C
C  IF LAST ROW, NO NEED TO PERFORM OPERATIONS ON J+1 ROW
C
      IF(J.EQ.JMTM1) GO TO 176
C
C-----------------------------------------------------------------------
C  COMPUTE EXTERNAL MODE VELOCITIES FOR ROW J+1
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE FOR TAU-1 TIME LEVEL
C
      DO 150 I=1,IMTM1
        DIAG1=PB(I+1,J+2)-PB(I  ,J+1)
        DIAG2=PB(I  ,J+2)-PB(I+1,J+1)
        SFUB(I)=-(DIAG1+DIAG2)*DYU2R(J+1)*HR(I,J+1)
        SFVB(I)= (DIAG1-DIAG2)*DXU2R(I  )*HR(I,J+1)*CSR(J+1)
 150  CONTINUE
C
C  2ND, COMPUTE FOR TAU TIME LEVEL
C
      DO 155 I=1,IMTM1
        DIAG1=P (I+1,J+2)-P (I  ,J+1)
        DIAG2=P (I  ,J+2)-P (I+1,J+1)
        SFU (I)=-(DIAG1+DIAG2)*DYU2R(J+1)*HR(I,J+1)
        SFV (I)= (DIAG1-DIAG2)*DXU2R(I  )*HR(I,J+1)*CSR(J+1)
 155  CONTINUE
C
C  3RD, SET CYCLIC BOUNDARY CONDITIONS
C
      SFUB(IMT)=SFUB(2)
      SFVB(IMT)=SFVB(2)
      SFU (IMT)=SFU (2)
      SFV (IMT)=SFV (2)
C
C-----------------------------------------------------------------------
C  ADD EXTERNAL MODE TO INTERNAL MODE FOR ROW J+1  (OCEAN PTS. ONLY)
C-----------------------------------------------------------------------
C
      DO 170 K=1,KM
      DO 170 I=1,IMU
        IF(KMUP(I).GE.KAR(K)) THEN
          UBP(I,K)=UBP(I,K)+SFUB(I)
          VBP(I,K)=VBP(I,K)+SFVB(I)
          UP (I,K)=UP (I,K)+SFU (I)
          VP (I,K)=VP (I,K)+SFV (I)
        ENDIF
        WHERE (KMUP(1;IMU).GE.KAR(K))
          UBP(1,K;IMU)=UBP(1,K;IMU)+SFUB(1;IMU)
          VBP(1,K;IMU)=VBP(1,K;IMU)+SFVB(1;IMU)
          UP (1,K;IMU)=UP (1,K;IMU)+SFU (1;IMU)
          VP (1,K;IMU)=VP (1,K;IMU)+SFV (1;IMU)
        ENDWHERE
 170  CONTINUE
C
C-----------------------------------------------------------------------
C  ACCUMULATE KINETIC ENERGY FROM ROW J+1 EVERY NTSI TIMESTEPS
C-----------------------------------------------------------------------
C
      IF(MOD(ITT,NTSI).EQ.0) THEN
        FX=0.25*CS(J+1)*DYU(J+1)
C
C  WEIGHT SYMMETRY ROW BY ONE HALF
C
        IF(J.EQ.JMTM2) FX=FX*0.5
        DO 173 K=1,KM
        DO 173 I=1,IMT
          UENG(I,K)=(FX*(UP(I,K)*UP(I,K)+VP(I,K)*VP(I,K)))
     *              *C2D2Q(I,K)*DXUQ(I,K)
 173    CONTINUE
        DO 175 K=1,KM
        DO 175 I=2,IMUM1
          EKTOT=EKTOT+UENG(I,K)
C***********************************************************************
C       DO 175 I=2,IMUM1
C         EKTOT=EKTOT+UENG(I,K)
C ***** ABOVE 2 STMNTS REPLACED BY FOLLOWING FOR VECTORIZATION ******
          EKTOT=EKTOT+Q8SSUM(UENG(2,K;IMUM2))
C***********************************************************************
 175    CONTINUE
      ENDIF
 176  CONTINUE
C
C-----------------------------------------------------------------------
```

```
C  SET SYMMETRY CONDITIONS ON THE LAST ROW
C-----------------------------------------------------------------------
C
      IF(J.EQ.JMTM1) THEN
        DO 178 K=1,KM
        DO 178 I=1,IMT
          FVN(I,K)=-FVSU(I,K)
          UBP(I,K)= UBM (I,K)
          UP (I,K)= UM  (I,K)
 178    CONTINUE
C
C  ON 1ST PASS OF MIXING TSTEP, REPLACE TAU-1 U VEL. WITH TAU U VEL.
C
        IF(MIX.NE.0) THEN
          DO 179 K=1,KM
          DO 179 I=1,IMT
            UBP(I,K)=UP(I,K)
 179      CONTINUE
        ENDIF
      ENDIF
C
C-----------------------------------------------------------------------
C  COMPUTE DENSITY OF ROW J+1
C-----------------------------------------------------------------------
C
      CALL STATE(TP(1,1,1),TP(1,1,2),RHON,TDIF(1,1,1),TDIF(1,1,2))
C
C  SET CYCLIC BOUNDARY CONDITIONS
C
      DO 232 K=1,KM
        RHON(IMT,K)=RHON(2,K)
 232  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE VERTICAL VELOCITY IN U,V COLUMNS
C-----------------------------------------------------------------------
C
C 1ST, SET VERTICAL VELOCITY AT THE SURFACE TO ZERO (RIGID-LID)
C
      FX=0.0
      DO 240 I=1,IMT
        W(I,1)=FX
 240  CONTINUE
C
C  2ND, COMPUTE CHANGE OF W BETWEEN LEVELS
C
      DO 250 K=1,KM
      DO 250 I=1,IMT
        W(I,K+1)=C2DZQ(I,K)*((FUW(I+1,K)-FUW (I,K))*DXU2RQ(I,K)
     *                       +FVN(I   ,K)-FVSU(I,K))
 250  CONTINUE
C
C  3RD, INTEGRATE DOWNWARD FROM THE SURFACE
C
      DO 255 K=1,KM
      DO 255 I=1,IMT
        W(I,K+1)=W(I,K)+W(I,K+1)
 255  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE HYDROSTATIC PRESSURE GRADIENT
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE IT AT THE FIRST LEVEL
C
      FXA=GRAV*DZZ(1)*CSR(J)
      FXB=GRAV*DZZ(1)*DYU2R(J)
      DO 260 I=1,IMT
        UDIF(I,1)=RHON(I+1,1)-RHOS(I   ,1)
        VDIF(I,1)=RHON(I   ,1)-RHOS(I+1,1)
        DPDX(I,1)=((UDIF(I,1)-VDIF(I,1))*FXA)*DXU2R(I)
        DPDY(I,1)= (UDIF(I,1)+VDIF(I,1))*FXB
 260  CONTINUE
C
C  2ND, COMPUTE THE CHANGE IN PRESSURE GRADIENT BETWEEN LEVELS
C
      FXA=GRAV*CSR(J)*0.5
      FXB=GRAV*DYU4R(J)
      DO 270 K=2,KM
      DO 270 I=1,IMT
        DPDX(I,K)=RHON(I,K-1)+RHON(I,K)
        DPDY(I,K)=RHOS(I,K-1)+RHOS(I,K)
 270  CONTINUE
      DO 273 K=2,KM
```

```
      DO 273 I=1,IMT
        UDIF(I,K)=DPDX(I+1,K)-DPDY(I   ,K)
        VDIF(I,K)=DPDX(I   ,K)-DPDY(I+1,K)
        DPDX(I,K)=(FXA*(UDIF(I,K)-VDIF(I,K)))*DZZQ(I,K)*DXU2RQ(I,K)
        DPDY(I,K)=(FXB*(UDIF(I,K)+VDIF(I,K)))*DZZQ(I,K)
  273 CONTINUE
C
C  3RD, INTEGRATE DOWNWARD FROM THE FIRST LEVEL
C
      DO 275 K=1,KMM1
      DO 275 I=1,IMT
        DPDX(I,K+1)=DPDX(I,K)+DPDX(I,K+1)
        DPDY(I,K+1)=DPDY(I,K)+DPDY(I,K+1)
  275 CONTINUE
C
C---------------------------------------------------------------------
C  SET BOUNDARY CONDITIONS FOR THE COMPUTATION OF
C       VERTICAL DIFFUSION OF MOMENTUM
C---------------------------------------------------------------------
C
C  1ST, TRANSFER INTERIOR POINTS INTO DIFFUSION COMPUTATION ARRAYS
C
      DO 280 K=1,KM
      DO 280 I=1,IMT
        UDIF(I,K)=UB(I,K)
        VDIF(I,K)=VB(I,K)
  280 CONTINUE
C
C  2ND, SET K=0 ELEMENTS OF DIFF. COMP. ARRAYS TO REFLECT WIND STRESS
C
      FX=DZZ(1)/FKPM
      DO 290 I=1,IMT
        UOVER(I)=UB(I,1)+WSX(I)*FX
        VOVER(I)=VB(I,1)+WSY(I)*FX
  290 CONTINUE
C
C  3RD, SET FIRST LAND LEVEL IN EACH COLUMN TO REFLECT BOTTOM CONDITION
C
      DO 295 I=1,IMT
        KZ=KMU(I)
        UDIF(I,KZ+1)=UB(I,KZ)
        VDIF(I,KZ+1)=VB(I,KZ)
  295 CONTINUE
C
C=========================================================================
C  END INTRODUCTORY SECTION  ==============================================
C=========================================================================
C
C=========================================================================
C  BEGIN COMPUTATION OF THE INTERNAL MODES.          ====================
C  THE NEW VALUES "UA" AND "VA", WILL FIRST BE LOADED WITH  =============
C  THE TIME RATE OF CHANGE, AND THEN UPDATED.        ====================
C=========================================================================
C
C---------------------------------------------------------------------
C  COMPUTE TOTAL ADVECTION OF MOMENTUM
C---------------------------------------------------------------------
C
C  1ST, COMPUTE FLUX THROUGH WEST FACE OF U,V BOX
C
      DO 300 K=1,KM
      DO 300 I=1,IMT
        TEMPA(I,K)=FUW(I,K)*(U(I-1,K)+U(I,K))
        TEMPB(I,K)=FUW(I,K)*(V(I-1,K)+V(I,K))
  300 CONTINUE
C
C  2ND, COMPUTE ZONAL FLUX DIVERGENCE
C
      DO 303 K=1,KM
      DO 303 I=1,IMT
        UA(I,K)=(TEMPA(I,K)-TEMPA(I+1,K))*DXU2RQ(I,K)
        VA(I,K)=(TEMPB(I,K)-TEMPB(I+1,K))*DXU2RQ(I,K)
  303 CONTINUE
C
C  3RD, ADD IN MERIDIONAL FLUX DIVERGENCE
C
      DO 305 K=1,KM
      DO 305 I=1,IMT
        UA(I,K)=UA(I,K)-FVN (I,K)*(UP(I,K)+U (I,K))
     *                 +FVSU(I,K)*(U (I,K)+UM(I,K))
        VA(I,K)=VA(I,K)-FVN (I,K)*(VP(I,K)+V (I,K))
     *                 +FVSU(I,K)*(V (I,K)+VM(I,K))
  305 CONTINUE
C
```

```
C  4TH, COMPUTE FLUX THROUGH TOP OF U,V BOX
C
      DO 340 K=1,KMP1
      DO 340 I=1,IMT
        TEMPA(I,K)=W(I,K)*(U(I,K-1)+U(I,K))
        TEMPB(I,K)=W(I,K)*(V(I,K-1)+V(I,K))
  340 CONTINUE
C
C  5TH, ADD IN VERTICAL FLUX DIVERGENCE
C
      DO 343 K=1,KM
      DO 343 I=1,IMT
        UA(I,K)=UA(I,K)+(TEMPA(I,K+1)-TEMPA(I,K))*DZ2RQ(I,K)
        VA(I,K)=VA(I,K)+(TEMPB(I,K+1)-TEMPB(I,K))*DZ2RQ(I,K)
  343 CONTINUE
C
C-----------------------------------------------------------------------
C  ADD IN HORIZONTAL DIFFUSION OF MOMENTUM (EVAL. AT TAU-1 TSTEP)
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE SEVERAL COEFFICIENTS DEPENDENT ONLY ON LATITUDE
C
      BBUJ=8.0*AM*CSR(J)*CSR(J)
      CCUJ=AM*CST(J+1)*DYTR(J+1)*DYUR(J)*CSR(J)
      DDUJ=AM*CST(J   )*DYTR(J   )*DYUR(J)*CSR(J)
      GGUJ=AM*(1.0-TNG(J)*TNG(J))/(RADIUS*RADIUS)
      HHUJ=2.0*AM*SINE(J)/(RADIUS*CS(J)*CS(J))
C
C  2ND, COMPUTE GRADIENTS AT WEST FACE OF U,V BOX
C
      DO 320 K=1,KM
      DO 320 I=1,IMT
        TEMPA(I,K)=DXT4RQ(I,K)*(UB(I,K)-UB(I-1,K))
        TEMPB(I,K)=DXT4RQ(I,K)*(VB(I,K)-VB(I-1,K))
  320 CONTINUE
C
C  3RD, ADD IN FINAL CONTRIBUTION FROM HOR. DIFF. OF MOMENTUM
C
      DO 323 K=1,KM
      DO 323 I=1,IMT
        UA(I,K)=UA(I,K)+BBUJ*(DXU2RQ(I,K)*(TEMPA(I+1,K)-TEMPA(I,K)))
     *          +CCUJ*(UBP(I,K)-UB(I,K))+DDUJ*(UBM(I,K)-UB(I,K))
     *          +GGUJ*UB(I,K)-HHUJ*DXU2RQ(I,K)*(VB(I+1,K)-VB(I-1,K))
        VA(I,K)=VA(I,K)+BBUJ*(DXU2RQ(I,K)*(TEMPB(I+1,K)-TEMPB(I,K)))
     *          +CCUJ*(VBP(I,K)-VB(I,K))+DDUJ*(VBM(I,K)-VB(I,K))
     *          +GGUJ*VB(I,K)+HHUJ*DXU2RQ(I,K)*(UB(I+1,K)-UB(I-1,K))
  323 CONTINUE
C
C-----------------------------------------------------------------------
C  ADD IN VERTICAL DIFFUSION OF MOMENTUM
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE GRADIENTS AT TOP OF U,V BOX
C
      DO 345 K=1,KMP1
      DO 345 I=1,IMT
        TEMPA(I,K)=UDIF(I,K-1)-UDIF(I,K)
        TEMPB(I,K)=VDIF(I,K-1)-VDIF(I,K)
  345 CONTINUE
C
C  2ND, ADD IN FINAL CONTRIBUTION FROM VERT. DIFF. OF MOMENTUM
C
      DO 348 K=1,KM
      DO 348 I=1,IMT
        UA(I,K)=UA(I,K)+EEMQ(I,K)*TEMPA(I,K)-FFMQ(I,K)*TEMPA(I,K+1)
        VA(I,K)=VA(I,K)+EEMQ(I,K)*TEMPB(I,K)-FFMQ(I,K)*TEMPB(I,K+1)
  348 CONTINUE
C
C-----------------------------------------------------------------------
C  ADD IN CORIOLIS FORCE (EVAL. ON TAU   TSTEP FOR EXPLICIT TRTMNT;
C                         EVAL. ON TAU-1 TSTEP FOR IMPLICIT TRTMNT
C                         WITH REMAINDER OF TERM TO BE ADDED LATER)
C-----------------------------------------------------------------------
C
      FX=2.0*OMEGA*SINE(J)
      IF(ACOR.EQ.0.) THEN
        DO 357 K=1,KM
        DO 357 I=1,IMT
          UA(I,K)=UA(I,K)+FX*V(I,K)
          VA(I,K)=VA(I,K)-FX*U(I,K)
  357   CONTINUE
      ELSE
        DO 359 K=1,KM
        DO 359 I=1,IMT
```

```
          UA(I,K)=UA(I,K)+FX*VB(I,K)
          VA(I,K)=VA(I,K)-FX*UB(I,K)
  359   CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  ADD IN PRESSURE TERM AND MASK OUT LAND
C-----------------------------------------------------------------------
C
      DO 360 K=1,KM
      DO 360 I=1,IMT
        UA(I,K)=GM(I,K)*(UA(I,K)-DPDX(I,K))
        VA(I,K)=GM(I,K)*(VA(I,K)-DPDY(I,K))
  360 CONTINUE
C
C-----------------------------------------------------------------------
C  FORM TIME CHANGE OF VERTICALLY AVERAGED FORCING
C-----------------------------------------------------------------------
C
C  1ST, INTEGRATE TIME CHANGE VERTICALLY
C
      FX=0.0
      DO 370 I=1,IMT
        ZUN(I)=FX
        ZVN(I)=FX
  370 CONTINUE
      DO 380 K=1,KM
        FX=C2DTSF*DZ(K)
      DO 380 I=1,IMT
        ZUN(I)=ZUN(I)+UA(I,K)*FX
        ZVN(I)=ZVN(I)+VA(I,K)*FX
  380 CONTINUE
C
C  (SET SYMMETRY ROW TO ZERO)
C
      IF(J.EQ.JMTM1) ZVN=0.
C
C  2ND, FORM AVERAGE BY DIVIDING BY DEPTH
C
      DO 390 I=1,IMT
        ZUN(I)=ZUN(I)*HR(I,J)
        ZVN(I)=ZVN(I)*HR(I,J)
  390 CONTINUE
C
C-----------------------------------------------------------------------
C  DO ANALYSIS OF INTERNAL MODE FORCING ON ENERGY TIMESTEP
C  ALSO, FORM VERT AVE. FOR USE LATER IN EXT. MODE ANALYSIS
C-----------------------------------------------------------------------
C
      FX=0.0
      DO 395 LL=1,8
      DO 395 I=1,IMT
        ZUNENG(I,LL)=FX
        ZVNENG(I,LL)=FX
  395 CONTINUE
C
C  1ST, COMPUTE KE CHANGE DUE TO PRESSURE TERM
C
      IF(NERGY.EQ.0.OR.MXP.EQ.1) GO TO 550
      FX=CS(J)*DYU(J)
C
C  (WEIGHT SYMMETRY ROW BY ONE HALF)
C
      IF(J.EQ.JMTM1) FX=FX*0.5
      DO 400 K=1,KM
      DO 400 I=2,IMUM1
        UENG(I,K)=GM(I,K)*(-DPDX(I,K))
        VENG(I,K)=GM(I,K)*(-DPDY(I,K))
        ENGINT(6)=ENGINT(6)+(USAV(I,K)*UENG(I,K)
     *                      +VSAV(I,K)*VENG(I,K))*FX*DXU(I)*DZ(K)
        ZUNENG(I,6)=ZUNENG(I,6)+UENG(I,K)*DZ(K)*HR(I,J)
        ZVNENG(I,6)=ZVNENG(I,6)+VENG(I,K)*DZ(K)*HR(I,J)
  400 CONTINUE
C
C  2ND, COMPUTE KE CHANGE DUE TO ADVECTION OF MOMENTUM
C
      DO 430 K=1,KM
      DO 430 I=2,IMUM1
        UENG(I,K)=GM(I,K)*((-FUW (I+1,K)*(U (I+1,K)+U (I  ,K))
     *                      +FUW (I  ,K)*(U (I  ,K)+U (I-1,K)))*DXU2R(I)
     *                      -FVN (I  ,K)*(UP(I  ,K)+U (I  ,K))
     *                      +FVSU(I  ,K)*(U (I  ,K)+UM(I  ,K)))
        VENG(I,K)=GM(I,K)*((-FUW (I+1,K)*(V (I+1,K)+V (I  ,K))
     *                      +FUW (I  ,K)*(V (I  ,K)+V (I-1,K)))*DXU2R(I)
```

```
     *                          -FVM (I   ,K)*(VP(I   ,K)+V (I   ,K))
     *                          +FVSU(I   ,K)*(V (I   ,K)+VM(I   ,K)))
        ENGINT(2)=ENGINT(2)+(USAV(I,K)*UENG(I,K)
     *                          +VSAV(I,K)*VENG(I,K))*FX*DXU(I)*DZ(K)
        ZUNENG(I,2)=ZUNENG(I,2)+UENG(I,K)*DZ(K)*HR(I,J)
        ZVNENG(I,2)=ZVNENG(I,2)+VENG(I,K)*DZ(K)*HR(I,J)
  430 CONTINUE
      DO 460 K=1,KM
      DO 460 I=2,IMUM1
        UENG(I,K)=GM(I,K)*(-(W(I,K   )*(U(I,K-1)+U(I,K   ))
     *                      -W(I,K+1)*(U(I,K   )+U(I,K+1)))*DZ2RQ(I,K))
        VENG(I,K)=GM(I,K)*(-(W(I,K   )*(V(I,K-1)+V(I,K   ))
     *                      -W(I,K+1)*(V(I,K   )+V(I,K+1)))*DZ2RQ(I,K))
        ENGINT(3)=ENGINT(3)+(USAV(I,K)*UENG(I,K)
     *                          +VSAV(I,K)*VENG(I,K))*FX*DXU(I)*DZ(K)
        ZUNENG(I,3)=ZUNENG(I,3)+UENG(I,K)*DZ(K)*HR(I,J)
        ZVNENG(I,3)=ZVNENG(I,3)+VENG(I,K)*DZ(K)*HR(I,J)
  460 CONTINUE
C
C  3RD, COMPUTE KE CHANGE DUE TO HOR. DIFFUSION OF MOMENTUM
C
      DO 490 K=1,KM
      DO 490 I=2,IMUM1
        UENG(I,K)=GM(I,K)*(
     *             +BBUJ*DXU2R(I)*(DXT4R(I+1)*(UB(I+1,K)-UB(I,K))
     *                            +DXT4R(I   )*(UB(I-1,K)-UB(I,K)))
     *             +CCUJ*(UBP(I,K)-UB(I,K))+DDUJ*(UBM(I,K)-UB(I,K))
     *             +GGUJ*UB(I,K)-HHUJ*DXU2R(I)*(VB(I+1,K)-VB(I-1,K)))
        VENG(I,K)=GM(I,K)*(
     *             +BBUJ*DXU2R(I)*(DXT4R(I+1)*(VB(I+1,K)-VB(I,K))
     *                            +DXT4R(I   )*(VB(I-1,K)-VB(I,K)))
     *             +CCUJ*(VBP(I,K)-VB(I,K))+DDUJ*(VBM(I,K)-VB(I,K))
     *             +GGUJ*VB(I,K)+HHUJ*DXU2R(I)*(UB(I+1,K)-UB(I-1,K)))
        ENGINT(4)=ENGINT(4)+(USAV(I,K)*UENG(I,K)
     *                          +VSAV(I,K)*VENG(I,K))*FX*DXU(I)*DZ(K)
        ZUNENG(I,4)=ZUNENG(I,4)+UENG(I,K)*DZ(K)*HR(I,J)
        ZVNENG(I,4)=ZVNENG(I,4)+VENG(I,K)*DZ(K)*HR(I,J)
  490 CONTINUE
C
C  4TH, COMPUTE KE CHANGE DUE TO WIND STRESS
C
      DO 522 I=2,IMUM1
        UENG(I,1)=GM(I,1)*EEM(1)*(UOVER(I)-UDIF(I,1))
        VENG(I,1)=GM(I,1)*EEM(1)*(VOVER(I)-VDIF(I,1))
        ENGINT(7)=ENGINT(7)+(USAV(I,1)*UENG(I,1)
     *                          +VSAV(I,1)*VENG(I,1))*FX*DXU(I)*DZ(1)
        ZUNENG(I,7)=ZUNENG(I,7)+UENG(I,1)*DZ(1)*HR(I,J)
        ZVNENG(I,7)=ZVNENG(I,7)+VENG(I,1)*DZ(1)*HR(I,J)
  522 CONTINUE
C
C  5TH, COMPUTE KE CHANGE DUE TO BOTTOM DRAG
C
      DO 524 I=2,IMUM1
        KZ=KMU(I)
        UENG(I,KZ)=GM(I,KZ)*FFM(KZ)*(UDIF(I,KZ+1)-UDIF(I,KZ))
        VENG(I,KZ)=GM(I,KZ)*FFM(KZ)*(VDIF(I,KZ+1)-VDIF(I,KZ))
        ENGINT(8)=ENGINT(8)+(USAV(I,KZ)*UENG(I,KZ)
     *                          +VSAV(I,KZ)*VENG(I,KZ))*FX*DXU(I)*DZ(KZ)
        ZUNENG(I,8)=ZUNENG(I,8)+UENG(I,KZ)*DZ(KZ)*HR(I,J)
        ZVNENG(I,8)=ZVNENG(I,8)+VENG(I,KZ)*DZ(KZ)*HR(I,J)
  524 CONTINUE
C
C  6TH, COMPUTE KE CHANGE DUE TO VERT. DIFFUSION OF MOMENTUM
C
      DO 520 I=2,IMUM1
        KZ=KMU(I)
      DO 520 K=1,KZ
        FXA=1.0
        FXB=1.0
        IF(K.EQ.1) FXA=0.0
        IF(K.EQ.KZ) FXB=0.0
        UENG(I,K)=GM(I,K)*( FXA*EEM(K)*(UDIF(I,K-1)-UDIF(I,K   ))
     *                     -FXB*FFM(K)*(UDIF(I,K   )-UDIF(I,K+1)))
        VENG(I,K)=GM(I,K)*( FXA*EEM(K)*(VDIF(I,K-1)-VDIF(I,K   ))
     *                     -FXB*FFM(K)*(VDIF(I,K   )-VDIF(I,K+1)))
        ENGINT(5)=ENGINT(5)+(USAV(I,K)*UENG(I,K)
     *                          +VSAV(I,K)*VENG(I,K))*FX*DXU(I)*DZ(K)
        ZUNENG(I,5)=ZUNENG(I,5)+UENG(I,K)*DZ(K)*HR(I,J)
        ZVNENG(I,5)=ZVNENG(I,5)+VENG(I,K)*DZ(K)*HR(I,J)
  520 CONTINUE
  550 CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE NEW VELOCITIES (WITH INCORRECT VERTICAL MEANS)
```

```
C  ALSO, ADD IN REMAINDER OF CORIOLIS TERM IF TREATED IMPLICITLY
C-----------------------------------------------------------------------
C
      IF(ACOR.EQ.0.) THEN
        DO 560 K=1,KM
        DO 560 I=1,IMT
          UA(I,K)=UB(I,K)+C2DTUV*UA(I,K)
          VA(I,K)=VB(I,K)+C2DTUV*VA(I,K)
 560    CONTINUE
      ELSE
        FX=C2DTUV*ACOR*2.0*OMEGA*SINE(J)
        DETMR=1.0/(1.0+FX*FX)
        DO 565 K=1,KM
        DO 565 I=1,IMT
          UDIF(I,K)=(UA(I,K)+FX*VA(I,K))*DETMR
          VDIF(I,K)=(VA(I,K)-FX*UA(I,K))*DETMR
          UA(I,K)=UB(I,K)+C2DTUV*UDIF(I,K)
          VA(I,K)=VB(I,K)+C2DTUV*VDIF(I,K)
 565    CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  DETERMINE THE INCORRECT VERTICAL MEANS OF THE NEW VELOCITIES
C-----------------------------------------------------------------------
C
      FX=0.0
      DO 575 I=1,IMT
        SFU(I)=FX
        SFV(I)=FX
 575  CONTINUE
      DO 580 K=1,KM
      DO 580 I=1,IMT
        SFU(I)=SFU(I)+UA(I,K)*DZ(K)
        SFV(I)=SFV(I)+VA(I,K)*DZ(K)
 580  CONTINUE
      DO 590 I=1,IMT
        SFU(I)=SFU(I)*HR(I,J)
        SFV(I)=SFV(I)*HR(I,J)
 590  CONTINUE
C
C-----------------------------------------------------------------------
C  SUBTRACT INCORRECT VERTICAL MEAN TO GET INTERNAL MODE
C-----------------------------------------------------------------------
C
      DO 600 K=1,KM
      DO 600 I=1,IMT
        UA(I,K)=UA(I,K)-SFU(I)
        VA(I,K)=VA(I,K)-SFV(I)
 600  CONTINUE
      DO 602 K=1,KM
      DO 602 I=1,IMT
        UA(I,K)=GM(I,K)*UA(I,K)
        VA(I,K)=GM(I,K)*VA(I,K)
 602  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE TOTAL CHANGE OF K.E. OF INTERNAL MODE ON ENERGY TIMESTEP
C-----------------------------------------------------------------------
C
      IF(NERGY.EQ.1 .AND. MXP.NE.1) THEN
      DO 605 K=1,KM
        FX=CS(J)*DYU(J)*DZ(K)/C2DTUV
        IF(J.EQ.JMTM1)FX=FX*0.5
        DO 605 I=2,IMUM1
          ENGINT(1)=ENGINT(1)+(USAV(I,K)*(UA(I,K)-UB(I,K))
     *                        +VSAV(I,K)*(VA(I,K)-VB(I,K)))*FX*DXU(I)
 605    CONTINUE
      ENDIF
C
C=========================================================================
C  END COMPUTATION OF INTERNAL MODES  ====================================
C=========================================================================
C
C=========================================================================
C  BEGIN COMPUTATION OF VORTICITY FOR INPUT TO "RELAX"  ==================
C=========================================================================
C
C-----------------------------------------------------------------------
C  SET CYCLIC BOUNDARY CONDITIONS ON EXT. MODE FORCING FUNCTIONS
C-----------------------------------------------------------------------
C
      ZUN(1)=ZUN(IMUM1)
      ZVN(1)=ZVN(IMUM1)
      IF(NERGY.EQ.0.OR.MXP.EQ.1) GO TO 613
```

```
      DO 612 LL=2,8
      ZUNENG(1,LL)=ZUNENG(IMUM1,LL)
      ZVNENG(1,LL)=ZVNENG(IMUM1,LL)
 612  CONTINUE
 613  CONTINUE
C
C-----------------------------------------------------------------------
C  FORM CURL OF TIME CHANGE IN VERTICALLY AVERAGED EQUATIONS
C-----------------------------------------------------------------------
C
C  NON-ACTIVE VORTICITY PTS. ARE KEPT AT ZERO FOR CONVENIENCE IN RELAX.
C  TO ACCOMPLISH THIS, ZTD IS COMPUTED ONLY BETWEEN GIVEN STARTING AND
C  ENDING INDICES.
C
      DO 620 L=1,LSEG
        IS=ISZ(J,L)
        IF(IS.EQ.0) GO TO 621
        IE=IEZ(J,L)
C  ALL VORTICITY PTS. ARE COMPUTED SO THAT THOSE NEEDED FOR THE LINE
C  INTEGRAL OF HOLE RELAXATION (IMMEDIATELY ADJACENT TO ISLANDS) WILL
C  BE DEFINED.
C
      IS=2
      IE=IMTM1
      DO 620 I=IS,IE
        ZTD(I,J)=((ZUN(I)*DXU(I)+ZUN(I-1)*DXU(I-1))*CS(J  )
     *           -(ZUS(I)*DXU(I)+ZUS(I-1)*DXU(I-1))*CS(J-1))
        ZTD(I,J)=(((ZVN(I)-ZVN(I-1))*DYU(J  )
     *           +(ZVS(I)-ZVS(I-1))*DYU(J-1)
     *           -ZTD(I,J))*DXT2R(I)*DYTR(J))*CSTR(J)
 620  CONTINUE
 621  CONTINUE
C
C-----------------------------------------------------------------------
C  DO ANALYSIS OF EXTERNAL MODE FORCING ON ENERGY TIMESTEP
C-----------------------------------------------------------------------
C
      IF(NERGY.EQ.1. AND .MXP.NE.1) THEN
        DO 630 LL=2,8
          IF(J.EQ.JMTM1) THEN
            DO 628 I=1,IMT
              ZVNENG(I,LL)=0.0
 628        CONTINUE
          ENDIF
        DO 630 I=2,IMTM1
          ENGEXT(LL)=ENGEXT(LL)
     *      -P(I,J)*(((ZVNENG(I,LL)-ZVNENG(I-1,LL))*DYU(J)
     *              +(ZVSENG(I,LL)-ZVSENG(I-1,LL))*DYU(J-1))
     *            *DXT2R(I)*DYTR(J)
     *          -((ZUNENG(I,LL)*DXU(I)+ZUNENG(I-1,LL)*DXU(I-1))*CS(J)
     *           -(ZUSENG(I,LL)*DXU(I)+ZUSENG(I-1,LL)*DXU(I-1))*CS(J-1))
     *            *DYT2R(J)*DXTR(I))*DXT(I)*DYT(J)
 630    CONTINUE
        FX=CST(J)*DYT(J)/C2DTSF
        ENGTMP=0.
        DO 635 I=2,IMTM1
          ENGTMP   =ENGTMP    -P(I,J)*ZTD(I,J)*FX*DXT(I)
 635    CONTINUE
        ENGEXT(1)=ENGEXT(1)+ENGTMP
      ENDIF
C
C=======================================================================
C  END COMPUTATION OF VORTICITY  =======================================
C=======================================================================
C
C-----------------------------------------------------------------------
C  FOURIER FILTER U AND V AT HIGH LATITUDES
C-----------------------------------------------------------------------
C
      IF((J.GT.JFU1.AND.J.LT.JFU2).OR.J.LT.JFRST)GO TO 840
      JJ=J-JFRST+1
      IF (J.GE.JFU2) JJ=JJ-JSKPU+1
      FX=-1.0
      IF (PHI(J).GT.0.) FX=1.0
      ISAVE=0
      IEAVE=0
      DO 740 L=1,LSEGF
        DO 730 K=1,KM
          IF(ISUF(JJ,L,K).EQ.0) GO TO 730
          IS=ISUF(JJ,L,K)
          IE=IEUF(JJ,L,K)
          IREDO=1
          IF(IS.NE.ISAVE .OR. IE.NE.IEAVE) THEN
            IREDO=0
```

```
          IM=IE-IS+1
          ISAVE=IS
          IEAVE=IE
          M=2
          N=IFIX(IM*CS(J)*CSR(JFUO)+0.5)
          IF(IM.NE.IMTM2) THEN
            M=2
            N=IFIX(IM*CS(J)*CSR(JFUO)+0.5)
          ELSE
            M=3
            N=IFIX(IM*CS(J)*CSR(JFUO)*.5+.5)
          ENDIF
        ENDIF
        DO 700 II=IS,IE
          I=MOD(II-2,IMUM2)+2
          UDIF(II+1-IS,K)=-FX*UA(I,K)*SPSIN(I)-VA(I,K)*SPCOS(I)
          VDIF(II+1-IS,K)= FX*UA(I,K)*SPCOS(I)-VA(I,K)*SPSIN(I)
  700   CONTINUE
C
        CALL FILTER(UDIF(1,K),IM,M,N,IREDO)
C
        CALL FILTER(VDIF(1,K),IM,M,N,1)
C
        DO 720 II=IS,IE
          I=MOD(II-2,IMUM2)+2
          UA(I,K)=-FX*UDIF(II+1-IS,K)*SPSIN(I)
     *             +FX*VDIF(II+1-IS,K)*SPCOS(I)
          VA(I,K)=-UDIF(II+1-IS,K)*SPCOS(I)-VDIF(II+1-IS,K)*SPSIN(I)
  720   CONTINUE
  730 CONTINUE
  740 CONTINUE
      DO 750 I=1,IMT
        UOVER(I)=0.0
        VOVER(I)=0.0
  750 CONTINUE
      DO 760 K=1,KM
      DO 760 I=1,IMT
        UOVER(I)=UOVER(I)+UA(I,K)*DZ(K)
        VOVER(I)=VOVER(I)+VA(I,K)*DZ(K)
  760 CONTINUE
      DO 770 I=1,IMT
        UOVER(I)=UOVER(I)*HR(I,J)
        VOVER(I)=VOVER(I)*HR(I,J)
  770 CONTINUE
      DO 780 K=1,KM
      DO 780 I=1,IMT
        UA(I,K)=UA(I,K)-UOVER(I)
        VA(I,K)=VA(I,K)-VOVER(I)
  780 CONTINUE
      DO 785 K=1,KM
      DO 785 I=1,IMT
        UA(I,K)=UA(I,K)*GM(I,K)
        VA(I,K)=VA(I,K)*GM(I,K)
  785 CONTINUE
C
C-----------------------------------------------------------------------
C   FOURIER FILTER ZTD AT HIGH LATITUDES
C-----------------------------------------------------------------------
C
      IF(J.EQ.JFU2) GO TO 840
      DO 830 L=1,LSEGF
        IS=ISZF(JJ,L)
        IF(IS.EQ.0) GO TO 840
        IE=IEZF(JJ,L)
        DO 800 II=IS,IE
          I=MOD(II-2,IMTM2)+2
          UDIF(II+1-IS,1)=ZTD(I,J)
  800   CONTINUE
        IM=IE-IS+1
        M=1
        N=IFIX(IM*CST(J)*CSR(JFUO)+.5)
        IF(IM.NE.IMTM2) THEN
          M=1
          N=IFIX(IM*CST(J)*CSR(JFUO)+.5)
        ELSE
          M=3
          N=IFIX(IM*CST(J)*CSR(JFUO)*.5+.5)
        ENDIF
C
        CALL FILTER(UDIF(1,1),IM,M,N,0)
C
        DO 820 II=IS,IE
          I=MOD(II-2,IMTM2)+2
          ZTD(I,J)=UDIF(II+1-IS,1)
```

```
  820     CONTINUE
  830   CONTINUE
  840   CONTINUE
C
C-----------------------------------------------------------------------
C  TRANSFER QUANTITIES COMPUTED TO THE NORTH OF THE PRESENT ROW
C  TO BE DEFINED TO THE SOUTH IN THE COMPUTATION OF THE NEXT ROW
C-----------------------------------------------------------------------
C
      FX=CS(J)*DYU(J)*CSR(J+1)*DYUR(J+1)
      DO 644 K=1,KM
      DO 644 I=1,IMT
        FVSU(I,K)=FVN(I,K)*FX
  644 CONTINUE
      DO 650 I=1,IMT
        ZUS(I)=ZUN(I)
        ZVS(I)=ZVN(I)
  650 CONTINUE
      IF(NERGY.EQ.1. AND .MXP.NE.1) THEN
        DO 660 LL=2,8
        DO 660 I=1,IMT
          ZUSENG(I,LL)=ZUNENG(I,LL)
          ZVSENG(I,LL)=ZVNENG(I,LL)
  660   CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  SET CYCLIC BOUNDARY CONDITIONS ON NEWLY COMPUTED INTERNAL MODE
C-----------------------------------------------------------------------
C
      DO 662 K=1,KM
        UA(1,K)=UA(IMUM1,K)
        VA(1,K)=VA(IMUM1,K)
        UA(IMU,K)=UA(2,K)
        VA(IMU,K)=VA(2,K)
  662 CONTINUE
C
C-----------------------------------------------------------------------
C  SET MERIDIONAL COMPONENT OF INTERNAL MODE TO ZERO ON SYMMETRY ROW
C-----------------------------------------------------------------------
C
      IF(J.EQ.JMTM1) THEN
        FX=0.0
        DO 850 K=1,KM
        DO 850 I=1,IMT
          VA(I,K)=FX
  850   CONTINUE
      ENDIF
C
      RETURN
      END
```

```
*DECK TRACER
      SUBROUTINE TRACER(J)
C
C=========================================================================
C
C  TRACER COMPUTES, FOR ONE ROW, THE NT TRACERS, WHERE:            ===========
C           J=THE ROW NUMBER                                        ===========
C                                                                   ===========
C=========================================================================
C
C-----------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C-----------------------------------------------------------------------
C
*CALL PARAM
*CALL FULLWD
*CALL SCALAR
*CALL ONEDIM
*CALL FIELDS
*CALL WRKSPA
*CALL BITVEC
C
C-----------------------------------------------------------------------
C  DEMENSION LOCAL DATA
C-----------------------------------------------------------------------
C
      REAL TFULL(IMT,KM)
C
C-----------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C-----------------------------------------------------------------------
C
C=========================================================================
C  BEGIN INTRODUCTORY SECTION, PREPARING VARIOUS   ==========================
C  ARRAYS FOR THE COMPUTATION OF THE TRACERS       ==========================
C=========================================================================
C
C-----------------------------------------------------------------------
C  FIND ADVECTIVE COEFFICIENT 'FUW' FOR WEST  FACE OF T BOX
C                           & 'FVN' FOR NORTH FACE OF T BOX
C-----------------------------------------------------------------------
C
      FXA=CSTR(J)*DYTR(J)
      FXB=FXA*CS(J)
      DO 690 K=1,KM
      DO 690 I=1,IMT
        FUW(I,K)=(U(I-1,K)*DYU (J  )+UM(I-1,K)*DYU (J-1  ))*FXA
        FVN(I,K)=(V(I   ,K)*DXUQ(I,K)+V (I-1,K)*DXUQ(I-1,K))*FXB
     *            *DXT4RQ(I,K)
 690  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE VERTICAL VELOCITY IN T COLUMNS
C-----------------------------------------------------------------------
C
C  1ST, SET VERTICAL VELOCITY AT THE SURFACE TO ZERO (RIGID-LID)
C
      FX=0.0
      DO 700 I=1,IMT
        W(I,1)=FX
 700  CONTINUE
C
C  2ND, COMPUTE CHANGE OF W BETWEEN LEVELS
C
      DO 710 K=1,KM
      DO 710 I=1,IMT
        W(I,K+1)=C2DZQ(I,K)*((FUW(I+1,K)-FUW (I,K))*DXT4RQ(I,K)
     *                       +FVN(I   ,K)-FVST(I,K))
 710  CONTINUE
C
C  3RD, INTEGRATE DOWNWARD FROM THE SURFACE
C
      DO 712 K=1,KM
      DO 712 I=1,IMT
        W(I,K+1)=W(I,K)+W(I,K+1)
 712  CONTINUE
C
C-----------------------------------------------------------------------
C  SET BOUNDARY CONDITIONS FOR VERTICAL DIFFUSION OF TRACERS
C-----------------------------------------------------------------------
C
C  1ST, TRANSFER INTERIOR POINTS INTO DIFFUSION COMPUTATION ARRAY
C
      DO 720 M=1,NT
```

```
      DO 720 K=1,KM
      DO 720 I=1,IMT
        TDIF(I,K+1,M)=TB(I,K,M)
 720  CONTINUE
C
C  2ND, SET TOP POINT OF THE COLUMN TO REFLECT SURFACE FLUX,
C           BOTTOM POINT OF THE COLUMN TO REFLECT INSULATION.
C  (THE ROUNDOFF ERROR IN W AT THE BOTTOM IS ALSO ELIMINATED HERE)
C
      FX=0.0
      DO 730 I=1,IMT
        KZ=KMT(I)
        W(I,KZ+1)=FX
      DO 730 M=1,NT
        TDIF(I,1    ,M)=TB(I,1  ,M)
        TDIF(I,KZ+2,M)=TB(I,KZ,M)
 730  CONTINUE
C
C=======================================================================
C  END INTRODUCTORY SECTION  ********************************************
C=======================================================================
C
C=======================================================================
C  BEGIN COMPUTATION OF THE TRACERS.               **********************
C  THE NEW VALUES "TA", WILL FIRST BE LOADED WITH  **********************
C  THE TIME RATE OF CHANGE, AND THEN UPDATED.      **********************
C=======================================================================
C
      DO 855 M=1,NT
C
C-----------------------------------------------------------------------
C  COMPUTE TOTAL ADVECTION OF TRACERS
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE FLUX THROUGH WEST FACE OF T BOX
C
      DO 810 K=1,KM
      DO 810 I=1,IMT
        TEMPA(I,K)=FUW(I,K)*(T(I,K,M)+T(I-1,K,M))
 810  CONTINUE
C
C  2ND, COMPUTE ZONAL FLUX DIVERGENCE
C
      DO 815 K=1,KM
      DO 815 I=1,IMT
        TA(I,K,M)=(TEMPA(I,K)-TEMPA(I+1,K))*DXT4RQ(I,K)
 815  CONTINUE
C
C  3RD, ADD IN MERIDIONAL FLUX DIVERGENCE
C
      DO 820 K=1,KM
      DO 820 I=1,IMT
        TA(I,K,M)=TA(I,K,M)-FVN (I,K)*(TP(I,K,M)+T (I,K,M))
     *                     +FVST(I,K)*(T  (I,K,M)+TM(I,K,M))
 820  CONTINUE
C
C  4TH, COMPUTE FLUX THROUGH TOP OF T BOX
C
      DO 822 K=1,KMP1
      DO 822 I=1,IMT
        TEMPB(I,K)=W(I,K)*(T(I,K-1,M)+T(I,K,M))
 822  CONTINUE
C
C  5TH, ADD IN VERTICAL FLUX DIVERGENCE
C
      DO 824 K=1,KM
      DO 824 I=1,IMT
        TA(I,K,M)=TA(I,K,M)+(TEMPB(I,K+1)-TEMPB(I,K))*DZ2RQ(I,K)
 824  CONTINUE
C
C-----------------------------------------------------------------------
C  ADD IN HORIZONTAL DIFFUSION OF TRACERS (EVAL. AT TAU-1 TIMESTEP)
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE SEVERAL COEFFICIENTS DEPENDENT ONLY ON LATITUDE
C
      BBTJ=8.0*AH*CSTR(J)*CSTR(J)
      CCTJ=AH*CS(J  )*DYUR(J  )*DYTR(J)*CSTR(J)
      DDTJ=AH*CS(J-1)*DYUR(J-1)*DYTR(J)*CSTR(J)
C
C  2ND, COMPUTE GRADIENTS AT WEST FACE OF T BOX
C
      DO 838 K=1,KM
      DO 838 I=1,IMT
```

```
      TEMPA(I,K)=DXU2RQ(I-1,K)*(TB(I,K,M)-TB(I-1,K,M))
  838 CONTINUE
C
C  3RD, ADD IN FINAL CONTRIBUTION FROM HOR. DIFF. OF TRACERS.
C  (TO PROVIDE FOR INSULATED WALLS, EACH GRADIENT IS MULTIPLIED BY
C  THE MASK OF THE POINT IN ITS RESPECTIVE DIRECTION, THUS
C  CAUSING IT TO BE ZERO IF IT IS TAKEN ACROSS A WALL)
C
      DO 840 K=1,KM
      DO 840 I=1,IMT
        TA(I,K,M)=TA(I,K,M)+BBTJ*DXT4RQ(I,K)*
     *            (FM(I+1,K)*TEMPA(I+1,K)-FM(I-1,K)*TEMPA(I,K))
     *            +CCTJ*FMP(I,K)*(TBP(I,K,M)-TB(I,K,M))
     *            +DDTJ*FMM(I,K)*(TBM(I,K,M)-TB(I,K,M))
  840 CONTINUE
C
C-----------------------------------------------------------------------
C  ADD IN VERTICAL DIFFUSION OF TRACERS
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE GRADIENTS AT TOP OF T BOX
C
      DO 842 K=1,KMP1
      DO 842 I=1,IMT
        TEMPB(I,K)=TDIF(I,K,M)-TDIF(I,K+1,M)
  842 CONTINUE
C
C  2ND, ADD IN FINAL CONTRIBUTION FROM VERT. DIFF. OF TRACERS
C
      DO 844 K=1,KM
      DO 844 I=1,IMT
        TA(I,K,M)=TA(I,K,M)+EEHQ(I,K)*TEMPB(I,K  )
     *                     -FFHQ(I,K)*TEMPB(I,K+1)
  844 CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE NEW TRACERS, RESETTING LAND POINTS TO ZERO
C-----------------------------------------------------------------------
C
      DO 850 K=1,KM
      DO 850 I=1,IMT
        TA(I,K,M)=(TB(I,K,M)+C2DTTS*TA(I,K,M))*FM(I,K)
C
C  FORCE UPDATING IN FULL PRECISION TO ASSURE CONSERVATION OF TRACERS
C
      TFULL(I,K)=TB(I,K,M)+EXTEND(C2DTTS)*TA(I,K,M)
  850 CONTINUE
C
C  ROUND TO HALF PRECISION (NOTE.. DOES NOT TRUNCATE)
C
      CALL Q8RCONV(B'00000000',0,TFULL(1,1;IMTKM),,,0,TA(1,1,M;IMTKM))
      TA(*,*,M)=TA(*,*,M)*FM(*,*)
C
  855 CONTINUE
C
C-----------------------------------------------------------------------
C  SET SALINITY TO 45 PPT OVER LAND TO STOP CONVECTION THERE
C  (..NOTE THAT THIS IS .01 IN MODEL UNITS -- (PPT-35)/1000..)
C-----------------------------------------------------------------------
C
      IF(NT.GE.2) THEN
        FXA=0.01
        FXB=1.0
        DO 860 K=1,KM
        DO 860 I=1,IMT
          TA(I,K,2)=TA(I,K,2)+FXA*(FXB-FM(I,K))
  860   CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  DO ANALYSIS OF TRACER FORCING ON ENERGY TIMESTEP
C-----------------------------------------------------------------------
C
      IF(NERGY.EQ.0.OR.MXP.EQ.1) GO TO 920
      DO 910 I=2,IMTM1
        KZ=KMT(I)
        IF (KZ.EQ.0) GOTO 910
        DO 900 M=1,NT
        DO 900 K=1,KZ
          BOXVOL = CST(J)*DXT(I)*DYT(J)*DZ(K)
C
C  1ST, COMPUTE TRACER CHANGE DUE TO ADVECTION
C
          TTDTOT(2,M)=TTDTOT(2,M)+BOXVOL*
```

```
     *              ((-FUW (I+1,K)*(T (I+1,K,M)+T (I   ,K,M))
     *                +FUW (I   ,K)*(T (I   ,K,M)+T (I-1,K,M)))*DXT4R(I)
     *                -FVN (I   ,K)*(TP(I   ,K,M)+T (I   ,K,M))
     *                +FVST(I   ,K)*(T (I   ,K,M)+TM(I   ,K,M)))
          TTDTOT(3,M)=TTDTOT(3,M)+BOXVOL*
     *                (W(I,K+1)*(T(I,K   ,M)+T(I,K+1,M))
     *                -W(I,K   )*(T(I,K-1,M)+T(I,K   ,M)))*DZ2R(K)
C
C  2ND, COMPUTE TRACER CHANGE DUE TO HORIZONTAL DIFFUSION
C
          TTDTOT(4,M)=TTDTOT(4,M)+BOXVOL*
     *     (BBTJ*DXU2R(I   )*DXT4R(I)*FM(I+1,K)*(TB(I+1,K,M)-TB(I,K,M))
     *      +BBTJ*DXU2R(I-1)*DXT4R(I)*FM(I-1,K)*(TB(I-1,K,M)-TB(I,K,M))
     *              +CCTJ*FMP(I,K)*(TBP(I,K,M)-TB(I,K,M))
     *              +DDTJ*FMM(I,K)*(TBM(I,K,M)-TB(I,K,M)))
C
C  3RD, COMPUTE TRACER CHANGE DUE TO VERTICAL DIFFUSION
C
          TTDTOT(5,M)=TTDTOT(5,M)+BOXVOL*
     *                (EEH(K)*(TDIF(I,K,M)-TDIF(I,K+1,M))
     *                -FFH(K)*(TDIF(I,K+1,M)-TDIF(I,K+2,M)))
 900    CONTINUE
C
C  4TH, COMPUTE TOTAL ENERGY EXCHANGE BETWEEN POTENTIAL AND KINETIC
C
        IF(KZ.LT.2) GO TO 910
        FX=CST(J)*DXT(I)*DYT(J)*GRAV*0.5
        DO 905 K=2,KZ
          BUOY=BUOY-FX*DZZ(K)*W(I,K)*(RHOS(I,K-1)+RHOS(I,K))
 905    CONTINUE
 910  CONTINUE
 920  CONTINUE
C
C-----------------------------------------------------------------------
C  CONVECTIVELY ADJUST WATER COLUMN IF GRAVITATIONALLY UNSTABLE
C-----------------------------------------------------------------------
C
C  SET NCON FOR NUMBER OF PASSES THROUGH CONVECTION LOOP
C  KS=1: COMPARE LEV. 1 TO 2; 3 TO 4; ETC. AND ADJUST IF NECESSARY
C  KS=2: COMPARE LEV. 2 TO 3; 4 TO 5; ETC. AND ADJUST IF NECESSARY
C
      NCON=1
      DO 965 N=1,NCON
      DO 965 KS=1,2
C
C  1ST, FIND DENSITY FOR ENTIRE SLAB FOR STABILITY DETERMINATION
C
        CALL STATEC(TA,TA(1,1,2),TEMPA,TDIF,TDIF(1,1,2),KS)
C
C  2ND, FOR EACH TRACER, MIX ADJOINING LEVELS IF UNSTABLE
C
        DO 960 K=KS,KMM1,2
        DO 960 I=2,IMTM1
          IF(TEMPA(I,K).GT.TEMPA(I,K+1)) THEN
            DO 948 M=1,NT
              TA(I,K,M)=(DZ(K)*TA(I,K,M)+DZ(K+1)*TA(I,K+1,M))*DZZ2R(K+1)
              TA(I,K+1,M)=TA(I,K,M)
 948        CONTINUE
          ENDIF
 960    CONTINUE
        FX=0.5
        LN=(2*((KMM1-KS)/2)+1)*IMT
        DO 960 M=1,NT
          TEMPB(1,1;IMTKM)=FX*C2DZQ(1,1;IMTKM)*TA(1,1,M;IMTKM)
          WHERE (KALTBV(1,KS,KS;LN)
     *         .AND. TEMPA(1,KS;LN).GT.TEMPA(1,KS+1;LN))
            TA(1,KS,M;LN)=(TEMPB(1,KS;LN)+TEMPB(1,KS+1;LN))
     *                  *DZZ2RQ(1,KS+1;LN)
            TA(1,KS+1,M;LN)=TA(1,KS,M;LN)
          ENDWHERE
 960    CONTINUE
 965  CONTINUE
C
C=======================================================================
C  END COMPUTATION OF THE TRACERS  =====================================
C=======================================================================
C
C-----------------------------------------------------------------------
C  INTEGRATE TOTAL CHANGES IN T,S AND SQUARED T,S ON ENERGY TIMESTEP
C-----------------------------------------------------------------------
C
      IF(NERGY.EQ.1 .AND. MXP.NE.1) THEN
       DO 970 M=1,NT
       DO 970 K=1,KM
```

```
        FX=CST(J)*DYT(J)*DZ(K)/C2DTTS
        DO 970 I=2,IMTM1
          TTDTOT(1,M)=TTDTOT(1,M)+(TA(I,K,M)    -TB(I,K,M)    )*FX*DXT(I)
          TVAR(M)     =TVAR(M)     +(TA(I,K,M)**2-TB(I,K,M)**2)*FX*DXT(I)
 970    CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  FOURIER FILTER TRACERS AT HIGH LATITUDES
C-----------------------------------------------------------------------
C
      IF((J.GT.JFT1.AND.J.LT.JFT2).OR.J.LT.JFRST)GO TO 1190
      JJ=J-JFRST+1
      IF (J.GE.JFT2) JJ=JJ-JSKPT+1
C
C  IF PREVIOUS STRIPS WERE OF SAME LENGTH, DONT RECOMPUTE FOURIER COEFFS
C
      ISAVE=0
      IEAVE=0
      DO 1140 L=1,LSEGF
        DO 1135 K=1,KM
          IF(ISTF(JJ,L,K).EQ.0) GO TO 1135
          IS=ISTF(JJ,L,K)
          IE=IETF(JJ,L,K)
          IREDO=0
          IF(IS.NE.ISAVE .OR. IE.NE.IEAVE) THEN
            IREDO=-1
            ISAVE=IS
            IEAVE=IE
            IM=IE-IS+1
            M=1
            N=IFIX(IM*CST(J)*CSTR(JFTO)+.5)
            IF(IM.NE.IMTM2.OR.KMT(1).LT.K) THEN
              M=1
              N=IFIX(IM*CST(J)*CSTR(JFTO)+.5)
            ELSE
              M=3
              N=IFIX(IM*CST(J)*CSTR(JFTO)*.5+.5)
            ENDIF
          ENDIF
          DO 1130 MM=1,NT
            IDX=IREDO+MM
            DO 1100 II=IS,IE
              I=MOD(II-2,IMTM2)+2
              TDIF(II+1-IS,K,1)=TA(I,K,MM)
 1100       CONTINUE
C
            CALL FILTER(TDIF(1,K,1),IM,M,N,IDX)
C
            DO 1120 II=IS,IE
              I=MOD(II-2,IMTM2)+2
              TA(I,K,MM)=TDIF(II+1-IS,K,1)
 1120       CONTINUE
 1130     CONTINUE
 1135   CONTINUE
 1140 CONTINUE
 1190 CONTINUE
C
C-----------------------------------------------------------------------
C  ACCUMULATE INTEGRATED ABSOLUTE CHANGES IN T EVERY NTSI TIMESTEPS
C-----------------------------------------------------------------------
C
      IF(MOD(ITT,NTSI).EQ.0) THEN
        FX=0.5*CST(J)*DYT(J)/C2DTTS
        DO 983 M=1,NT
        DO 983 K=1,KM
        DO 983 I=1,IMT
          TDIF(I,K,M)=ABS(TA(I,K,M)-TB(I,K,M))*C2DZQ(I,K)*FX*DXTQ(I,K)
 983    CONTINUE
        DO 985 M=1,NT
        DO 985 K=1,KM
        DO 985 I=2,IMTM1
          DTABS(M)=DTABS(M)+TDIF(I,K,M)
C  **********************************************************************
C       DO 985 I=2,IMTM1
C         DTABS(M)=DTABS(M)+TDIF(I,K,M)
C  ****  ABOVE 2 STMNTS REPLACED BY FOLLOWING FOR VECTORIZATION  *****
          DTABS(M)=DTABS(M)+Q8SSUM(TDIF(2,K,M;IMTM2))
C  **********************************************************************
 985    CONTINUE
      ENDIF
C
C-----------------------------------------------------------------------
C  TRANSFER QUANTITIES COMPUTED TO THE NORTH OF THE PRESENT ROW
```

```
C  TO BE DEFINED TO THE SOUTH IN THE COMPUTATION OF THE NEXT ROW
C-----------------------------------------------------------------------
C
      FX=CST(J)*DYT(J)*CSTR(J+1)*DYTR(J+1)
      DO 990 K=1,KM
      DO 990 I=1,IMT
        FVST(I,K)=FVN(I,K)*FX
        RHOS(I,K)=RHON(I,K)
 990  CONTINUE
C
C-----------------------------------------------------------------------
C  SET CYCLIC BOUNDARY CONDITIONS ON NEWLY COMPUTED TRACERS
C-----------------------------------------------------------------------
C
      DO 992 M=1,NT
      DO 992 K=1,KM
        TA(1   ,K,M)=TA(IMTM1,K,M)
        TA(IMT,K,M)=TA(2     ,K,M)
 992  CONTINUE
C
C-----------------------------------------------------------------------
C  SET NEW VELOCITIES AT NORTHERN WALL TO ZERO SINCE NO PASS THROUGH
C  CLINIC IS MADE FOR THIS ROW
C-----------------------------------------------------------------------
C
      IF(J.EQ.JMTM1) THEN
        FX=0.0
        DO 680 K=1,KM
        DO 680 I=1,IMT
          UA(I,K)=FX
          VA(I,K)=FX
 680    CONTINUE
      ENDIF
      RETURN
      END
```

```
*DECK RELAX
      SUBROUTINE RELAX
C
C*****************************************************************************
C                                                                ***********
C  RELAX TAKES AS INPUT THE VORTICITY DRIVING FUNCTION COMPUTED IN ***********
C        "CLINIC" (ZTD) AND, USING SEQUENTIAL OVER-RELAXATION,     ***********
C        SOLVES THE LAPLACIAN EQUATION FOR THE EXTERNAL MODE OF    ***********
C        VELOCITY IN TERMS OF A MASS TRANSPORT STREAM FUNCTION (P). ***********
C                                                                ***********
C*****************************************************************************
C
C-----------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C-----------------------------------------------------------------------
C
*CALL PARAM
*CALL FULLWD
*CALL SCALAR
*CALL ONEDIM
*CALL FIELDS
*CALL WRKSPB
*CALL TIME
C
C-----------------------------------------------------------------------
C  DEMENSION AND EQUIVALENCE LOCAL DATA
C-----------------------------------------------------------------------
C
      DIMENSION PTDB(IMT,JMT),H(IMT,JMT)
      EQUIVALENCE (RES,PTDB),(PTD,H)
      BIT INTBV(IMT,JMT)
C
C-----------------------------------------------------------------------
C  DO SOME PRELIMINARY SETUP FOR HAND-TUNED SCALAR LOOP
C-----------------------------------------------------------------------
C
      INTEGER CFWLOC,RESLOC
      PARAMETER (IEND5L=16,ISTORE=17,CFWLOC=18,RESLOC=19)
      DESCRIPTOR CFWADR,RESADR
C
C-----------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C-----------------------------------------------------------------------
C
C=============================================================================
C  BEGIN INTRODUCTORY SECTION TO PREPARE FOR THE RELAXATION   ==================
C=============================================================================
C
C-----------------------------------------------------------------------
C  INITIATE READIN OF RELAXATION SOLUTION OF 2 TIMESTEPS BACK
C  FOR THE PURPOSE OF COMPUTING AN INITIAL GUESS FOR THE PRESENT PASS
C  (INPUT UNIT ALTERNATES ON TSTEPS BETWEEN 5 & 6)
C-----------------------------------------------------------------------
C
      LUPTDB=6-MOD(ITT,2)
      LUPTD =5+MOD(ITT,2)
      CALL OFIND(KFLDS,NWDS,(LUPTDB-1)*NWDS+1)
C
C-----------------------------------------------------------------------
C  INITIALIZE THE WORK AREA
C-----------------------------------------------------------------------
C
      FX=0.0
      DO 90 J=1,JMT
      DO 90 I=1,IMT
        CFN(I,J)=FX
        CFS(I,J)=FX
        CFE(I,J)=FX
        CFW(I,J)=FX
        COF(I,J)=FX
        ISMASK(I,J)=0
 90   CONTINUE
C
C  INITIALIZE BIT INTERIOR INCICATOR TO ZEROES
C
      INTBV(1,1;NWDS)=B'0'
C
C-----------------------------------------------------------------------
C  COMPLETE READIN OF RELAXATION SOLUTION OF 2 TIMESTEPS BACK AND
C  INITIATE READIN OF RELAXATION SOLUTION OF PREVIOUS TIMESTEP
C  (INPUT UNIT ALTERNATES ON TIMESTEPS BETWEEN 6 & 5)
C-----------------------------------------------------------------------
C
      CALL OGET(KFLDS,NWDS,(LUPTDB-1)*NWDS+1,PTDB)
```

```
      CALL OFIND(KFLDS,NWDS,(LUPTD-1)*NWDS+1)
C
C-----------------------------------------------------------------------
C  FORM ISLAND MASK  (ISMASK=0 OVER INTERIOR OCEAN POINTS
C                     ISMASK=1 OVER PERIMETER OCEAN POINTS
C                     ISMASK=2 OVER LAND POINTS)
C-----------------------------------------------------------------------
C
      DO  95 I=2,IMU
      DO  95 J=2,JMTM1
        TEST1=HR(I,J)*HR(I-1,J)*HR(I,J-1)*HR(I-1,J-1)
        TEST2=HR(I,J)+HR(I-1,J)+HR(I,J-1)+HR(I-1,J-1)
        IF(TEST1.EQ.0.0) ISMASK(I,J)=1
        IF(TEST2.EQ.0.0) ISMASK(I,J)=2
  95  CONTINUE
C
C-----------------------------------------------------------------------
C  CALCULATE DEPTH FIELD FROM ITS RECIPROCAL
C  (..NOTE.. H IS EQUIVALENCED WITH PTD)
C-----------------------------------------------------------------------
C
C  COMPUTE RECIPROCALS WITH AN EPSILON ADDED TO AVOID ZERO DIVIDE
C
      FXA=1.0
      FXB=1.E-20
      DO 110 J=2,JMTM1
      DO 110 I=1,IMT
        H(I,J)=FXA/(HR(I,J)+FXB)
 110  CONTINUE
C
C  THEN RESET H OVER LAND TO ZERO
C
      FX=0.0
      DO 112 J=1,JMT
      DO 112 I=1,IMT
        IF(HR(I,J).EQ.FX)H(I,J)=FX
 112  CONTINUE
C
C  THEN SET CYCLIC BOUNDARY CONDITIONS ON H
C
      DO 117 J=1,JMT
        H(1,J)=H(IMUM1,J)
 117  CONTINUE
C
C-----------------------------------------------------------------------
C  GENERATE ARRAYS OF COEFFICIENTS FOR RELAXATION
C-----------------------------------------------------------------------
C
C  1ST, COMPUTE COEFFICIENTS OF THE LAPLACIAN STAR
C  (HOLD NON-INTERIOR POINTS TO ZERO USING START AND END INDICES)
C
      DO 130 J=3,JMTM1
      DO 130 L=1,LSEG
        IS=ISZ(J,L)
        IF(IS.EQ.0) GO TO 130
        IE=IEZ(J,L)
        FXA=2.0*CSTR(J)*CSTR(J)
        FXB=2.0*CS(J  )*CSTR(J)*DYTR(J)*DYUR(J  )
        FXC=2.0*CS(J-1)*CSTR(J)*DYTR(J)*DYUR(J-1)
C
C  (SET BIT INTERIOR INDICATOR TO ONES OVER INTERIOR POINTS)
C
        INTBV(IS,J;IE-IS+1)=.NOT.INTBV(IS,J;IE-IS+1)
        FX=1.0
        DO 120 I=IS,IE
          CFN(I,J)=FXB/(H(I-1,J)+H(I,J))
          CFS(I,J)=FXC/(H(I-1,J-1)+H(I,J-1))
          CFE(I,J)=FXA*DXUR(I)*DXTR(I)/(H(I,J)+H(I,J-1))
          CFW(I,J)=FXA*DXUR(I-1)*DXTR(I)/(H(I-1,J)+H(I-1,J-1))
          CPF(I)=FX/(CFN(I,J)+CFS(I,J)+CFE(I,J)+CFW(I,J))
 120    CONTINUE
C
C  2ND, AUGMENT COEFFICIENTS FOR IMPLICIT TREATMENT OF CORIOLIS TERM
C
        IF(ACOR.NE.0.0) THEN
          FX=-C2DTSF*ACOR*CSTR(J)*DYTR(J)*OMEGA
          DO 125 I=IS,IE
            CFN(I,J)=CFN(I,J)+(HR(I,J  )-HR(I-1,J  ))*SINE(J  )
     *                       *FX*DXTR(I)
            CFS(I,J)=CFS(I,J)-(HR(I,J-1)-HR(I-1,J-1))*SINE(J-1)
     *                       *FX*DXTR(I)
            CFE(I,J)=CFE(I,J)-(HR(I  ,J)*SINE(J)-HR(I  ,J-1)*SINE(J-1))
     *                       *FX*DXTR(I)
            CFW(I,J)=CFW(I,J)+(HR(I-1,J)*SINE(J)-HR(I-1,J-1)*SINE(J-1))
```

```
     *                   *FX*DXTR(I)
 125      CONTINUE
        ENDIF
C
C  3RD, NORMALIZE COEFFICIENTS AND FORCING TERM FOR EFFICIENCY
C
        DO 128 I=IS,IE
          CFN(I,J)=CFN(I,J)*CPF(I)
          CFS(I,J)=CFS(I,J)*CPF(I)
          CFE(I,J)=CFE(I,J)*CPF(I)
          CFW(I,J)=CFW(I,J)*CPF(I)
          ZTD(I,J)=ZTD(I,J)*CPF(I)
 128    CONTINUE
 130  CONTINUE
C
C  4TH, COMPUTE COEFFICIENTS ON ISLAND PERIMETER POINTS
C
      DO 180 ISLE=1,NISLE
        COFIS(ISLE)=0.0
        IS=ISIS(ISLE)
        IE=IEIS(ISLE)
        JS=JSIS(ISLE)
        JE=JEIS(ISLE)
        DO 175 J=JS,JE
          FXA=2.0*CSTR(J)*CSTR(J)
          FXB=2.0*CS(J  )*DYUR(J  )*DYTR(J)*CSTR(J)
          FXC=2.0*CS(J-1)*DYUR(J-1)*DYTR(J)*CSTR(J)
          FXD=-C2DTSF*ACOR*CSTR(J)*DYTR(J)*OMEGA
        DO 175 I=IS,IE
          IF(ISMASK(I,J).NE.1) GO TO 175
          IF(HR(I-1,J  ).NE.0.0 .OR. HR(I  ,J  ).NE.0.0)
     *      CFN(I,J)=FXB/(PTD(I-1,J)+PTD(I,J))
     *           +FXD*DXTR(I)*(HR(I,J)-HR(I-1,J))*SINE(J)
          IF(HR(I-1,J-1).NE.0.0 .OR. HR(I  ,J-1).NE.0.0)
     *      CFS(I,J)=FXC/(PTD(I-1,J-1)+PTD(I,J-1))
     *           -FXD*DXTR(I)*(HR(I,J-1)-HR(I-1,J-1))*SINE(J-1)
          IF(HR(I  ,J  ).NE.0.0 .OR. HR(I  ,J-1).NE.0.0)
     *      CFE(I,J)=FXA*DXTR(I)*DXUR(I)/(PTD(I,J)+PTD(I,J-1))
     *           -FXD*DXTR(I)*(HR(I,J)*SINE(J)-HR(I,J-1)*SINE(J-1))
          IF(HR(I-1,J  ).NE.0.0 .OR. HR(I-1,J-1).NE.0.0)
     *      CFW(I,J)=FXA*DXTR(I)*DXUR(I-1)/(PTD(I-1,J)+PTD(I-1,J-1))
     *           +FXD*DXTR(I)*(HR(I-1,J)*SINE(J)-HR(I-1,J-1)*SINE(J-1))
          COF(I,J)=1.0/(CFN(I,J)+CFS(I,J)+CFE(I,J)+CFW(I,J))
          CFN(I,J)=CFN(I,J)*COF(I,J)
          CFS(I,J)=CFS(I,J)*COF(I,J)
          CFE(I,J)=CFE(I,J)*COF(I,J)
          CFW(I,J)=CFW(I,J)*COF(I,J)
          ZTD(I,J)=ZTD(I,J)*COF(I,J)
          COF(I,J)=CST(J)*DXT(I)*DYT(J)/COF(I,J)
          COFIS(ISLE)=COFIS(ISLE)+COF(I,J)
 175    CONTINUE
        COFIS(ISLE)=1./COFIS(ISLE)
 180  CONTINUE
C
C  FINALLY, MULTIPLY THE COEFFICIENTS BY THE OVERRELAXATION FACTOR
C
      DO 182 J=1,JMT
      DO 182 I=1,IMT
        CFN(I,J)=CFN(I,J)*SOR
        CFS(I,J)=CFS(I,J)*SOR
        CFE(I,J)=CFE(I,J)*SOR
        CFW(I,J)=CFW(I,J)*SOR
 182  CONTINUE
C
C-----------------------------------------------------------------------
C  COMPUTE A FIRST GUESS FOR THE RELAXATION BY EXTRAPOLATING THE TWO
C  PREVIOUS SOLUTIONS FORWARD IN TIME.
C-----------------------------------------------------------------------
C
C  1ST, COMPLETE READIN OF RELAXATION SOLUTION OF PREVIOUS TIMESTEP
C
      CALL OGET(KFLDS,NWDS,(LUPTD-1)*NWDS+1,PTD)
C
C  2ND, PERFORM TIME EXTRAPOLATION (ACCOUNTING FOR MIXING TIMESTEP)
C
      FXA=1.0
      FXB=2.0
      IF(MIX.EQ.1.OR.MXP.EQ.1) FXA=0.5
      DO 135 J=1,JMT
      DO 135 I=1,IMT
        PTD(I,J)=FXA*(FXB*PTD(I,J)-PTDB(I,J))
 135  CONTINUE
C
C  COMPUTE CRITERION FOR CONVERGENCE OF RELAXATION AND SET RESIDUALS O
```

```
C                                                                        C
      CRTP=CRIT*FXA*SOR
      FX=0.0
      DO 140 J=1,JMT
      DO 140 I=1,IMT
        RES(I,J)=FX
  140 CONTINUE
      CALL GETIME(TO,T2)
C                                                                        T
C*************************************************************************C
C  END INTRODUCTORY SECTION   ********************************************C
C*************************************************************************C
C                                                                        C
C*************************************************************************C
C  BEGIN SECTION TO DO THE RELAXATION   **********************************C
C*************************************************************************C
C                                                                        C
      MSCAN=0
  300 MSCAN=MSCAN+1
C
C-----------------------------------------------------------------------  C
C  COMPUTE ENTIRE FIELD OF RESIDUALS AS IN SIMULTANEOUS RELAXATION        C
C  (..NOTE.. RESIDUALS WILL REMAIN 0 OVER NON-INTERIOR POINTS)            C
C-----------------------------------------------------------------------  -IC
C                                                                         C
      DO 230 J=3,JSCAN                                                    C
      DO 230 I=1,IMT
      RES(I,J)=CFN(I,J)*PTD(I,J+1)
     *        +CFS(I,J)*PTD(I,J-1)
     *        +CFE(I,J)*PTD(I+1,J)
     *        +CFW(I,J)*PTD(I-1,J)
     *        -SOR*(PTD(I,J)+ZTD(I,J))
  230 CONTINUE
C
C-----------------------------------------------------------------------  IC
C  RESET RESIDUALS TO ZERO OVER LAND                                      IC
C-----------------------------------------------------------------------  IC
C                                                                         IC
      FX=0.0                                                              IC
      DO 224 J=3,JSCAN                                                    I
      DO 220 I=1,IMT                                                      I-W
        CPF(I)=FX                                                         I-W
  220 CONTINUE                                                            I-W
      DO 221 L=1,LSEG                                                     I
        IS=ISZ(J,L)                                                       I
        IF(IS.EQ.0) GO TO 222                                             I-W
        IE=IEZ(J,L)                                                       I-W
      DO 221 I=IS,IE                                                      I-W
        CPF(I)=RES(I,J)                                                   I-W
  221 CONTINUE                                                            I-W
  222 CONTINUE                                                            I-W
      DO 223 I=1,IMT                                                      I-W
        RES(I,J)=CPF(I)                                                   I-W
  223 CONTINUE                                                            I-W
  224 CONTINUE                                                            I-W
      LENV=(JSCAN-2)*IMT                                                  I-W
      WHERE (.NOT.INTBV(1,3;LENV)) RES(1,3;LENV)=FX                       IW
C                                                                         IW
C-----------------------------------------------------------------------  OC
C  SET CYCLIC BOUNDARY CONDITIONS ON THE RESIDUALS                        OC
C-----------------------------------------------------------------------  OC
C                                                                         OC
      DO 226 J=3,JSCAN                                                    OC
        RES(1,J)=RES(IMUM1,J)                                             O
        RES(IMT,J)=RES(2,J)                                               O
  226 CONTINUE                                                            O
C                                                                         O
C-----------------------------------------------------------------------  C
C  PERFORM CORRECTION ON SOUTHERN POINT TO YIELD SEQUENTIAL RELAXATION    C
C-----------------------------------------------------------------------  C
C                                                                         C
      DO 247 J=3,JSCAN                                                    C
        DO 236 I=2,IMTM1
          RES(I,J)=RES(I,J)+CFS(I,J)*RES(I,J-1)                           -I
  236   CONTINUE                                                          -I
        DO 254 L=1,LSEG                                                   -I
          IS=ISZ(J,L)                                                     I-W
          IF(IS.EQ.0) GO TO 255                                           I-W
          IE=IEZ(J,L)                                                     I-W
        DO 254 I=IS,IE                                                    I-W
          RES(I,J)=RES(I,J)+CFS(I,J)*RES(I,J-1)                           I-W
  254   CONTINUE                                                          I
  255   CONTINUE                                                          I
        WHERE (INTBV(1,J;IMT))                                            I-W
                                                                          IW
```

```
          RES(1,J;IMT)=RES(1,J;IMT)+CFS(1,J;IMT)*RES(1,J-1;IMT)
        ENDWHERE
C
C-----------------------------------------------------------------------
C  PERFORM CORRECTION ON WESTERN POINT TO YIELD SEQUENTIAL RELAXATION
C-----------------------------------------------------------------------
C
        DO 245 L=1,LSEG
          IS=IS2(J,L)
          IF(IS.EQ.0) GO TO 247
          IE=IE2(J,L)
          DO 237 I=IS,IE
            RES(I,J)=RES(I,J)+CFW(I,J)*RES(I-1,J)
 237      CONTINUE
C***********************************************************************
C  THE FOLLOWING IS A HAND-TUNED VERSION OF THE RECURSIVE SCALAR LOOP:
C         DO 237 I=IS,IE
C           RES(I,J)=RES(I,J)+CFW(I,J)*RES(I-1,J)
C237      CONTINUE
C***********************************************************************
          IEND=IE+4
          I=0
          ASSIGN RESADR,RES(I,J;0)
          ASSIGN CFWADR,CFW(I,J;0)
          CALL Q8ADDX(IS,X'16',I)
          CALL Q8SUBX(IS,X'16',ISTORE)
          CALL Q8RTOR(IE,,IEND5L)
          CALL Q8RTOR(CFWADR,,CFWLOC)
          CALL Q8RTOR(RESADR,,RESLOC)
          CALL Q8LOD(RESLOC,ISTORE,15)
          CALL Q8LOD(CFWLOC,IS,6)
          CALL Q8LOD(RESLOC,IS,11)
          CALL Q8LOD(CFWLOC,I,7)
          CALL Q8LOD(RESLOC,I,12)
          CALL Q8LODH(RESLOC,ISTORE,15)
          CALL Q8LODH(CFWLOC,IS,6)
          CALL Q8LODH(RESLOC,IS,11)
          CALL Q8LODH(CFWLOC,I,7)
          CALL Q8LODH(RESLOC,I,12)
          CALL Q8IBXGE(B'00000100',I,X'16',&2237,IEND5L,I)
C
1237      CALL Q8MPYS(15,6,6)
          CALL Q8LOD(CFWLOC,I,8)
          CALL Q8STO(RESLOC,ISTORE,15)
          CALL Q8ADDN(11,6,11)
          CALL Q8LOD(RESLOC,I,13)
1237      CALL Q8MPYSH(15,6,6)
          CALL Q8LODH(CFWLOC,I,8)
          CALL Q8STOH(RESLOC,ISTORE,15)
          CALL Q8ADDNH(11,6,11)
          CALL Q8LODH(RESLOC,I,13)
          CALL Q8ADDX(I,X'16',I)
          CALL Q8ADDX(ISTORE,X'16',ISTORE)
C
          CALL Q8MPYS(11,7,7)
          CALL Q8LOD(CFWLOC,I,9)
          CALL Q8STO(RESLOC,ISTORE,11)
          CALL Q8ADDN(12,7,12)
          CALL Q8LOD(RESLOC,I,14)
          CALL Q8MPYSH(11,7,7)
          CALL Q8LODH(CFWLOC,I,9)
          CALL Q8STOH(RESLOC,ISTORE,11)
          CALL Q8ADDNH(12,7,12)
          CALL Q8LODH(RESLOC,I,14)
          CALL Q8ADDX(I,X'16',I)
          CALL Q8ADDX(ISTORE,X'16',ISTORE)
C
          CALL Q8MPYS(12,8,8)
          CALL Q8LOD(CFWLOC,I,10)
          CALL Q8STO(RESLOC,ISTORE,12)
          CALL Q8ADDN(13,8,13)
          CALL Q8LOD(RESLOC,I,15)
          CALL Q8MPYSH(12,8,8)
          CALL Q8LODH(CFWLOC,I,10)
          CALL Q8STOH(RESLOC,ISTORE,12)
          CALL Q8ADDNH(13,8,13)
          CALL Q8LODH(RESLOC,I,15)
          CALL Q8ADDX(I,X'16',I)
          CALL Q8ADDX(ISTORE,X'16',ISTORE)
C
          CALL Q8MPYS(13,9,9)
          CALL Q8LOD(CFWLOC,I,6)
          CALL Q8STO(RESLOC,ISTORE,13)
          CALL Q8ADDN(14,9,14)
```

```
            CALL Q8LOD(RESLOC,I,11)
            CALL Q8MPYSH(13,9,9)
            CALL Q8LODH(CFWLOC,I,6)
            CALL Q8STOH(RESLOC,ISTORE,13)
            CALL Q8ADDNH(14,9,14)
            CALL Q8LODH(RESLOC,I,11)
            CALL Q8ADDX(I,X'16',I)
            CALL Q8ADDX(ISTORE,X'16',ISTORE)
C
            CALL Q8MPYS(14,10,10)
            CALL Q8LOD(CFWLOC,I,7)
            CALL Q8STO(RESLOC,ISTORE,14)
            CALL Q8ADDN(15,10,15)
            CALL Q8LOD(RESLOC,I,12)
            CALL Q8MPYSH(14,10,10)
            CALL Q8LODH(CFWLOC,I,7)
            CALL Q8STOH(RESLOC,ISTORE,14)
            CALL Q8ADDNH(15,10,15)
            CALL Q8LODH(RESLOC,I,12)
            CALL Q8ADDX(ISTORE,X'16',ISTORE)
            CALL Q8IBXLT(B'00000110',I,X'16',&1237,IEND5L,I)
            CALL Q8IBXGE(B'00000100',I,,&237,IEND,I)
C
 2237       CALL Q8MPYS(15,6,6)
            CALL Q8LOD(CFWLOC,I,8)
            CALL Q8STO(RESLOC,ISTORE,15)
            CALL Q8ADDN(11,6,11)
            CALL Q8LOD(RESLOC,I,13)
 2237       CALL Q8MPYSH(15,6,6)
            CALL Q8LODH(CFWLOC,I,8)
            CALL Q8STOH(RESLOC,ISTORE,15)
            CALL Q8ADDNH(11,6,11)
            CALL Q8LODH(RESLOC,I,13)
            CALL Q8ADDX(ISTORE,X'16',ISTORE)
            CALL Q8IBXGE(B'00000100',I,X'16',&237,IEND,I)
C
            CALL Q8MPYS(11,7,7)
            CALL Q8STO(RESLOC,ISTORE,11)
            CALL Q8ADDN(12,7,12)
            CALL Q8MPYSH(11,7,7)
            CALL Q8STOH(RESLOC,ISTORE,11)
            CALL Q8ADDNH(12,7,12)
            CALL Q8ADDX(ISTORE,X'16',ISTORE)
            CALL Q8IBXGE(B'00000100',I,X'16',&237,IEND,I)
C
            CALL Q8MPYS(12,8,8)
            CALL Q8STO(RESLOC,ISTORE,12)
            CALL Q8ADDN(13,8,13)
            CALL Q8MPYSH(12,8,8)
            CALL Q8STOH(RESLOC,ISTORE,12)
            CALL Q8ADDNH(13,8,13)
            CALL Q8ADDX(ISTORE,X'16',ISTORE)
            CALL Q8IBXGE(B'00000100',I,X'16',&237,IEND,I)
C
            CALL Q8STO(RESLOC,ISTORE,13)
            CALL Q8STOH(RESLOC,ISTORE,13)
C
  237       CONTINUE
C
C*******************************************************************
C  END UNROLLED RECURSIVE SCALAR LOOP
C*******************************************************************
  245   CONTINUE
  247 CONTINUE
C
C-------------------------------------------------------------------
C  MAKE A CORRECTION TO PTD BASED ON THE RESIDUALS
C-------------------------------------------------------------------
C
      DO 235 J=3,JSCAN
      DO 235 I=1,IMT
        PTD(I,J)=PTD(I,J)+RES(I,J)
  235 CONTINUE
C
C-------------------------------------------------------------------
C  FIND THE MAXIMUM ABSOLUTE RESIDUAL TO DETERMINE CONVERGENCE
C-------------------------------------------------------------------
C
      RESMAX=0.0
      DO 253 J=3,JSCAN
      DO 253 I=2,IMTM1
        RESMAX=MAX(ABS(RES(I,J)),RESMAX)
  253 CONTINUE
      RESMAX=MAX(ABS(Q8SMAX(RES(1,1;NWDS))),
```

```
     *          ABS(Q8SMIN(RES(1,1;NWDS))))
C
C-----------------------------------------------------------------------
C  DO HOLE RELAXATION FOR EACH ISLAND
C-----------------------------------------------------------------------
C
      FX=0.0
      DO 260 ISLE=1,NISLE
        IS=ISIS(ISLE)
        IE=IEIS(ISLE)
        JS=JSIS(ISLE)
        JE=JEIS(ISLE)
        RESIS=FX
        DO 262 J=JS,JE
        DO 262 I=IS,IE
          IF(ISMASK(I,J).EQ.1) THEN
            RESIS=RESIS+(CFN(I,J)*PTD(I  ,J+1)
     *                  +CFS(I,J)*PTD(I  ,J-1)
     *                  +CFE(I,J)*PTD(I+1,J  )
     *                  +CFW(I,J)*PTD(I-1,J  )
     *                  -SOR*(PTD(I,J)+ZTD(I,J)))*COF(I,J)
          ENDIF
262     CONTINUE
C
C  COMPUTE A RESIDUAL AT ALL PERIMETER POINTS IN THE LATITUDINAL
C  BAND COVERED BY THE ISLAND.
C
        LENV2=IE-IS+1
        LENV1=(JE-JS)*IMT+LENV2
        RES(IS,JS;LENV1)=0.0
        WHERE(ISMASK(IS,JS;LENV1).EQ.1)
          RES(IS,JS;LENV1)=
     *          (CFN(IS,JS;LENV1)*PTD(IS,JS+1;LENV1)
     *          +CFS(IS,JS;LENV1)*PTD(IS,JS-1;LENV1)
     *          +CFE(IS,JS;LENV1)*PTD(IS+1,JS;LENV1)
     *          +CFW(IS,JS;LENV1)*PTD(IS-1,JS;LENV1)
     *     -SOR*(PTD(IS,JS;LENV1)+ZTD(IS,JS;LENV1) ))*COF(IS,JS;LENV1)
        ENDWHERE
C
C  DO THE LINE INTEGRAL ONLY AROUND THE ISLAND IN QUESTION
C
        RESIS=0.0
        DO 240 J=JS,JE
        DO 240 I=IS,IE
          RESIS=RESIS+RES(I,J)
C  *********************************************************************
C       DO 240 I=IS,IE
C         RESIS=RESIS+RES(I,J)
C  ***** ABOVE 2 STATEMENTS REPLACED BY FOLLOWING FOR VECTORIZATION ***
          RESIS=RESIS+Q8SSUM(RES(IS,J;LENV2))
C  *********************************************************************
 240    CONTINUE
C
C  NORMALIZE THE ISLAND RESIDUAL AND UPDATE THE MAXIMUM
C  ABSOLUTE RESIDUAL OF THE RELAXATION IF NECESSARY
C
        RESIS=RESIS*COFIS(ISLE)
        RESMAX=MAX(ABS(RESIS),RESMAX)
C
C  MAKE A CORRECTION TO PTD OVER THE ISLAND AND ITS PERIMETER POINTS
C
        DO 250 J=JS,JE
        DO 250 I=IS,IE
          IF(ISMASK(I,J).GE.1) THEN
            PTD(I,J)=PTD(I,J)+RESIS
          ENDIF
          WHERE(ISMASK(IS,J;LENV2).GE.1)
            PTD(IS,J;LENV2)=PTD(IS,J;LENV2)+RESIS
          ENDWHERE
 250    CONTINUE
 260  CONTINUE
C
C-----------------------------------------------------------------------
C  SET SYMMETRY BOUNDARY CONDITION
C-----------------------------------------------------------------------
C
      DO 270 I=1,IMT
        PTD(I,JMT)=-PTD(I,JMTM1)
 270  CONTINUE
C
C-----------------------------------------------------------------------
C  SET CYCLIC BOUNDARY CONDITION
C-----------------------------------------------------------------------
C
```

```
      DO 272 J=1,JMT
        PTD(1,J)=PTD(IMUM1,J)
        PTD(IMU,J)=PTD(2,J)
 272  CONTINUE
C
C-----------------------------------------------------------------------
C  TEST MAXIMUM RESIDUAL FOR CONVERGENCE OF THE RELAXATION.
C  IF NOT CONVERGED, PROCEED WITH ANOTHER SCAN.
C  (..NOTE.. IF THE NUMBER OF SCANS REACHES MXSCAN, LEAVE THE LOOP)
C-----------------------------------------------------------------------
C
      IF(RESMAX.GE.CRTP .AND. MSCAN.LT.MXSCAN) GO TO 300
C
C=======================================================================
C  END OF THE RELAXATION  ==============================================
C=======================================================================
C
      CALL GETIME(T1,T2)
      TIME(8)=TIME(8)+T1-TO
C
C-----------------------------------------------------------------------
C  UPDATE THE STREAM FUNCTION BASED UPON THE RELAXATION SOLUTION
C-----------------------------------------------------------------------
C
      DO 340 J=1,JMT
      DO 340 I=1,IMT
        PTDB(I,J)=PB(I,J)+PTD(I,J)
        PB(I,J)=P(I,J)
        P(I,J)=PTDB(I,J)
 340  CONTINUE
C
C-----------------------------------------------------------------------
C  SAVE PTD TO COMPUTE 1ST GUESS FOR RELAXATION NEXT TIMESTEP
C  (..NOTE.. ON 1ST PASS OF EULER BACKWARD TIMESTEP, BYPASS THIS
C            SAVE, SINCE IT WILL BE DONE ON THE 2ND PASS)
C  (..NOTE.. ON A MIXING TIMESTEP, ALTER PTD TO BE CONSISTENT WITH
C            NORMAL, LEAP-FROG STEPPING)
C-----------------------------------------------------------------------
C
      IF(MIX.EQ.1 .AND. EB) RETURN
      IF(MXP.NE.O. OR. MIX.NE.O) THEN
        FX=2.O
        DO 350 J=1,JMT
        DO 350 I=1,IMT
          PTD(I,J)=FX*PTD(I,J)
 350    CONTINUE
      ENDIF
 360  CALL OPUT(KFLDS,NWDS,(LUPTDB-1)*NWDS+1,PTD)
      RETURN
      END
```

```
*DECK STATE
      SUBROUTINE STATE(TX,SX,RHO,TQ,SQ)
C
C=============================================================================
C                                                                ===========
C  STATE COMPUTES ONE ROW OF NORMALIZED DENSITIES BY USING A 3RD  ===========
C        ORDER POLYNOMIAL FIT TO THE KNUDSEN FORMULA, LEVEL BY    ===========
C        LEVEL, WHERE:                                            ===========
C          TX =THE INPUT ROW OF TEMPERATURES                      ===========
C          SX =THE INPUT ROW OF SALINITIES (UNITS: (PPT-35)/1000) ===========
C          RHO=THE RETURNED ROW OF NORMALIZED DENSITIES           ===========
C          TQ =ONE ROW OF WORK SPACE PROVIDED BY THE CALLING ROUTINE ===========
C          SQ =ONE ROW OF WORK SPACE PROVIDED BY THE CALLING ROUTINE ===========
C                                                                ===========
C=============================================================================
C
C---------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C---------------------------------------------------------------------
C
*CALL PARAM
*CALL SCALAR
*CALL WRKSPA
*CALL TIME
C
C---------------------------------------------------------------------
C  DEMENSION LOCAL DATA
C---------------------------------------------------------------------
C
      DIMENSION TX(IMT,KM),SX(IMT,KM),RHO(IMT,KM),TQ(IMT,KM),SQ(IMT,KM)
      DIMENSION TO(KM),SO(KM),C(KM,9)
C
C---------------------------------------------------------------------
C  ENTER NORMALIZING TEMPERATURES AND SALINITIES,
C  AND COEFFICIENTS GENERATED BY THE PROGRAM ("KNUDSN") WHICH
C  FITS 3RD ORDER POLYNOMIALS TO THE KNUDSEN FORMULA, LEVEL BY LEVEL.
C---------------------------------------------------------------------
C
      DATA TO/??/
      DATA SO/??/
      DATA  C/??/
C
C---------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C---------------------------------------------------------------------
      CALL GETIME(TO,T2)
C
C---------------------------------------------------------------------
C  SUBTRACT NORMALIZING CONSTANTS FROM TEMPERATURE AND SALINITY
C  AND COMPUTE POLYNOMIAL APPROXIMATION OF KNUDSEN DENSITY.
C  (..NOTE.. FOR PRECISION PURPOSES, THERE IS A CONSTANT SUBTRACTED
C   FROM THE DENSITY RETURNED BY THIS ROUTINE. THIS MAKES NO DIFFERENCE
C   HOWEVER, SINCE ONLY HORIZONTAL GRADIENTS ARE USED BY THE MODEL.)
C---------------------------------------------------------------------
C
      DO 100 K=1,KM
      DO 100 I=1,IMT
        TQ(I,K)=TX(I,K)-TOQ(I,K)
        SQ(I,K)=SX(I,K)-SOQ(I,K)
        RHO(I,K)=(CQ(I,K,1)+(CQ(I,K,4)+CQ(I,K,7)*SQ(I,K))*SQ(I,K)
     *  +(CQ(I,K,3)+CQ(I,K,8)*SQ(I,K)+CQ(I,K,6)*TQ(I,K))*TQ(I,K))
     *  *TQ(I,K)+(CQ(I,K,2)+(CQ(I,K,5)+CQ(I,K,9)
     *  *SQ(I,K))*SQ(I,K))*SQ(I,K)
 100  CONTINUE
      CALL GETIME(T1,T2)
      TIME(6)=TIME(6)+T1-TO
      RETURN
C
      ENTRY STATEC(TX,SX,RHO,TQ,SQ,IND)
C
C=============================================================================
C                                                                ===========
C  STATEC COMPUTES, FOR ONE ROW, THE NORMALIZED DENSITIES BY USING ===========
C         A 3RD ORDER POLYNOMIAL FIT TO THE KNUDSEN FORMULA, FOR   ===========
C         PURPOSES OF CHECKING VERTICAL STABILITY BETWEEN ADJACENT ===========
C         LEVELS.  THE REFERENCE DEPTH FOR PRESSURE DEPENDENCE IN  ===========
C         THE KNUDSEN FORMULA MUST BE HELD CONSTANT FOR THIS PURPOSE.===========
C         THAT LEVEL IS DETERMINED BY "IND".  THE ARGUMENTS ARE:   ===========
C       TX =THE INPUT ROW OF TEMPERATURES                         ===========
C       SX =THE INPUT ROW OF SALINITIES (UNITS: (PPT-35)/1000)    ===========
C       RHO=THE RETURNED ROW OF NORMALIZED DENSITIES              ===========
C       TQ =ONE ROW OF WORK SPACE PROVIDED BY THE CALLING ROUTINE ===========
C       SQ =ONE ROW OF WORK SPACE PROVIDED BY THE CALLING ROUTINE ===========
C       IND=1 FOR COMPARING LEVELS 1 TO 2, 3 TO 4, ETC.           ===========
```

```
C          --COEFFICIENTS FOR THE LOWER OF THE 2 LEVELS ARE USED      ===========
C       IND=2 FOR COMPARING LEVELS 2 TO 3, 4 TO 5, ETC.                ===========
C          --COEFFICIENTS FOR THE UPPER OF THE 2 LEVELS ARE USED      ===========
C                                                                       ===========
C========================================================================================
C
      CALL GETIME(TO,T2)
      DO 1100 L=1,KM
      DO 1100 I=1,IMT
      TQ(I,L)=TX(I,L)-TOIQ(I,L,IND)
      SQ(I,L)=SX(I,L)-SOIQ(I,L,IND)
      RHO(I,L)=(CIQ(I,L,1,IND)+(CIQ(I,L,4,IND)+CIQ(I,L,7,IND)
     1*SQ(I,L))*SQ(I,L)+(CIQ(I,L,3,IND)+CIQ(I,L,8,IND)*SQ(I,L)
     2+CIQ(I,L,6,IND)*TQ(I,L))*TQ(I,L))*TQ(I,L)+(CIQ(I,L,2,IND)
     3+(CIQ(I,L,5,IND)+CIQ(I,L,9,IND)*SQ(I,L))*SQ(I,L))*SQ(I,L)
 1100 CONTINUE
      CALL GETIME(T1,T2)
      TIME(7)=TIME(7)+T1-TO
      RETURN
C
      ENTRY STINIT
C
C========================================================================================
C
C  STINIT LOADS THE APPROPRIATE NORMALIZATION CONSTANTS AND COEF-     ===========
C         FICIENTS INTO ARRAYS OF PROPER DIMENSION TO PERMIT VEC-      ===========
C         TORIZATION IN THE SUBSEQUENT CALLS TO "STATE" AND "STATEC"  ===========
C                                                                       ===========
C========================================================================================
C
C------------------------------------------------------------------------
C  LOAD COEFFICIENTS FOR USE IN STATE
C------------------------------------------------------------------------
C
      DO 10 N=1,9
      DO 10 K=1,KM
      DO 10 I=1,IMT
        CQ(I,K,N)=C(K,N)
 10   CONTINUE
      DO 20 K=1,KM
      DO 20 I=1,IMT
        TOQ(I,K)=TO(K)
        SOQ(I,K)=SO(K)
 20   CONTINUE
C
C------------------------------------------------------------------------
C  LOAD COEFFICIENTS FOR USE IN STATEC.
C  DETERMINE THE REFERENCE LEVEL INDICATOR, "KREF" IN ACCORD WITH
C  COMMENT ON "IND" IN INTRODUCTORY STATEMENT FOR ENTRY STATEC.
C------------------------------------------------------------------------
C
      DO 70 IND=1,2
        DO 52 K=1,KM,2
          IF (IND.EQ.1) THEN
            KREF=K+1
            IF(KREF.GT.KM) KREF=KM
          ELSE
            KREF=K
          ENDIF
          DO 50 I=1,IMT
            TOIQ(I,K,IND)=TOQ(I,KREF)
            SOIQ(I,K,IND)=SOQ(I,KREF)
 50       CONTINUE
          DO 52 N=1,9
          DO 52 I=1,IMT
            CIQ(I,K,N,IND)=CQ(I,KREF,N)
 52     CONTINUE
        DO 62 K=2,KM,2
          IF (IND.EQ.2) THEN
            KREF=K+1
            IF(KREF.GT.KM) KREF=KM
          ELSE
            KREF=K
          ENDIF
          DO 60 I=1,IMT
            TOIQ(I,K,IND)=TOQ(I,KREF)
            SOIQ(I,K,IND)=SOQ(I,KREF)
 60       CONTINUE
          DO 62 N=1,9
          DO 62 I=1,IMT
            CIQ(I,K,N,IND)=CQ(I,KREF,N)
 62     CONTINUE
 70   CONTINUE
      RETURN
```

END

616800000

```
*DECK MATRIX
      SUBROUTINE MATRIX(ARRAY,IRDIM,ISTRT,IM,JM,KK,SCALE)
C
C***********************************************************************
C                                                            ***********
C  MATRIX IS A GENERAL TWO-DIMENSIONAL ARRAY PRINTING ROUTINE,   ***********
C      WHERE:                                                ***********
C      ARRAY=THE ARRAY TO BE PRINTED                         ***********
C      IRDIM=THE 1ST DIMENSION OF ARRAY                      ***********
C      ISTRT=THE 1ST ELEMENT OF THE 1ST DIMENSION TO BE PRINTED  ***********
C      IM   =THE LAST ELEMENT OF THE 1ST DIMENSION TO BE PRINTED ***********
C      JM   =THE LAST ELEMENT OF THE 2ND DIMENSION TO BE PRINTED ***********
C            --THE ROWS ARE PRINTED IN DESCENDING ORDER--    ***********
C            (IF THIS IS 0, KK IS USED)                      ***********
C      KK   =THE LAST ELEMENT OF THE 2ND DIMENSION TO BE PRINTED ***********
C            --THE ROWS ARE PRINTED IN ASCENDING ORDER--     ***********
C            (IF THIS IS 0, JM IS USED)                      ***********
C      SCALE=A SCALING FACTOR BY WHICH ARRAY IS DIVIDED BEFORE   ***********
C            PRINTING.  (IF THIS IS ZERO, NO SCALING IS DONE.)   ***********
C            IF SCALE=0, 10 COLUMNS ARE PRINTED ACROSS IN E FORMAT ***********
C            IF SCALE>0, 20 COLUMNS ARE PRINTED ACROSS IN F FORMAT ***********
C***********************************************************************
C
C-----------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C-----------------------------------------------------------------------
C
*CALL PARAM
C
C-----------------------------------------------------------------------
C  DEFINE LOCAL DATA
C-----------------------------------------------------------------------
C
      DIMENSION ARRAY(IRDIM,1),NUM(20),PLINE(20)
C
C-----------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C-----------------------------------------------------------------------
C
      IF(SCALE.NE.0) GO TO 500
      DO 251 IS=ISTRT,IM,10
      IE=IS+9
      IF(IE.GT.IM) IE=IM
      IDIF=IE-IS+1
      DO 100 I=1,IDIF
  100 NUM(I)=IS+I-1
      PRINT 9990, (NUM(I),I=1,IDIF)
 9990 FORMAT(10I13,/)
      JMORKM=JM+KK
      DO 252 JORK=1,JMORKM
      IF(JM.NE.0) L=JMORKM+1-JORK
      IF(KK.NE.0) L=JORK
      PRINT 9966, L, (ARRAY(I,L),I=IS,IE)
  252 CONTINUE
      PRINT 9984
  251 CONTINUE
 9966 FORMAT(1X,I2,10(1PE13.5))
 9984 FORMAT(//)
      RETURN
  500 CONTINUE
      SCALER=1.0/SCALE
      DO 751 IS=ISTRT,IM,20
      IE=IS+19
      IF(IE.GT.IM) IE=IM
      IDIF=IE-IS+1
      DO 600 I=1,IDIF
  600 NUM(I)=IS+I-1
      PRINT 9991, (NUM(I),I=1,IDIF)
 9991 FORMAT(3X,20I6,/)
      JMORKM=JM+KK
      DO 752 JORK=1,JMORKM
      IF(JM.NE.0) L=JMORKM+1-JORK
      IF(KK.NE.0) L=JORK
      DO 753 I=1,IDIF
      PLINE(I)=ARRAY(IS+I-1,L)*SCALER
  753 CONTINUE
      PRINT 9967,L,(PLINE(I),I=1,IDIF)
  752 CONTINUE
      PRINT 9984
  751 CONTINUE
 9967 FORMAT(1X,I3,1X,20F6.2)
      RETURN
      END
```

```fortran
*DECK ODAM
      SUBROUTINE ODAM
C
C===============================================================================
C                                                                   ===========
C  ODAM (OCEAN DIRECT ACCESS MANAGER) IS A SET OF ROUTINES          ===========
C      WHICH IS RESPONSIBLE FOR HANDLING THE TRANSFER OF DATA       ===========
C      FROM CENTRAL MEMORY TO DISC AND VICE VERSA.  FOR THIS        ===========
C      PURPOSE, IT USES THE GFDL LOCAL UTILITIES, QDAM (QUEUED      ===========
C      DIRECT ACCESS MANAGER), WHERE:                               ===========
C        LU   =I/O UNIT NUMBER                                      ===========
C        NTOT =LENGTH OF UNIT, IN WORDS                             ===========
C        NBLK =LENGTH OF EACH BLOCK ON THE UNIT                     ===========
C        NBUF =NUMBER OF BUFFERS SUPPLIED TO THE UNIT               ===========
C        NWDS =NUMBER OF WORDS TO TRANSFER                          ===========
C        NFRST=UNIT ADDRESS OF THE FIRST WORD TO BE TRANSFERRED     ===========
C        A    =ORIGINATION/DESTINATION ARRAY IN MEMORY              ===========
C                                                                   ===========
C===============================================================================
C
C--------------------------------------------------------------------
C  DEMENSION ARRAY PASSED IN ARGUMENT  (..NOTE.. 8 IS ARBITRARY)
C--------------------------------------------------------------------
C
      DIMENSION A(8)
C
C--------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C--------------------------------------------------------------------
C
C--------------------------------------------------------------------
C  INITIALIZE ONE DISC UNIT
C--------------------------------------------------------------------
C
      ENTRY OSTART(LU,NTOT,NBLK,NBUF)
      CALL QSTART(LU,NTOT,NBLK,NBUF)
      CALL QSTART(LU,NTOT/2,NBLK/2,NBUF)
      RETURN
C
C--------------------------------------------------------------------
C  INITIATE A DISC-TO-MEMORY TRANSFER (BEGIN DATA FEED TO BUFFER)
C--------------------------------------------------------------------
C
      ENTRY OFIND(LU,NWDS,NFRST)
      CALL QFIND(LU,NWDS,NFRST)
      CALL QFIND(LU,NWDS/2,(NFRST-1)/2+1)
      RETURN
C
C--------------------------------------------------------------------
C  COMPLETE A DISC-TO-MEMORY TRANSFER (FINISH DATA FEED TO BUFFER AND
C      TRANSFER INTO MEMORY)
C--------------------------------------------------------------------
C
      ENTRY OGET(LU,NWDS,NFRST,A)
      CALL QGET(LU,NWDS,NFRST,A)
      CALL QGET(LU,NWDS/2,(NFRST-1)/2+1,A)
      RETURN
C
C--------------------------------------------------------------------
C  INITIATE A MEMORY-TO-DISC TRANSFER (TRANSFER DATA TO BUFFER AND
C      BEGIN A FEED TO THE DISC)
C--------------------------------------------------------------------
C
      ENTRY OPUT(LU,NWDS,NFRST,A)
      CALL QPUT(LU,NWDS,NFRST,A)
      CALL QPUT(LU,NWDS/2,(NFRST-1)/2+1,A)
      RETURN
C
C--------------------------------------------------------------------
C  TERMINATE ALL I/O ON THE SPECIFIED UNIT
C     (GENERATE A SUMMARY REPORT FOR I/O ACTIVITY WHEN CLOSING UNIT 15)
C--------------------------------------------------------------------
C
      ENTRY OCLOSE(LU)
      CALL QSNAP(LU)
      IF(LU.EQ.15)CALL QSUMMARY
      RETURN
C
      END
      SUBROUTINE ODAM
C
C===============================================================================
C                                                                   ===========
C  ODAM (OCEAN DIRECT ACCESS MANAGER) IS A SET OF ROUTINES WHICH    ===========
```

```
C       SIMULATES MEMORY/DISC DATA TRANSFER MANAGEMENT, USING THE         ***********
C       CENTRAL MEMORY ARRAY, "BIG" AS A VIRTUAL DISC. THIS RESULTS       ***********
C       IN A TOTALLY CORE CONTAINED SYSTEM. THE ARGUMENTS, IN TERMS       ***********
C       OF NORMAL DISC I/O, ARE:                                          ***********
C         LU    =I/O UNIT NUMBER                                          ***********
C         NTOT  =LENGTH OF UNIT, IN WORDS                                 ***********
C         NBLK  =LENGTH OF EACH BLOCK ON THE UNIT                         ***********
C         NBUF  =NUMBER OF BUFFERS SUPPLIED TO THE UNIT                   ***********
C         NWRS  =NUMBER OF WORDS TO TRANSFER                              ***********
C         NFRST=UNIT ADDRESS OF THE FIRST WORD TO BE TRANSFERRED          ***********
C         A     =ORIGINATION/DESTINATION ARRAY IN MEMORY                  ***********
C                                                                         ***********
C**************************************************************************************
C
C-----------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C-----------------------------------------------------------------------
C
*CALL PARAM
C
C-----------------------------------------------------------------------
C  DEFINE AND EQUIVALENCE LOCAL DATA
C-----------------------------------------------------------------------
C
      PARAMETER(N11=5,N12=IMT*JMT*NKFLDS,
     * NSL=((NT+2)*IMTKM+LBC*IMT)*JMT,NTB=N11+N12+3*NSL)
      COMMON /BG/ BIG(NTB)
      DIMENSION A(8),C(8)
      EQUIVALENCE (BIG,C)
C
C-----------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C-----------------------------------------------------------------------
C
      ENTRY OSTART(LU,NTOT,NBLK,NBUF)
C
C-----------------------------------------------------------------------
C  FORCE THE ARRAY "BIG" TO BE MAPPED ON LARGE PAGES
C-----------------------------------------------------------------------
C
      IF(LU.NE.11) RETURN
      LEN=NTB/512+1
      LEN=NTB/1024+1
      CALL Q5MAPIN('DROPF','VBA=',BIG,'LEN=',LEN,'LPAGE')
      RETURN
      ENTRY OFIND(LU,NWRS,NFRST)
      RETURN
C
C-----------------------------------------------------------------------
C  VIRTUAL DISC-TO-MEMORY TRANSFER
C-----------------------------------------------------------------------
C
      ENTRY OGET(LU,NWRS,NFRST,A)
      NSKP=0
      IF(LU.GT.11) NSKP=NSKP+N11
      IF(LU.GT.12) NSKP=NSKP+N12
      NS=NFRST+NSKP
      IF(LU.GE.13) NS=NSKP+(3*((NFRST-1)/NWRS)+(LU-13))*NWRS+1
      NE=NS+NWRS-1
      DO 100 N=NS,NE
        A(N-NS+1)=C(N)
 100  CONTINUE
      RETURN
C
C-----------------------------------------------------------------------
C  MEMORY-TO-VIRTUAL DISC TRANSFER
C-----------------------------------------------------------------------
C
      ENTRY OPUT(LU,NWRS,NFRST,A)
      NSKP=0
      IF(LU.GT.11) NSKP=NSKP+N11
      IF(LU.GT.12) NSKP=NSKP+N12
      NS=NFRST+NSKP
      IF(LU.GE.13) NS=NSKP+(3*((NFRST-1)/NWRS)+(LU-13))*NWRS+1
      NE=NS+NWRS-1
      DO 120 N=NS,NE
        C(N)=A(N-NS+1)
 120  CONTINUE
      RETURN
      ENTRY OCLOSE(LU)
      RETURN
C
C-----------------------------------------------------------------------
C  INITIALIZE VIRTUAL DISC FROM TAPE
```

```
C------------------------------------------------------------------------
C
      ENTRY ORD(LO)
      READ(LO)BIG
      RETURN
C
C------------------------------------------------------------------------
C  SAVE VIRTUAL DISC TO TAPE
C------------------------------------------------------------------------
C
      ENTRY OWRT(LO)
      WRITE(LO)BIG
      RETURN
      END
```

```
*DECK FILTER
      SUBROUTINE FILTER(S,IM,MM,N,ISS)
C
C*****************************************************************************
C                                                                 ***********
C  FILTER FOURIER ANALYSES THE ARRAYS OF VARIOUS                  ***********
C         PHYSICAL QUANTITIES, THEN TRUNCATES THE SERIES AND      ***********
C         RESYNTHESIZES THE FILTERED QUANTITIES WHERE:            ***********
C              S  =THE STRING TO BE FILTERED                      ***********
C              IM =THE LENGTH OF S                                ***********
C              MM =1 (COSINE SERIES, BNDRY PTS=0)                 ***********
C                 =2 (SINE SERIES, DERIV AT BNDRY=0)              ***********
C                 =3 (FULL SERIES, CYCLIC)                        ***********
C              N  =NUMBER OF WAVES TO KEEP                        ***********
C              ISS=0 (CANT USE FOURIER COEFS FROM PREVIOUS CALL)  ***********
C              ISS>0 (CAN  USE FOURIER COEFS FROM PREVIOUS CALL)  ***********
C                                                                 ***********
C*****************************************************************************
C
C-----------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C-----------------------------------------------------------------------
C
*CALL PARAM
C
C-----------------------------------------------------------------------
C  DEFINE LOCAL DATA AND DIMENSION ARGUMENT ARRAYS
C-----------------------------------------------------------------------
C
      PARAMETER (IMTX2=IMT*2,NI=IMT)
      PARAMETER (IMTD2=IMT/2,LQMSUM=IMTD2*(IMT-IMTD2),LHSUM=IMT*IMTP1/2)
      PARAMETER (IMTX4=IMT*4,IMTX8=IMT*8,IMTIMT=IMT*IMT)
      PARAMETER (IMTP1X2=IMTP1*2)
C
C     COSSAV MUST REMAIN FULL PRECISION IF MOST OF FILTER IS MADE HALF-P
      REAL COSSAV
C
      DIMENSION ICBASE(IMTP1),IDBASE(IMTP1),IND(IMTP1)
      DIMENSION COSSAV(LQMSUM),DENOMSAV(LHSUM),COSNPI(IMT)
      DIMENSION CIRCLE(4)
      DIMENSION INDX(IMTX8),COF(IMTX8)
      DIMENSION COSINE(IMTX8),FTARR(IMTIMT)
      DIMENSION DENOM(IMTX4)
      DIMENSION S(IMT),SPRIME(IMT)
      DIMENSION DIFF(IMTP1X2)
C
      REAL TO,T1,T2,TIME
      COMMON /TIME/ TIME(10)
C
      LOGICAL INITDONE/.FALSE./
C
      DATA PI/3.141592653589793/, CIRCLE/0.,-1.,0.,1./
C
C-----------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C-----------------------------------------------------------------------
C
      CALL GETIME(TO,T2)
      IF (IM.LT.1 .OR. MM.LT.1 .OR. MM.GT.3 .OR. N.LT.0 .OR. ISS.LT.0)
     * GOTO 4000
C
      IF (INITDONE) GOTO 90
C
C  THIS SECTION SETS UP TABLES FOR FILTER; IT MUST BE CALLED ONCE PER
C    EXECUTION OF OCEAN
C
C  NOTE: LQMSUM IS THE SUM OF (IM-1)/2 FOR IM=1,IMTP1
C        LHSUM IS THE SUM OF IM-1 FOR IM=1,IMTP1
C
      IMSAVE=IM
C
C  ASSEMBLE INDEX ARRAY
C
      DO 10 I=1,IMTP1
      IND(I)=I
 10   CONTINUE
C
C  CALCULATE AND SAVE ALL COSINES WHICH WILL BE NEEDED
C
      IBASE=0
      JBASE=0
      DO 50 IM=1,IMTP1
      FIMR=1.0/FLOAT(IM)
C
```

```
      IMM1=IM-1
      IF (IMM1.EQ.0) GOTO 25
      DO 20 I=1,IMM1
      DENOMSAV(IBASE+I)=1.0/(1.0-COS(PI*FLOAT(I)*FIMR))
 20   CONTINUE
 25   IDBASE(IM)=IBASE
      IBASE=IBASE+IMM1
C
      IMQC=(IM-1)/2
      IF (IMQC.EQ.0) GOTO 35
      DO 30 I=1,IMQC
      COSSAV(JBASE+I)=COS(PI*FLOAT(I)*FIMR)
 30   CONTINUE
 35   ICBASE(IM)=JBASE
      JBASE=JBASE+IMQC
C
 50   CONTINUE
C
C  CALCULATE ADJUSTMENTS FOR GENERAL FOURIER CASE IF IM=2*N
C
      DO 60 IM=1,IMT
      COSNPI(IM)=CIRCLE(MOD(IM-1,4)+1)
 60   CONTINUE
      INITDONE=.TRUE.
      IM=IMSAVE
C
C
C  CALCULATE SOME USEFUL CONSTANTS
C
 90   IF(MM.EQ.2 .AND. N.EQ.0) THEN
        DO 92 I=1,IM
          S(I)=0.0
 92     CONTINUE
        GO TO 3950
      ENDIF
      IF (MM.EQ.1) THEN
        NMAX=N-1
      ELSE
        NMAX=N
      ENDIF
      NMAXP1=NMAX+1
C
      IF (MM.EQ.2) THEN
        LCY=2*(IM+1)
        FNORM=2.0/FLOAT(IM+1)
      ELSE
        LCY=2*IM
        FNORM=2.0/FLOAT(IM)
      ENDIF
      LH=LCY/2
      LHM1=LH-1
      LQM=(LH-1)/2
      L2CY=2*LCY
      LCYM1=LCY-1
      LCYP1=LCY+1
C
      IMX2=IM*2
      IMX4=IM*4
      IMX8=IM*8
C
C  AVERAGE INCOMING ARRAY
C
      SSUM=0.0
      DO 100 I=1,IM
 100  SSUM=SSUM + S(I)
C
C     MM = 1  DERIVATIVE MUST BE ZERO AT BOUNDARIES (COSINE)
C     MM = 2  VALUE MUST BE ZERO AT BOUNDARIES (SINE)
C     MM = 3  CYCLIC BOUNDARY CONDITIONS (GENERAL FOURIER SERIES)
C
      FIM=FLOAT(IM)
      FIMR=1.0/FIM
      STEMP=SSUM*FIMR
      IF (N.GT.1 .OR. MM.NE.1) GO TO 400
      DO 300 I=1,IM
 300  S(I)=STEMP
      GO TO 3950
 400  CONTINUE
      IF(MM.NE.2) THEN
        DO 450 I=1,IM
        S(I)=S(I)-STEMP
 450    CONTINUE
      ENDIF
      IF (ISS.GT.0) GO TO 3000
```

```
C
C
C  ASSEMBLE APPROPRIATE 1-CYCLE (2*PI) COSINE ARRAY
C
C  USE STORED 1/4 CYCLE TO CALCULATE FIRST 1/2 CYCLE
      JBASE=ICBASE(LH)
      DO 700 I=1,LQM
      COSINE(I)=COSSAV(JBASE+I)
      COSINE(LH-I)=-COSSAV(JBASE+I)
 700  CONTINUE
C  FILL IN COS(PI/2) IF LH IS EVEN
      IF (2*(LQM+1).EQ.LH) COSINE(LQM+1)=0.0
C  FILL IN COS(PI) IN ANY CASE
      COSINE(LH)=-1.0
C  FILL IN REST OF CYCLE
      DO 710 I=1,LH
      COSINE(LH+I)=-COSINE(I)
 710  CONTINUE
C
C  ASSEMBLE DENOMINATOR ARRAY
C
      IBASE=IDBASE(LH)
      FXA=0.25
      DO 720 I=1,LHM1
      DENOM(I)=FXA*DENOMSAV(IBASE+I)
 720  CONTINUE
      DENOM(LH)=0.125
      DO 721 I=1,LHM1
      DENOM(LH+I)=DENOM(LH-I)
 721  CONTINUE
      NPRINT=0
      DENOM(LCY)=0.0
      DO 730 I=LCYP1,IMX4
      DENOM(I)=DENOM(I-LCY)
 730  CONTINUE
C
C  ASSEMBLE APPROPRIATE SUBSCRIPT ARRAYS
C
C  CALCULATE NEEDED INDICES
C
      IF (MM.EQ.3) THEN
        FACT1=2*NMAX
        FACT2=2*NMAXP1
      ELSE
        FACT1=NMAX
        FACT2=NMAXP1
      ENDIF
      DO 740 I=1,IMX4
      INDX(I)=I*FACT1
      INDX(IMX4+I)=I*FACT2
 740  CONTINUE
C  CALCULATE PARAMETERS FOR REDUCING INDICES
      MAXIND=IMX4*FACT2
      NCYC=(MAXIND-1)/LCY + 1
      MAXNDX=LCY
      IF (MAXNDX.GE.MAXIND) GOTO 790
      DO 750 NPWR=1,NCYC+2
      MAXNDX=2*MAXNDX
      IF (MAXNDX.GE.MAXIND) GOTO 760
 750  CONTINUE
      STOP 'ERROR: FELL THROUGH DO-LOOP TERMINATION AT 750'
 760  DO 770 NP=1,NPWR
      MAXNDX=MAXNDX/2
      DO 765 I=1,IMX8
        IF(INDX(I).GT.MAXNDX) INDX(I)=INDX(I)-MAXNDX
 765  CONTINUE
      WHERE (INDX(1;IMX8).GT.MAXNDX)
        INDX(1;IMX8)=INDX(1;IMX8)-MAXNDX
      ENDWHERE
 770  CONTINUE
 790  CONTINUE
C
C  GATHER COEFFICIENTS
C
 800  DO 810 J=1,IMX8
      COF(J)=COSINE(INDX(J))
 810  CONTINUE
C
      GO TO (1000,1300,1600),MM
C
C  ASSEMBLE TRANSFORMATION ARRAY WHICH WILL FILTER S
C
C
C  COSINE TRANSFORM
```

```
 1000 IOFF1=LCY
      IOFF2=LCY+IMX4
      FXA=0.5
      DO 1200 J=1,IM
      JOFF=(J-1)*IMT
      DO 1100 I=1,IM
      FTARR(JOFF+I)=(COF(I-J+IOFF1)-COF(I-J+IOFF2))*DENOM(I-J+IOFF1)
     *              +(COF(I+J-1)-COF(IMX4+I+J-1))*DENOM(I+J-1) - FXA
 1100 CONTINUE
      FTARR(JOFF+J)=FTARR(JOFF+J)+0.5*FLOAT(NMAX)+0.25
 1200 CONTINUE
      GOTO 3000
C
C  SINE TRANSFORM
 1300 IOFF1=LCY
      IOFF2=LCY+IMX4
      DO 1500 J=1,IM
      JOFF=(J-1)*IMT
      DO 1400 I=1,IM
      FTARR(JOFF+I)=(COF(I-J+IOFF1)-COF(I-J+IOFF2))*DENOM(I-J+IOFF1)
     *              -(COF(I+J)-COF(IMX4+I+J))*DENOM(I+J)
 1400 CONTINUE
      FTARR(JOFF+J)=FTARR(JOFF+J)+0.5*FLOAT(NMAX)+0.25
 1500 CONTINUE
      GOTO 3000
C
C  GENERAL FOURIER TRANSFORM
 1600 IF (2*N.EQ.IM) THEN
        GENADJ=0.5
      ELSE
        GENADJ=0.0
      ENDIF
      IOFF1=LCY
      IOFF2=LCY+IMX4
      FXA=2.0
      FXB=0.5
      DO 1800 J=1,IM
      JOFF=(J-1)*IMT
      DO 1700 I=1,IM
      FTARR(JOFF+I)=
     * (FXA*(COF(I-J+IOFF1)-COF(I-J+IOFF2)))*DENOM(2*I-2*J+IOFF1)
     * -FXB - GENADJ*COSNPI(I)*COSNPI(J)
 1700 CONTINUE
      FTARR(JOFF+J)=FTARR(JOFF+J)+FLOAT(NMAX)+0.5
 1800 CONTINUE
C     GOTO 3000
C
C  FILTER S
C
 3000 SPRIME(1;IM)=0.0
      DO 3100 I=1,IM
      IOFF=(I-1)*IMT
      DO 3100 J=1,IM
C  NOTE THAT FTARR(J,I)=FTARR(I,J), SO FOLLOWING IS LEGAL
      SPRIME(J)=SPRIME(J)+S(I)*FTARR(IOFF+J)
 3100 CONTINUE
      SPRIME(1;IM)=FNORM*SPRIME(1;IM)
      IF(MM.EQ.2) THEN
        DO 3150 I=1,IM
          S(I)=SPRIME(I)
 3150   CONTINUE
        GO TO 3950
      ENDIF
C
 3700 SSM=0.0
      DO 3800 I=1,IM
      SSM=SSM+SPRIME(I)
 3800 CONTINUE
      SSM=(SSUM-SSM)*FIMR
      DO 3900 I=1,IM
      S(I)=SSM+SPRIME(I)
 3900 CONTINUE
 3950 CONTINUE
      CALL GETIME(T1,T2)
      TIME(9)=TIME(9)+T1-TO
      RETURN
 4000 PRINT 4001, IM,MM,N,ISS
 4001 FORMAT (' BAD ARGUMENT(S) IN CALL TO FILTER'/' IM,MM,N,ISS = ',
     *  4I10)
      STOP 'BAD ARGUMENT(S) IN CALL TO FILTER'
      END
```

```
*DECK FINDEX
      SUBROUTINE FINDEX(FKXX,JJMAX,KMAX,JF1,JF2,IMAX,ISF,IEF)
C
C***********************************************************************
C                                                            ***********
C  FINDEX FINDS AND PRINTS STARTING AND ENDING INDICES       ***********
C         FOR FILTERING, WHERE:                              ***********
C              FKXX =FIELD OF MAXIMUM LEVELS FOR THE QUANTITY ***********
C                    BEING FILTERED                          ***********
C              JJMAX=NUMBER OF ROWS TO BE FILTERED           ***********
C              KMAX =MAXIMUM NUMBER OF LEVELS TO BE FILTERED ***********
C              JF1  =LAST ROW IN THE SOUTH TO BE FILTERED    ***********
C              JF2  =FIRST ROW IN THE NORTH TO BE FILTERED   ***********
C              IMAX =MAXIMUM I INDEX TO BE FILTERED          ***********
C              ISF  =RETURNED VALUES OF STARTING INDICES     ***********
C              IEF  =RETURNED VALUES OF ENDING INDICES       ***********
C                                                            ***********
C***********************************************************************
C
C-----------------------------------------------------------------------
C  DEFINE GLOBAL DATA
C-----------------------------------------------------------------------
C
*CALL PARAM
C
C-----------------------------------------------------------------------
C  DEFINE LOCAL DATA AND DIMENSION ARGUMENT ARRAYS
C-----------------------------------------------------------------------
C
      DIMENSION FKXX(IMT,JMT)
     * ,ISF(JJMAX,LSEGF,KMAX),IEF(JJMAX,LSEGF,KMAX)
     * ,IIS(LSEGF+1),IIE(LSEGF+1)
C
C-----------------------------------------------------------------------
C  BEGIN EXECUTABLE CODE
C-----------------------------------------------------------------------
C
C FIND START AND END INDICES
C
      JJ = 0
      DO 100 J = JFRST,JMTM1
        IF (J.GT.JF1 .AND. J.LT.JF2) GOTO 100
        JJ = JJ+1
        DO 80 K = 1,KMAX
          DO 30 L = 1,LSEGF+1
            IIS(L) = 0
            IIE(L) = 0
 30       CONTINUE
          L = 1
          IF (FKXX(2,J).GE.K) THEN
            IIS(1) = 2
          ENDIF
          DO 50 I = 2,IMAX-1
            IF (FKXX(I-1,J).LT.K .AND. FKXX(I,J).GE.K) THEN
              IIS(L) = I
            ENDIF
            IF (FKXX(I,J).GE.K .AND. FKXX(I+1,J).LT.K) THEN
              IF (I.NE.IIS(L) .OR. (I.EQ.2 .AND. FKXX(1,J).GE.K))THEN
                IIE(L) = I
                L = L+1
              ELSE
                IIS(L) = 0
              ENDIF
            ENDIF
 50       CONTINUE
          IF (FKXX(IMAX-1,J).GE.K .AND. FKXX(IMAX,J).GE.K) THEN
            IIE(L) = IMAX-1
            L = L+1
          ENDIF
          LM = L-1
          IF (LM.GT.1) THEN
            IF (IIS(1).EQ.2 .AND. IIE(LM).EQ.IMAX-1
     *                  .AND. FKXX(1,J).GE.K) THEN
              IIS(1) = IIS(LM)
              IIE(1) = IIE(1) + IMAX-2
              IIS(LM) = 0
              IIE(LM) = 0
              LM = LM-1
            ENDIF
          ENDIF
          IF (LM .GT. LSEGF) THEN
            PRINT 1000, J, K
            STOP 34
          ENDIF
```

```
1000      FORMAT (1X,'LSEGF NOT LARGE ENOUGH.  J=',I4,'    K=',I3)
          DO 70 L = 1,LSEGF
             ISF(JJ,L,K) = IIS(L)
             IEF(JJ,L,K) = IIE(L)
 70       CONTINUE
 80     CONTINUE
 100  CONTINUE
C
C PRINT THEM
C
      LLAST=LSEGF
      IF (LLAST .GT. 11) LLAST=11
      JJ=JJ+1
      DO 200 J=JMTM1,JFRST,-1
         IF (J.GT.JF1 .AND. J.LT.JF2) GO TO 200
         JJ = JJ-1
         IF (KMAX .GT. 1) THEN
            PRINT 1010,J
            DO 150 K=1,KMAX
               PRINT 1011,K,(ISF(JJ,L,K),IEF(JJ,L,K),L=1,LLAST)
 150        CONTINUE
         ELSE
            PRINT 1011,J,(ISF(JJ,L,1),IEF(JJ,L,1),L=1,LLAST)
         ENDIF
 200  CONTINUE
 1010 FORMAT (/1X,'INDICES FOR ROW ',I3,':')
 1011 FORMAT (1X,I9,3X,11(I5,I4))
      RETURN
      END
```

L. Forming and running models

The program tape includes the following files with record lengths as specified, and a block size of 4000:

FILE	RECL	
1	100	base code
2	80	UPDOC
3	80	KNUDSN
4	100	Model 1 update file
5	80	Model 1 data file
6	100	Model 2 update file
7	80	Model 2 data file
8	100	Model 3 update file
9	80	Model 3 data file

The formation of a specific model involves several steps, listed below:

1. determine the code options needed and enter them in record 1 of unit 10 (see section H)

2. determine other code alterations needed and enter the update lines on record 2 and on, in unit 10

3. run "KNUDSN" and place the update lines generated on unit 50, into their proper place in unit 10

4. place the base code on unit 20

5. run "UPDOC"

6. compile the code generated on unit 20 by UPDOC

7. establish the namelist input file for execution of the model

8. run the model

There are several places in the base code which _must_ be updated in step 2 above to obtain a successful compilation. As these lines currently stand, they have "??" on the right side of the FORTRAN assignment statements. They consist of:

SEQ. NO.	FUNCTION
000200000	PARAMETER assignments
111100000	designation of the latitude of the southern wall
111800000	designation of meridional grid spacing
112900000	designation of zonal grid spacing
114400000	designation of vertical grid spacing
133400000	designation of bottom topography
152400000	designation of zonal wind driving
152500000	designation of meridional wind driving
249100000	designation of rows to be printed for diagnosis
249400000	designation of columns to be printed for diagnosis
603800000 through 604000000	data for the equation of state from KNUDSN

Replacements for all of these lines must appear in the update file, along with any other changes to the base code that are needed. Several sample models have been constructed to illustrate the procedure and are described below.

Model 1 is a duplicate of the sample model Semtner supplies in his report. It consists of a closed, rectangular basin with a shelf along the western wall and a small island in the center. The update file, NAMELIST input data file, and output file for this model are given on the following pages. The option record specifies that an island is included, that the model is to be run core contained, and that comments are desired in the code listing. The updates which follow simply alter the base code to yield a model like that of Semtner's sample. Note that many of the updates arise because an alternate equation of state is used. There are differences in the manner in which the computation is handled between the two models, so that the answers will not be exactly the same. In particular, Semtner's version uses a relaxation which does not converge as rapidly as that used here, and, in fact, is cut off on most timesteps at 26 scans, before convergence occurs. In the present model, the convergence is completed for all but a few timesteps. The result is that the answers are somewhat different but presumably more accurate in the present model.

The NAMELIST input data file specifies the following:

NFIRST=1	start from scratch
NLAST=42	stop after timestep 42
NNERGY=20	print fields every 20 timesteps
NTSI=1	print single line of information every timestep
NMIX=10	do a mixing timestep every 10 timesteps
AMF=1.E9	horizontal mixing of momentum
AHF=2.E7	horizontal mixing of heat, salinity, tracers
FKPMF=1.0	vertical mixing of momentum
FKPHF=1.0	vertical mixing of heat, salinity, tracers
DTTSF=172800.	length of timestep on temp., sal., tracers
DTUVF=7200.	length of timestep on internal mode velocities
DTSFF=7200.	length of timestep on stream function
MXSCAN=25	cut off relaxation at 25 scans if not converged
SORF=1.50	coefficient of over-relaxation
CRITF=4.E9	criterion for convergence of relaxation
ACORF=0.5	weight forward and backward timestep equally in the semi-implicit Coriolis term
TINITF=...	initial values of temperature, salinity, tracers (note.. for purposes of precision, salinity is carried in the model in units of parts per part, with .035 subtracted off, however TINITF is expected in parts per part; .035 is subtracted after read-in)
ISIS=13,...	the island and its perimeter points are included in a box between I=13-16 and J=9-12

UPDATE FOR MODEL 1:

```
OPT=I,K,C
      PARAMETER (IMT=25,JMT=20,KM=6,NT=2,LSEG=2,NISLE=1,LBC=2           000200000
      EB=.FALSE.                                                        107400000
      SWLDEG=16.0                                                       111100000
        DYT(J)=2.0                                                      111800000
        DXT(I)=2.0                                                      112900000
      DZ(1)=  50.E2                                                     114400000
      DZ(2)= 200.E2                                                     114400010
      DZ(3)= 500.E2                                                     114400020
      DZ(4)= 800.E2                                                     114400030
      DZ(5)=1000.E2                                                     114400040
      DZ(6)=1200.E2                                                     114400050
        FKMP(I,J)=KM                                                    133400000
      DO 702 J=2,JMTM1                                                  133500010
        FKMP(2,J)=2                                                     133500020
      DO 702 I=3,6                                                      133500030
        FKMP(I,J)=I-1                                                   133500040
 702  CONTINUE                                                          133500050
      DO 704 J=7,14                                                     133500060
        FKMP(14,J)=5                                                    133500070
        FKMP(15,J)=5                                                    133500080
 704  CONTINUE                                                          133500090
      DO 706 J=8,13                                                     133500100
        FKMP(14,J)=4                                                    133500110
        FKMP(15,J)=4                                                    133500120
 706  CONTINUE                                                          133500130
      DO 708 J=9,12                                                     133500140
      DO 708 I=13,16                                                    133500150
        FKMP(I,J)=4                                                     133500160
 708  CONTINUE                                                          133500170
      FKMP(14,10)=0                                                     133500180
      FKMP(15,10)=0                                                     133500190
      FKMP(14,11)=0                                                     133500200
      FKMP(15,11)=0                                                     133500210
      DO 710 J=1,JMT                                                    133500220
      DO 710 I=1,IMT                                                    133500230
        IF(I-J.GE.18) FKMP(I,J)=0                                       133500240
 710  CONTINUE                                                          133500250
      DO 712 J=1,JMT                                                    133500260
      DO 712 I=1,IMT                                                    133500270
        IF(I+J.GE.39) FKMP(I,J)=0                                       133500280
 712  CONTINUE                                                          133500290
        FX=-COS(PI*FLOAT(J-2)/FLOAT(JMT-4))                             151900010
          WSX(I)=FX                                                     152400000
          WSY(I)=0.0                                                    152500000
      MTEST=MOD(J,7)                                                    249100000
      IPRT=IMT                                                          249400000
C****************  SPECIAL PATCH FOR PRESCRIBED SURFACE VALUES  ******  423500010
      FXA=SIN(2.*PI*FLOAT(J-2)/FLOAT(JMT-3))                            423500020
      IF(NERGY.EQ.1 .AND. MXP.EQ.0) THEN                                423500030
        FXB=CST(J)*DYT(J)*DZ(1)/C2DTTS                                  423500040
        DO 876 I=2,IMTM1                                                423500050
          TTDTOT(5,1)=TTDTOT(5,1)                                       423500060
     *    +(27.-25.*FLOAT(J-2)/FLOAT(JMT-3)-TA(I,1,1))*FXB*DXT(I)*FM(I,1) 423500070
          TTDTOT(5,2)=TTDTOT(5,2)                                       423500080
     *    +(.0007*FXA-TA(I,1,2))*FXB*DXT(I)*FM(I,1)                     423500090
 876    CONTINUE                                                        423500100
      ENDIF                                                             423500110
      DO 878 I=2,IMTM1                                                  423500120
        TA(I,1,1)=(27.-25.*FLOAT(J-2)/FLOAT(JMT-3))*FM(I,1)             423500130
        TA(I,1,2)=.0007*FXA*FM(I,1)                                     423500140
 878  CONTINUE                                                          423500150
C****************  END SPECIAL PATCH  *********************************  423500160
-600300000,601500000
*CALL ONEDIM                                                            602200000
-603100000,604000000
-604700000,606300000
      DO 100 K=1,KM                                                     606300010
        FACTOR=5891.+2DZZ(K)/1013.                                      606300020
      DO 100 I=1,IMT                                                    606300030
        RHO(I,K)=FACTOR+3000.*(SX(I,K)+.035)+(38.-0.375*TX(I,K))*TX(I,K) 606300040
 100  CONTINUE                                                          606300050
      DO 200 K=1,KM                                                     606300060
      DO 200 I=1,IMT                                                    606300070
        RHO(I,K)=(1779.5+(11.25-0.0745*TX(I,K))*TX(I,K)                 606300080
     *           -(3800.+10.*TX(I,K))*(SX(I,K)+.035))/RHO(I,K)          606300090
 200  CONTINUE                                                          606300110
      DO 300 K=1,KM                                                     606300120
      DO 300 I=1,IMT                                                    606300130
        RHO(I,K)=1.0/(0.698+RHO(I,K))-1.02                              606300140
 300  CONTINUE                                                          606300150
-607000000,608900000
```

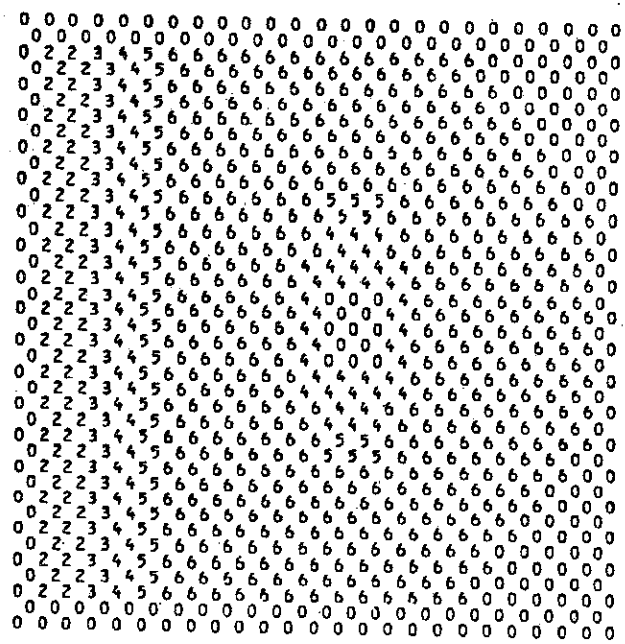
```
-609100000,609900000
      KD=IND-1                                                          609900010
      DO 400 K=1,KM                                                     609900020
        L=K+MOD(K+KD,2)                                                 609900030
        FACTOR=5891.+2DZZ(L)/1013.                                      609900040
      DO 400 I=1,IMT                                                    609900050
        RHO(I,K)=FACTOR+3000.*(SX(I,K)+.035)+(38.-0.375*TX(I,K))*TX(I,K) 609900060
 400  CONTINUE                                                          609900070
      DO 500 K=1,KM                                                     609900080
      DO 500 I=1,IMT                                                    609900090
        RHO(I,K)=(1779.5+(11.25-0.0745*TX(I,K))*TX(I,K)                 609900110
     *           -(3800.+10.*TX(I,K))*(SX(I,K)+.035))/RHO(I,K)          609900120
 500  CONTINUE                                                          609900130
      DO 600 K=1,KM                                                     609900140
      DO 600 I=1,IMT                                                    609900150
        RHO(I,K)=1.0/(0.698+RHO(I,K))-1.02                              609900160
 600  CONTINUE                                                          609900170
-610500000,616600000
```

DATA FOR MODEL 1:

```
 &CONTRL NFIRST=1,NLAST=42,NNERGY=20,NTSI=01,NMIX=10,&END
 &EDDY AMF=1.E9,AHF=2.E7,FKPMF=1.0,FKPHF=1.0,&END
 &TSTEPS DTTSF=172800.,DTUVF=7200.,DTSFF=7200.,&END
 &PARMS MXSCAN=25,SORF=1.50,CRITF=.4E10,ACORF=.5,&END
 &TSPROF TINITF=4.0,4.0,4.0,4.0,4.0,4.0,
 .0349,.0349,.0349,.0349,.0349,.0349,&END
 &IBOX ISIS=13,IEIS=16,JSIS=9,JEIS=12,&END
```

```
&CONTRL
NFIRST   =  1,
NLAST    =  42,
NNERGY   =  20,
NMIX     =  10,
NWRITE   =  1000000,
NDW      =  1000000,
NTSI     =  1,
NA       =  0,
NB       =  0,
NC       =  0,
&END

&EDDY
AMF      =  .1E+10,
AHF      =  .2E+08,
FKPMF    =  .1E+01,
FKPHF    =  .1E+01,
&END

&TSTEPS
DTTSF    =  .1728E+06,
DTUVF    =  .72E+04,
DTSFF    =  .72E+04,
&END

&PARMS
ACORF    =  .5E+00,
MXSCAN   =  25,
SORF     =  .15E+01,
CRITF    =  .4E+10,
&END

GRID BOX THICKNESS  'DZ'
  .50000E+04   .20000E+05   .50000E+05   .80000E+05   .10000E+06   .12000E+06

GRID POINT SEPARATION  'DZZ'
  .25000E+04   .12500E+05   .35000E+05   .65000E+05   .90000E+05   .11000E+06   .60000E+05

DEPTH OF BOX BOTTOM  'ZDZ'
  .50000E+04   .25000E+05   .75000E+05   .15500E+06   .25500E+06   .37500E+06

DEPTH OF GRID POINT  'ZDZZ'
  .25000E+04   .15000E+05   .50000E+05   .11500E+06   .20500E+06   .31500E+06   .37500E+06

LATITUDE OF T,S POINTS (RADIANS)  'PHIT'
  .26180E+00   .29671E+00   .33161E+00   .36652E+00   .40143E+00   .43633E+00   .47124E+00   .50615E+00   .54105E+00   .57596E+00
  .61087E+00   .64577E+00   .68068E+00   .71558E+00   .75049E+00   .78540E+00   .82030E+00   .85521E+00   .89012E+00   .92502E+00

LATITUDE OF U,V POINTS (RADIANS)  'PHI'
  .27925E+00   .31416E+00   .34907E+00   .38397E+00   .41888E+00   .45379E+00   .48869E+00   .52360E+00   .55851E+00   .59341E+00
  .62832E+00   .66323E+00   .69813E+00   .73304E+00   .76794E+00   .80285E+00   .83776E+00   .87266E+00   .90757E+00   .94248E+00

COSINE OF T,S LATITUDE  'CST'
  .96593E+00   .95630E+00   .94552E+00   .93358E+00   .92050E+00   .90631E+00   .89101E+00   .87462E+00   .85717E+00   .83867E+00
  .81915E+00   .79864E+00   .77715E+00   .75471E+00   .73135E+00   .70711E+00   .68200E+00   .65606E+00   .62932E+00   .60182E+00

COSINE OF U,V LATITUDE  'CS'
  .96126E+00   .95106E+00   .93969E+00   .92718E+00   .91355E+00   .89879E+00   .88295E+00   .86603E+00   .84805E+00   .82904E+00
  .80902E+00   .78801E+00   .76604E+00   .74314E+00   .71934E+00   .69466E+00   .66913E+00   .64279E+00   .61566E+00   .58779E+00

SINE OF U,V LATITUDE  'SINE'
  .27564E+00   .30902E+00   .34202E+00   .37461E+00   .40674E+00   .43837E+00   .46947E+00   .50000E+00   .52992E+00   .55919E+00
  .58779E+00   .61566E+00   .64279E+00   .66913E+00   .69466E+00   .71934E+00   .74314E+00   .75604E+00   .78801E+00   .80902E+00
```

NUMBER OF LEVELS AT T,S POINTS AND U,V POINTS



SURFACE AREA = 1.533836E+17   VOLUME = 4.615793E+22

```
 START AND END INDICES FOR STREAM FUNCTION
J= 20         0   0             0   0
J= 19         0   0             0   0
J= 18         3  18             0   0
J= 17         3  19             0   0
J= 16         3  20             0   0
J= 15         3  21             0   0
J= 14         3  22             0   0
J= 13         3  23             0   0
J= 12         3  12            17  23
J= 11         3  12            17  23
J= 10         3  12            17  23
J=  9         3  12            17  23
J=  8         3  23             0   0
J=  7         3  22             0   0
J=  6         3  21             0   0
J=  5         3  20             0   0
J=  4         3  19             0   0
J=  3         3  18             0   0
J=  2         0   0             0   0
J=  1         0   0             0   0

&TSPROF
TINITF   =  .4E+01,  .4E+01,  .4E+01,  .4E+01,  .4E+01,  .4E+01,  .349E-01,  .349E-01,  .349E-01,  .349E-01,  .349E-01,  .349E-01,
&END

&IBOX
ISIS     =  13,
IEIS     =  16,
JSIS     =  9,
JEIS     =  12,
&END

TS=    1  YEAR=   .01  DAY=  2.0  ENERGY=  .0            DTEMP= 1.111937E-06  DSALT= 2.056906E-08  SCANS=III
TS=    2  YEAR=   .01  DAY=  4.0  ENERGY= 6.891815E-03  DTEMP= 5.565899E-07  DSALT= 1.028455E-08  SCANS= 25
TS=    3  YEAR=   .02  DAY=  6.0  ENERGY= 3.415497E-02  DTEMP= 7.916117E-09  DSALT= 3.158423E-13  SCANS= 10
TS=    4  YEAR=   .02  DAY=  8.0  ENERGY= 6.164369E-02  DTEMP= 9.023050E-09  DSALT= 3.434125E-13  SCANS= 25
TS=    5  YEAR=   .03  DAY= 10.0  ENERGY= 1.084131E-01  DTEMP= 1.055111E-08  DSALT= 3.675650E-13  SCANS= 25
TS=    6  YEAR=   .03  DAY= 12.0  ENERGY= 1.397517E-01  DTEMP= 1.157102E-08  DSALT= 3.840328E-13  SCANS=  8
TS=    7  YEAR=   .04  DAY= 14.0  ENERGY= 1.671540E-01  DTEMP= 1.185513E-08  DSALT= 3.864932E-13  SCANS=  8
TS=    8  YEAR=   .04  DAY= 16.0  ENERGY= 1.929734E-01  DTEMP= 1.271343E-08  DSALT= 4.036965E-13  SCANS=  8
TS=    9  YEAR=   .05  DAY= 18.0  ENERGY= 1.825518E-01  DTEMP= 1.214641E-08  DSALT= 3.882834E-13  SCANS=  8
TS=   10  YEAR=   .05  DAY= 20.0  ENERGY= 2.040146E-01  DTEMP= 1.261721E-08  DSALT= 3.972806E-13  SCANS=  7
TS=   11  YEAR=   .06  DAY= 22.0  ENERGY= 1.708637E-01  DTEMP= 1.747840E-08  DSALT= 4.928565E-13  SCANS=  7
TS=   12  YEAR=   .07  DAY= 24.0  ENERGY= 1.624436E-01  DTEMP= 1.089264E-08  DSALT= 3.428672E-13  SCANS=  8
TS=   13  YEAR=   .07  DAY= 26.0  ENERGY= 1.610702E-01  DTEMP= 9.326853E-09  DSALT= 3.160610E-13  SCANS= 10
TS=   14  YEAR=   .08  DAY= 28.0  ENERGY= 1.514267E-01  DTEMP= 9.700860E-09  DSALT= 2.997903E-13  SCANS= 10
TS=   15  YEAR=   .08  DAY= 30.0  ENERGY= 1.585698E-01  DTEMP= 8.490989E-09  DSALT= 2.710977E-13  SCANS=  9
TS=   16  YEAR=   .09  DAY= 32.0  ENERGY= 1.477907E-01  DTEMP= 7.979076E-09  DSALT= 2.481277E-13  SCANS=  8
TS=   17  YEAR=   .09  DAY= 34.0  ENERGY= 1.590386E-01  DTEMP= 7.120141E-09  DSALT= 2.395921E-13  SCANS=  8
TS=   18  YEAR=   .10  DAY= 36.0  ENERGY= 1.546019E-01  DTEMP= 7.053467E-09  DSALT= 2.455859E-13  SCANS=  7
TS=   19  YEAR=   .10  DAY= 38.0  ENERGY= 1.706460E-01  DTEMP= 6.750960E-09  DSALT= 2.508019E-13  SCANS=  7
TEMPERATURE FOR J =   7 AT TIMESTEP    20
         2      3      4      5      6      7      8      9     10     11     12     13     14     15     16     17     18     19     20     21

  1  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65  19.65
  2   4.09   4.30   4.33   4.35   4.34   4.34   4.34   4.34   4.34   4.34   4.34   4.35   4.33   4.34   4.33   4.34   4.34   4.34   4.34   4.34
  3    .00    .00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00
  4    .00    .00    .00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00
  5    .00    .00    .00    .00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00   4.00
  6    .00    .00    .00    .00    .00   4.00   4.00   4.00   4.00   4.00   4.00   4.00    .00    .00   4.00   4.00   4.00   4.00   4.00   4.00

        22     23     24     25
  1  19.65  19.65  19.65    .00
  2   4.34   4.19   4.09    .00
  3   4.00   4.00   4.00    .00
  4   4.00   4.00   4.00    .00
  5   4.00   4.00   4.00    .00
  6   4.00   4.00   4.00    .00


SALINITY     FOR J =   7 AT TIMESTEP    20
         2      3      4      5      6      7      8      9     10     11     12     13     14     15     16     17     18     19     20     21

  1    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67    .67
  2   -.10   -.09   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08   -.08
  3  10.00  10.00   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10
  4  10.00  10.00  10.00   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10
  5  10.00  10.00  10.00  10.00   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10   -.10
  6  10.00  10.00  10.00  10.00  10.00   -.10   -.10   -.10   -.10   -.10   -.10   -.10  10.00  10.00   -.10   -.10   -.10   -.10   -.10   -.10
```

	22	23	24	25
1	.67	.67	.67	10.00
2	-.08	-.09	-.10	10.00
3	-.10	-.10	-.10	10.00
4	-.10	-.10	-.10	10.00
5	-.10	-.10	-.10	10.00
6	-.10	-.10	-.10	10.00

W VELOCITY FOR J = 7 AT TIMESTEP 20

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00
2	.04	-.21	-.19	-.22	-.20	-.20	-.20	-.20	-.20	-.20	-.20	-.21	-.19	-.20	-.19	-.20	-.20	-.20	-.20	-.16
3	.00	.00	-.12	-.27	-.22	-.19	-.19	-.18	-.18	-.18	-.18	-.23	-.15	-.19	-.15	-.19	-.19	-.20	-.20	-.13
4	.00	.00	.00	-.15	-.22	-.17	-.17	-.16	-.16	-.15	-.14	-.30	-.03	-.16	-.06	-.18	-.17	-.18	-.18	-.12
5	.00	.00	-.00	.00	-.11	-.12	-.15	-.12	-.11	-.11	-.09	-.42	.16	-.12	.08	-.15	-.13	-.14	-.16	-.13
6	.00	.00	-.00	.00	.00	.03	-.10	-.07	-.06	-.06	-.04	-.49	.00	.00	.39	-.09	-.07	-.08	-.10	-.10
7	.00	.00	-.00	.00	-.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00

	22	23	24	25
1	.00	.00	.00	.00
2	-.08	.15	.18	.00
3	-.08	.02	.15	.00
4	-.08	.05	.15	.00
5	-.08	.11	.15	.00
6	-.05	.12	.11	.00
7	.00	.00	.00	.00

U VELOCITY FOR J = 7 AT TIMESTEP 20

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	.00	.78	.81	.88	.84	.81	.78	.74	.73	.73	.75	.74	.65	.70	.75	.75	.71	.70	.70	.72
2	.00	-.11	.01	.11	.07	.04	.01	-.02	-.04	-.04	-.02	-.02	-.12	-.07	-.02	-.02	-.06	-.07	-.06	-.05
3	.00	.00	.00	-.01	.03	.01	-.02	-.05	-.06	-.06	-.04	-.05	-.15	-.09	-.04	-.04	-.09	-.09	-.09	-.09
4	.00	.00	.00	.00	-.00	.01	-.02	-.05	-.07	-.06	-.04	-.05	-.15	-.09	-.04	-.04	-.09	-.09	-.09	-.09
5	.00	.00	.00	.00	.00	-.01	-.02	-.05	-.07	-.06	-.05	-.06	.00	.00	.00	-.04	-.09	-.09	-.09	-.09
6	.00	.00	.00	.00	.00	.00	-.03	-.05	-.07	-.06	-.05	-.06	.00	.00	.00	-.04	-.09	-.09	-.09	-.09

	21	22	23	24	25
1	.80	.95	.84	.00	.00
2	-.01	.01	-.13	.00	.00
3	-.09	-.10	-.06	.00	.00
4	-.09	-.10	-.06	.00	.00
5	-.09	-.09	-.04	.00	.00
6	-.08	-.07	-.02	.00	.00

V VELOCITY FOR J = 7 AT TIMESTEP 20

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	.00	5.30	4.61	3.90	3.72	3.60	3.49	3.43	3.39	3.37	3.36	3.41	3.55	3.40	3.34	3.28	3.25	3.26	3.28	3.31
2	.00	2.13	1.15	.42	.24	.11	.01	-.05	-.09	-.12	-.13	-.07	.08	-.06	-.13	-.20	-.23	-.22	-.20	-.18
3	.00	.00	.00	.50	.28	.15	.04	-.01	-.05	-.08	-.10	-.04	.13	-.03	-.09	-.17	-.20	-.19	-.18	-.17
4	.00	.00	.00	.00	.30	.15	.04	-.01	-.05	-.08	-.10	-.04	.13	-.02	-.09	-.17	-.20	-.19	-.18	-.17
5	.00	.00	.00	.00	.00	.16	.04	-.01	-.05	-.08	-.10	-.04	.00	.00	.00	-.15	-.19	-.19	-.18	-.17
6	.00	.00	.00	.00	.00	.00	.05	-.01	-.05	-.08	-.10	-.04	.00	.00	.00	-.15	-.19	-.19	-.18	-.17

	21	22	23	24	25
1	3.33	3.11	2.58	.00	.00
2	-.16	-.24	-.21	.00	.00
3	-.16	-.15	-.07	.00	.00
4	-.16	-.14	-.07	.00	.00
5	-.16	-.14	-.06	.00	.00
6	-.16	-.13	-.06	.00	.00

TEMPERATURE FOR J = 14 AT TIMESTEP 20

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35	9.35
2	4.02	4.08	4.08	4.08	4.08	4.09	4.09	4.09	4.09	4.09	4.09	4.08	4.08	4.08	4.09	4.08	4.08	4.08	4.08	4.08
3	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00

4	.00	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
5	.00	.00	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
6	.00	.00	.00	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00

	22	23	24	25
1	9.35	9.35	9.35	.00
2	4.07	4.06	4.06	.00
3	4.00	4.00	4.00	.00
4	4.00	4.00	4.00	.00
5	4.00	4.00	4.00	.00
6	4.00	4.00	4.00	.00

SALINITY FOR J = 14 AT TIMESTEP 20

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67	-.67
2	-.10	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11	-.11
3	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10
4	10.00	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10
5	10.00	10.00	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10
6	10.00	10.00	10.00	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10	-.10	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10

	22	23	24	25
1	-.67	-.67	-.67	10.00
2	-.11	-.11	-.11	10.00
3	-.10	-.10	-.10	10.00
4	-.10	-.10	-.10	10.00
5	-.10	-.10	-.10	10.00
6	-.10	-.10	-.10	10.00

W VELOCITY FOR J = 14 AT TIMESTEP 20

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00
2	-.07	-.06	.01	.03	.05	.06	.06	.06	.06	.06	.06	.06	.07	.08	.06	.06	.06	.06	.07	.10
3	.00	.00	-.12	-.10	.01	.04	.05	.06	.06	.06	.06	.04	.08	.15	.04	.06	.06	.06	.09	.13
4	.00	.00	.00	.07	-.03	-.00	.02	.05	.05	.05	.05	.01	.12	.36	-.01	.05	.05	.05	.07	.11
5	.00	.00	.00	.00	.15	-.05	-.02	.03	.04	.04	.05	-.04	.19	.69	-.10	.04	.03	.04	.05	.08
6	.00	.00	.00	.00	.00	.04	-.06	.01	.02	.02	.03	.13	.00	.00	-.59	.02	.01	.02	.03	.05
7	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	-.00	.00	.00	.00	.00	.00	.00

	22	23	24	25
1	.00	.00	.00	.00
2	.15	.20	.10	.00
3	.14	.09	-.04	.00
4	.12	.07	-.03	.00
5	.09	.05	-.02	.00
6	.05	.03	-.01	.00
7	.00	.00	.00	.00

U VELOCITY FOR J = 14 AT TIMESTEP 20

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	.00	-.19	-.46	-.63	-.78	-.81	-.79	-.77	-.74	-.73	-.73	-.73	-.69	-.67	-.67	-.71	-.71	-.71	-.71	-.68
2	.00	.14	.33	.23	.08	.05	.07	.09	.12	.13	.13	.13	.17	.19	.19	.15	.15	.15	.14	.15
3	.00	.00	.00	.01	.02	.01	.03	.05	.08	.09	.09	.09	.13	.15	.15	.11	.12	.11	.11	.10
4	.00	.00	.00	.00	-.04	.00	.03	.05	.08	.09	.09	.09	.13	.15	.15	.11	.12	.11	.11	.10
5	.00	.00	.00	.00	.00	-.03	.02	.05	.08	.09	.09	.09	.13	.15	.15	.11	.11	.11	.11	.10
6	.00	.00	.00	.00	.00	.00	.01	.05	.08	.09	.09	.08	.00	.00	.00	.11	.11	.11	.11	.10

	21	22	23	24	25
1	-.49	.00	.00	.00	.00
2	.19	.12	.00	.00	.00
3	.09	.04	.00	.00	.00
4	.09	.04	.00	.00	.00
5	.09	.04	.00	.00	.00
6	.09	.04	.00	.00	.00

V VELOCITY FOR J = 14 AT TIMESTEP 20

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	.00	1.76	1.25	.09	-.42	-.68	-.79	-.84	-.87	-.89	-.88	-.88	-.95	-1.01	-1.15	-1.06	-1.05	-1.04	-1.03	-.99
2	.00	2.60	2.16	1.02	.52	.25	.14	.10	.06	.05	.05	.05	-.02	-.07	-.21	-.13	-.11	-.11	-.10	-.08
3	.00	.00	.00	.97	.49	.23	.13	.08	.05	.04	.04	.05	-.03	-.08	-.23	-.14	-.13	-.12	-.12	-.11
4	.00	.00	.00	.00	.49	.23	.13	.08	.05	.04	.04	.05	-.03	-.08	-.23	-.14	-.13	-.12	-.12	-.11
5	.00	.00	.00	.00	.00	.23	.13	.08	.05	.04	.04	.05	-.03	-.08	-.23	-.14	-.13	-.12	-.12	-.11
6	.00	.00	.00	.00	.00	.00	.12	.08	.05	.04	.04	.05	.00	.00	.00	-.13	-.13	-.12	-.12	-.11

	21	22	23	24	25
1	-.95	-.96	.00	.00	.00
2	-.07	-.03	.00	.00	.00
3	-.11	-.04	.00	.00	.00
4	-.11	-.04	.00	.00	.00
5	-.11	-.04	.00	.00	.00
6	-.11	-.04	.00	.00	.00

TS= 20 YEAR= .11 DAY= 40.0 ENERGY= 1.806804E-01 DTEMP= 6.787344E-09 DSALT= 2.631772E-13 SCANS= 6

WORK BY:	INTERNAL MODE	EXTERNAL MODE
TIME RATE OF CHANGE	1.894179E-06	7.113672E-07
HORIZONTAL ADVECTION	-2.968689E-10	-2.638344E-10
VERTICAL ADVECTION	2.148607E-10	3.458426E-10
HORIZONTAL FRICTION	-3.254516E-07	-2.244480E-07
VERTICAL FRICTION	-1.054360E-09	-1.746040E-24
PRESSURE FORCES	2.054136E-06	6.196989E-07
IMPLICIT EFFECTS	-9.024309E-08	2.190050E-08
WORK BY WIND	7.569083E-07	2.941338E-07
BOTTOM DRAG	-3.409384E-11	.0
WORK BY BUOYANCY FORCES	2.673835E-06	

	TEMPERATURE	SALINITY
TIME RATE OF CHANGE	2.083670E-10	1.616293E-13
HORIZONTAL ADVECTION	-1.212414E-20	-4.391313E-25
VERTICAL ADVECTION	-3.317415E-22	-7.685935E-27
HORIZONTAL DIFFUSION	-9.661547E-22	-6.564295E-26
SURFACE FLUX	2.083670E-10	1.616293E-13
TRUNCATION ERROR	-2.370766E-19	6.316946E-25
CHANGE OF VARIANCE	4.023143E-09	-2.722894E-17

ENERGY CONVERSION ERROR -2.147201E-15 NONLINEAR EXCHANGE ERROR -1.242425E-20

NORTHWARD TRANSPORT OF HEAT (X10**15 WATTS)

	X MEAN	X EDDY	Z MEAN	Z EDDY	EKMAN	TOT ADV	DIFFUS	TOTAL
18	-.002	-.001	-.003	-.005	.004	-.003	.059	.056
17	.000	-.000	.000	.002	-.002	.000	.011	.011
16	.001	-.000	.002	.017	-.018	.001	.009	.010
15	-.006	-.000	.005	.025	-.035	-.006	.010	.004
14	-.021	-.000	.007	.021	-.049	-.021	.011	-.010
13	-.041	-.000	.009	.007	-.057	-.041	.012	-.029
12	-.052	-.000	.012	-.012	-.052	-.052	.013	-.040
11	-.034	-.000	.015	-.017	-.031	-.034	.011	-.023
10	.009	-.000	.018	-.009	-.000	.009	.012	.021
9	.100	-.003	.021	.030	.048	.099	.012	.112
8	.273	-.000	.025	.123	.126	.273	.013	.286
7	.474	-.000	.029	.225	.220	.474	.012	.486
6	.626	-.001	.033	.271	.321	.625	.012	.637
5	.667	-.001	.039	.197	.430	.666	.012	.678
4	.559	-.001	.047	-.032	.543	.557	.011	.568
3	.364	-.002	.046	-.338	.654	.362	.006	.369
2	.289	.000	.019	-.490	.760	.289	-.033	.256

NORTHWARD TRANSPORT OF SALT (X10**10 CM**3/SEC)

	X MEAN	X EDDY	Z MEAN	Z EDDY	EKMAN	TOT ADV	DIFFUS	TOTAL
18	.003	.002	.002	-.012	.015	.005	-.092	-.087
17	-.008	.000	-.007	-.065	.065	-.008	-.033	-.041
16	-.003	.000	-.014	-.106	.118	-.003	-.023	-.026
15	.025	.000	-.019	-.105	.149	.025	-.012	.013
14	.063	.000	-.021	-.065	.149	.063	.003	.067
13	.082	.000	-.018	-.014	.115	.083	.020	.103
12	.060	.000	-.013	.013	.060	.060	.036	.096
11	.013	.000	-.006	.007	.012	.013	.044	.057
10	.002	-.000	.004	-.002	-.000	.002	.049	.051
9	.069	-.000	.015	.021	.033	.069	.047	.116
8	.274	-.000	.025	.123	.126	.273	.041	.314
7	.544	-.000	.033	.258	.253	.544	.024	.567
6	.716	-.001	.037	.310	.368	.716	.004	.720
5	.680	-.001	.039	.201	.439	.679	-.015	.664
4	.448	-.001	.037	-.026	.436	.447	-.031	.416
3	.193	-.001	.025	-.179	.347	.193	-.043	.149
2	.071	.000	.005	-.118	.184	.071	-.050	.021

MERIDIONAL MASS TRANSPORT

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	.00	3.07	4.09	6.86	8.91	9.11	7.59	4.84	1.95	.17	-.93	-1.68	-1.63	-1.10	-.46	.06	.30	-.05	.00	.00
2	.00	8.88	9.24	9.42	10.08	9.75	8.31	5.91	3.60	2.09	1.15	.54	.52	.90	1.38	1.69	1.56	1.57	.00	.00
3	.00	12.08	9.38	8.71	9.19	8.86	7.72	5.70	3.89	2.58	1.75	1.24	1.23	1.69	2.31	2.75	2.63	3.21	.00	.00
4	.00	2.55	6.52	6.39	6.99	6.79	6.07	4.52	3.39	2.32	1.59	1.21	1.15	1.65	2.34	2.79	2.74	3.15	.00	.00
5	.00	-4.84	2.93	3.27	3.79	3.76	3.47	2.67	2.09	1.52	1.13	.91	.86	.78	1.45	1.70	1.71	2.22	.00	.00
6	.00	-.00	.00	.00	.00	.00	.00	-.00	-.00	-.00	-.00	-.00	.00	.00	.00	.00	.00	.00	.00	.00

STREAM FUNCTION IN SVERDRUPS, TS= 20

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
20	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00
19	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00
18	.00	.51	1.15	3.39	6.02	7.41	4.56	3.06	2.33	1.96	1.73	1.56	1.39	1.19	.96	.69	.37	.00	.00	.00
17	.00	.77	1.51	3.51	5.77	6.72	5.81	4.82	4.21	3.83	3.56	3.32	3.03	2.67	2.22	1.71	1.14	.59	.00	.00
16	.00	.93	1.76	3.42	5.18	6.09	6.03	5.79	5.56	5.40	5.28	5.12	4.83	4.34	3.67	2.91	2.15	1.40	.00	.00
15	.00	1.05	1.94	3.34	4.79	5.74	6.22	6.45	6.55	6.62	6.75	7.00	6.74	6.25	5.21	4.12	3.20	2.33	.70	.00
14	.00	1.14	2.06	3.29	4.56	5.58	6.41	6.92	7.17	7.28	7.38	7.55	7.43	7.12	5.98	5.06	4.11	3.19	1.49	.74
13	.00	1.21	2.13	3.26	4.45	5.55	6.58	7.21	7.47	7.44	7.20	7.15	7.21	6.97	6.44	5.93	4.83	3.88	2.31	1.45
12	.00	1.25	2.17	3.26	4.43	5.59	6.71	7.37	7.57	7.42	7.01	7.01	7.01	7.01	7.01	5.92	5.17	4.32	2.97	2.09
11	.00	1.29	2.20	3.27	4.46	5.66	6.80	7.42	7.55	7.30	6.79	7.01	7.01	7.01	7.01	6.17	5.43	4.58	3.62	2.51
10	.00	1.32	2.24	3.30	4.51	5.75	6.84	7.38	7.43	7.10	6.59	7.01	7.01	7.01	7.01	6.39	5.63	4.69	3.67	2.73
9	.00	1.35	2.28	3.35	4.59	5.82	6.80	7.24	7.22	6.83	6.25	7.01	7.01	7.01	7.01	6.81	5.66	4.57	3.73	2.77
8	.00	1.40	2.34	3.42	4.69	5.88	6.67	6.97	6.90	6.53	6.02	6.23	6.82	7.02	6.87	6.41	5.27	4.16	3.56	2.60
7	.00	1.47	2.45	3.55	4.82	5.91	6.42	6.53	6.42	6.14	5.86	5.69	6.59	6.84	7.04	5.75	4.57	3.48	3.14	2.19
6	.00	1.59	2.65	3.82	5.09	5.96	5.99	5.85	5.66	5.46	5.27	5.21	5.64	5.83	5.84	4.70	3.60	2.60	2.48	1.55
5	.00	1.77	3.05	4.37	5.49	5.89	5.24	4.82	4.58	4.41	4.30	4.28	4.35	4.32	4.02	3.33	2.49	1.61	1.65	.81
																			.80	.00

4	.00	1.90	3.63	4.96	5.34	4.95	3.82	3.38	3.16	3.04	2.97	2.93	2.90	2.78	2.50	2.02	1.40	.73	.00	.00
3	.00	1.42	3.44	3.38	2.42	1.30	1.66	1.60	1.52	1.47	1.44	1.41	1.38	1.29	1.13	.88	.52	.00	.00	.00
2	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00
1	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.30	.00

	22	23	24
20	.00	.00	.00
19	.00	.00	.00
18	.00	.00	.00
17	.00	.00	.00
16	.00	.00	.00
15	.00	.00	.00
14	.71	.00	.00
13	1.25	.55	.00
12	1.62	.74	.00
11	1.78	.83	.00
10	1.80	.84	.00
9	1.67	.77	.00
8	1.31	.56	.00
7	.75	.00	.00
6	.00	.00	.00
5	.00	.00	.00
4	.00	.00	.00
3	.00	.00	.00
2	.00	.00	.00
1	.00	.00	.00

```
TS=    21  YEAR=   .11  DAY= 42.0  ENERGY= 2.065615E-01  DTEMP= 9.264710E-09  DSALT= 3.859141E-13  SCANS=  6
TS=    22  YEAR=   .12  DAY= 44.0  ENERGY= 2.313324E-01  DTEMP= 8.223412E-09  DSALT= 3.150436E-13  SCANS=  7
TS=    23  YEAR=   .13  DAY= 46.0  ENERGY= 2.571433E-01  DTEMP= 9.269025E-09  DSALT= 3.466718E-13  SCANS= 10
TS=    24  YEAR=   .13  DAY= 48.0  ENERGY= 2.702575E-01  DTEMP= 1.030431E-08  DSALT= 3.752311E-13  SCANS= 10
TS=    25  YEAR=   .14  DAY= 50.0  ENERGY= 2.886908E-01  DTEMP= 1.093742E-08  DSALT= 3.830820E-13  SCANS=  8
TS=    26  YEAR=   .14  DAY= 52.0  ENERGY= 2.809529E-01  DTEMP= 1.118391E-08  DSALT= 3.742770E-13  SCANS=  8
TS=    27  YEAR=   .15  DAY= 54.0  ENERGY= 2.844649E-01  DTEMP= 1.110237E-08  DSALT= 3.555508E-13  SCANS=  8
TS=    28  YEAR=   .15  DAY= 56.0  ENERGY= 2.712065E-01  DTEMP= 1.089004E-08  DSALT= 3.272171E-13  SCANS=  7
TS=    29  YEAR=   .16  DAY= 58.0  ENERGY= 2.714638E-01  DTEMP= 1.029730E-08  DSALT= 2.934911E-13  SCANS=  7
TS=    30  YEAR=   .16  DAY= 60.0  ENERGY= 2.739380E-01  DTEMP= 9.690161E-09  DSALT= 2.702316E-13  SCANS=  6
TS=    31  YEAR=   .17  DAY= 62.0  ENERGY= 2.917691E-01  DTEMP= 1.414463E-08  DSALT= 3.720253E-13  SCANS=  6
TS=    32  YEAR=   .18  DAY= 64.0  ENERGY= 3.173365E-01  DTEMP= 9.298759E-09  DSALT= 2.720715E-13  SCANS=  7
TS=    33  YEAR=   .18  DAY= 66.0  ENERGY= 3.570783E-01  DTEMP= 8.607250E-09  DSALT= 2.894679E-13  SCANS= 10
TS=    34  YEAR=   .19  DAY= 68.0  ENERGY= 3.845497E-01  DTEMP= 8.964661E-09  DSALT= 3.067476E-13  SCANS=  9
TS=    35  YEAR=   .19  DAY= 70.0  ENERGY= 4.352470E-01  DTEMP= 8.438829E-09  DSALT= 3.150542E-13  SCANS=  8
TS=    36  YEAR=   .20  DAY= 72.0  ENERGY= 4.515699E-01  DTEMP= 8.465327E-09  DSALT= 3.225132E-13  SCANS=  8
TS=    37  YEAR=   .20  DAY= 74.0  ENERGY= 4.933211E-01  DTEMP= 8.056424E-09  DSALT= 3.169483E-13  SCANS=  8
TS=    38  YEAR=   .21  DAY= 76.0  ENERGY= 5.007238E-01  DTEMP= 8.271957E-09  DSALT= 3.177156E-13  SCANS=  7
TS=    39  YEAR=   .21  DAY= 78.0  ENERGY= 5.249211E-01  DTEMP= 8.721262E-09  DSALT= 3.194789E-13  SCANS=  7
```

TEMPERATURE FOR J = 7 AT TIMESTEP 40

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65	19.65
2	4.07	4.64	4.78	4.83	4.83	4.82	4.82	4.82	4.82	4.82	4.82	4.84	4.81	4.82	4.80	4.81	4.82	4.82	4.82	4.80
3	.00	.00	4.00	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.00	4.01	4.00	4.01	4.01	4.01	4.01	4.00
4	.00	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
5	.00	.00	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
6	.00	.00	.00	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	.00	.00	4.00	4.00	4.00	4.00	4.00	4.00

	22	23	24	25
1	19.65	19.65	19.65	.00
2	4.79	4.58	4.29	.00
3	4.00	4.00	4.00	.00
4	4.00	4.00	4.00	.00
5	4.00	4.00	4.00	.00
6	4.00	4.00	4.00	.00

SALINITY FOR J = 7 AT TIMESTEP 40

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67	.67
2	-.10	-.07	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06	-.06
3	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10
4	10.00	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10
5	10.00	10.00	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10	-.10
6	10.00	10.00	10.00	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10	-.10	10.00	10.00	-.10	-.10	-.10	-.10	-.10	-.10

22 23 24 25

The two integral constraints addressed in Section D may be seen to hold for Model 1 from the energy integral printout on p. L11. The constraint of Eq. 31, proven by Eqs. 61-76, is verified by the "NON-LINEAR EXCHANGE ERROR" which is some 10 orders of magnitude less than the "HORIZONTAL" and "VERTICAL ADVECTION" terms individually. The former is simply the summation of the latter 4 terms. The constraint of Eq. 34, proven by Eqs. 77-83, is verified by the "ENERGY CONVERSION ERROR" which is 9 orders of magnitude smaller than the "WORK BY BUOYANCY FORCES", indicating a balance of the latter with the 2 "PRESSURE FORCES" terms. These balances should be checked carefully upon setting up a new model or changing an existing one. The error terms will generally be at least three orders of magnitude less than the constituent terms, depending on the precision of the computer used.

Model 2 is a 15 level, rectangular basin with a shelf along the western wall, extending from 54° S to the equator, where symmetry conditions are imposed (by means of the "S" option). It also includes a telescoping grid in the zonal direction, increasing from 2° spacing in the west to 3° spacing at the eastern wall. Bottom drag has been included in the updates as well as a Newtonian surface boundary condition for T and S. The NAMELIST input file is similar in structure to that of Model 1.

UPDATE FOR MODEL 2:

```
OPT=K,S,C
      PARAMETER (INT=025,JNT=020,KM=15,NT=2,LSEG=1,NISLE=0,LBC=2                000200000
      SWLDEG=-54.0                                                              111100000
        DYT(J)=3.0                                                              111800000
        DXT(I)=2.0                                                              112900000
        IF(I.GT.8)DXT(I)=2.5-0.5*COS(PI*(I-8)/16.)                              112900010
      DZ(1)=30.00E2                                                             114400000
      DZ(2)=46.30E2                                                             114400010
      DZ(3)=67.45E2                                                             114400020
      DZ(4)=94.46E2                                                             114400030
      DZ(5)=128.46E2                                                            114400040
      DZ(6)=170.57E2                                                            114400050
      DZ(7)=221.91E2                                                            114400060
      DZ(8)=283.51E2                                                            114400070
      DZ(9)=356.21E2                                                            114400080
      DZ(10)=440.57E2                                                           114400090
      DZ(11)=536.78E2                                                           114400100
      DZ(12)=644.54E2                                                           114400110
      DZ(13)=763.00E2                                                           114400120
      DZ(14)=890.71E2                                                           114400130
      DZ(15)=1025.53E2                                                          114400140
      DO 700 J=2,JMT                                                            133200000
        FKMP(I,J)=KM                                                            133400000
      DO 702 J=2,JMT                                                            133500010
      DO 702 I=2,7                                                              133500020
        FKMP(I,J)=2*I                                                           133500030
 702  CONTINUE                                                                  133500040
        FX=(-SIN(PI*FLOAT(J-2)/FLOAT(JMT-4)))*2.                                151900010
          WSX(I)=FX                                                             152400000
          WSY(I)=0.0                                                            152500000
      MTEST=MOD(J,7)                                                            249100000
      IPRT=20                                                                   249400000
C  ASSUME 10 DEGREE TURNING ANGLE AT BOTTOM BOUNDARY                            328900010
C                                                                               328900020
      CD=1.3E-3                                                                 328900030
      FXA=CD/FKPM                                                               328900040
        FXB=FXA*SQRT(UB(I,KZ)**2+VB(I,KZ)**2)*DZZ(KZ+1)                         329100010
        UDIF(I,KZ+1)=UB(I,KZ)-FXB*(UB(I,KZ)*.98481-VB(I,KZ)*.17365)             329200000
        VDIF(I,KZ+1)=VB(I,KZ)-FXB*(UB(I,KZ)*.17365+VB(I,KZ)*.98481)             329300000
C                                                                               420100010
C-----------------------------------------------------------------------        420100020
C  SET NEWTONIAN SURFACE BOUNDARY CONDITION                                     420100030
C-----------------------------------------------------------------------        420100040
C                                                                               420100050
      FXA=25.0                                                                  420100060
      FXB=20.0                                                                  420100070
      IF(M.EQ.2) THEN                                                           420100080
        FXA=0.0                                                                 420100090
        FXB=-.001                                                               420100100
      ENDIF                                                                     420100110
      DO 846 I=2,IMTM1                                                          420100120
        TA(I,1,M)=TA(I,1,M)+2.315E-7*(FXA+FXB*PHIT(J)-TB(I,1,M))                420100130
 846  CONTINUE                                                                  420100140
C                                                                               427200010
C  ADD IN CONTRIBUTION FROM NEWTONIAN BOUNDARY CONDITION                        427200020
C                                                                               427200030
          IF(M.EQ.1) TTDTOT(5,1)=TTDTOT(5,1)                                    427200040
     *                        +2.315E-7*(25.0+20.0*PHIT(J)-TB(I,1,1))           427200050
          IF(M.EQ.2) TTDTOT(5,2)=TTDTOT(5,2)                                    427200060
     *                        +2.315E-7*(     -.001*PHIT(J)-TB(I,1,2))          427200070
      DATA TO/                                                                  603800000
     *          13.4979166,13.4926065,13.4846595,13.4732852, 8.4679890,         603800010
     *           8.4517059, 5.9421330, 4.4281250, 3.9040599, 2.8802061,         603800020
     *           2.8360754, 2.7780657, 2.7025290, 2.6052822, 2.9625234/         603800030
      DATA SO/                                                                  603900000
     *          -.0022500, -.0022500, -.0022500, -.0022500,  .0001500,          603900010
     *           .0001500, -.0001000, -.0001000, -.0002500, -.0002000,          603900020
     *          -.0002000, -.0002000, -.0002000, -.0002000, -.0002000/          603900030
      DATA (C( 1,N),N=1,9)/                                                     604000000
     *          -.2019152E-03, .7709387E+00,-.4915766E-05,-.2007672E-02,        604000010
     *           .4496149E+00, .3652747E-07, .4725372E-02, .3768380E-04,        604000020
     *           .6548196E+01/                                                  604000030
      DATA (C( 2,N),N=1,9)/                                                     604000040
     *          -.2026279E-03, .7706838E+00,-.4903913E-05,-.2004045E-02,        604000050
     *           .4497566E+00, .3639690E-07, .4712010E-02, .3761764E-04,        604000060
     *           .6546772E+01/                                                  604000070
      DATA (C( 3,N),N=1,9)/                                                     604000080
     *          -.2036885E-03, .7703042E+00,-.4886267E-05,-.1998629E-02,        604000090
     *           .4499598E+00, .3620095E-07, .4692196E-02, .3751913E-04,        604000100
     *           .6544648E+01/                                                  604000110
      DATA (C( 4,N),N=1,9)/                                                     604000120
     *          -.2051946E-03, .7697646E+00,-.4861205E-05,-.1990901E-02,        604000130
```

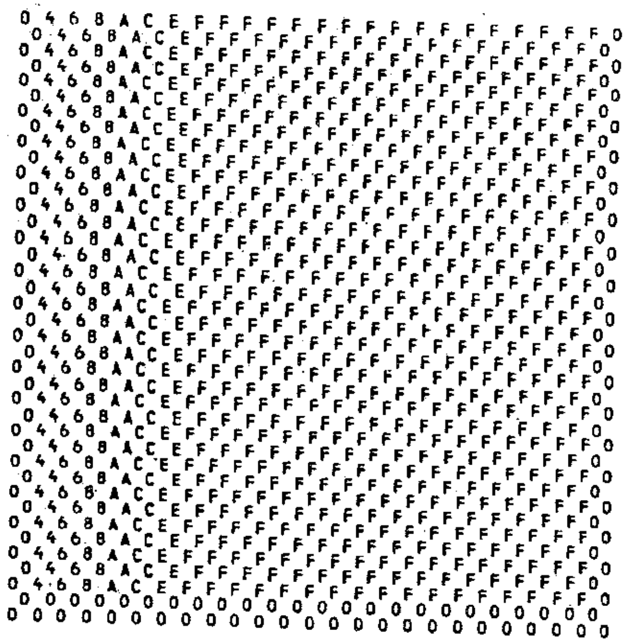
```
     *           .4502325E+00, .3591934E-07, .4664216E-02, .3737920E-04,
     *           .6541624E+01/
      DATA (C( 5,N),N=1,9)/
     *          -.1632162E-03, .7810057E+00,-.5277472E-05,-.2322824E-02,
     *           .7159905E+00, .4943163E-07, .5148399E-02, .3859627E-04,
     *           .6612560E+01/
      DATA (C( 6,N),N=1,9)/
     *          -.1664980E-03, .7799287E+00,-.5223968E-05,-.2305658E-02,
     *           .6964292E+00, .4903556E-07, .5108021E-02, .3838043E-04,
     *           .6607319E+01/
      DATA (C( 7,N),N=1,9)/
     *          -.1440378E-03, .7842682E+00,-.5519369E-05,-.2473774E-02,
     *           .5910575E+00, .5793752E-07, .6154590E-02, .3926047E-04,
     *           .6638471E+01/
      DATA (C( 8,N),N=1,9)/
     *          -.1336783E-03, .7860867E+00,-.5672235E-05,-.2557069E-02,
     *           .3528398E+00, .6399606E-07, .7004312E-02, .3962390E-04,
     *           .6655343E+01/
      DATA (C( 9,N),N=1,9)/
     *          -.1357195E-03, .7848000E+00,-.5628947E-05,-.2554850E-02,
     *           .1057585E+00, .6555468E-07, .7215849E-02, .3964243E-04,
     *           .6652038E+01/
      DATA (C(10,N),N=1,9)/
     *          -.1348395E-03, .7842621E+00,-.5657959E-05,-.2578544E-02,
     *          -.6064741E+00, .7002210E-07, .7931478E-02, .3974567E-04,
     *           .6657217E+01/
      DATA (C(11,N),N=1,9)/
     *          -.1472340E-03, .7804559E+00,-.5461105E-05,-.2512906E-02,
     *          -.2829495E+01, .6901463E-07, .7878454E-02, .3931736E-04,
     *           .6640514E+01/
      DATA (C(12,N),N=1,9)/
     *          -.1619146E-03, .7759107E+00,-.5224071E-05,-.2433677E-02,
     *          -.4372853E+01, .6757406E-07, .7827411E-02, .3881627E-04,
     *           .6620347E+01/
      DATA (C(13,N),N=1,9)/
     *          -.1789724E-03, .7705813E+00,-.4943833E-05,-.2339711E-02,
     *          -.6454825E+01, .6554043E-07, .7786706E-02, .3824815E-04,
     *           .6596461E+01/
      DATA (C(14,N),N=1,9)/
     *          -.1984058E-03, .7644511E+00,-.4619129E-05,-.2230396E-02,
     *          -.9215148E+01, .6271601E-07, .7769342E-02, .3762998E-04,
     *           .6568810E+01/
      DATA (C(15,N),N=1,9)/
     *          -.2240840E-03, .7565406E+00,-.4162414E-05,-.2069851E-02,
     *          -.9645579E+01, .5543124E-07, .7290867E-02, .3652558E-04,
     *           .6526380E+01/
```

DATA FOR MODEL 2:

```
&CONTRL NFIRST=1,NLAST=100,NNERGY=50,NTSI=01,NMIX=20,&END
&EDDY AMF=1.E9,AHF=2.E7,FKPMF=1.0,FKPHF=1.0,&END
&TSTEPS DTTSF=172800.,DTUVF=7200.,DTSFF=7200.,&END
&PARMS MXSCAN=100,SORF=1.60,CRITF=1.E8,ACORF=.6,&END
&TSPROF TINITF=4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,.0349,.0349,.0349,
.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,&END
```

```
&CONTRL
NFIRST   =  1,
NLAST    =  100,
NNERGY   =  50,
NMIX     =  20,
NWRITE   =  1000000,
NOW      =  1000000,
NTSI     =  1,
NA       =  0,
NB       =  0,
NC       =  0,
&END

&EDDY
AMF      =  .1E+10,
AHF      =  .2E+08,
FKPMF    =  .1E+01,
FKPHF    =  .1E+01,
&END

&TSTEPS
DTTSF    =  .1728E+06,
DTUVF    =  .72E+04,
DTSFF    =  .72E+04,
&END

&PARMS
ACORF    =  .6E+00,
MXSCAN   =  100,
SORF     =  .16E+01,
CRITF    =  .1E+09,
&END

GRID BOX THICKNESS  'DZ'
  .30000E+04   .46300E+04   .67450E+04   .94460E+04   .12846E+05   .17057E+05   .22191E+05   .28351E+05   .35621E+05   .44057E+05
  .53678E+05   .64454E+05   .76300E+05   .89071E+05   .10255E+06

GRID POINT SEPARATION  'DZZ'
  .15000E+04   .38150E+04   .56875E+04   .80955E+04   .11146E+05   .14952E+05   .19624E+05   .25271E+05   .31986E+05   .39839E+05
  .48868E+05   .59066E+05   .70377E+05   .82686E+05   .95812E+05   .51277E+05

DEPTH OF BOX BOTTOM  'ZDZ'
  .30000E+04   .76300E+04   .14375E+05   .23821E+05   .36667E+05   .53724E+05   .75915E+05   .10427E+06   .13989E+06   .18394E+06
  .23762E+06   .30208E+06   .37838E+06   .46745E+06   .57000E+06

DEPTH OF GRID POINT  'ZDZZ'
  .15000E+04   .53150E+04   .11003E+05   .19098E+05   .30244E+05   .45196E+05   .64820E+05   .90091E+05   .12208E+06   .16192E+06
  .21078E+06   .26985E+06   .34023E+06   .42291E+06   .51872E+06   .57000E+06

LATITUDE OF T,S POINTS (RADIANS)  'PHIT'
 -.96866E+00  -.91630E+00  -.86394E+00  -.81158E+00  -.75922E+00  -.70686E+00  -.65450E+00  -.60214E+00  -.54978E+00  -.49742E+00
 -.44506E+00  -.39270E+00  -.34034E+00  -.28798E+00  -.23562E+00  -.18326E+00  -.13090E+00  -.78540E-01  -.26180E-01   .26180E-01

LATITUDE OF U,V POINTS (RADIANS)  'PHI'
 -.94248E+00  -.89012E+00  -.83776E+00  -.78540E+00  -.73304E+00  -.68068E+00  -.62832E+00  -.57596E+00  -.52360E+00  -.47124E+00
 -.41888E+00  -.36652E+00  -.31416E+00  -.26180E+00  -.20944E+00  -.15708E+00  -.10472E+00  -.52360E-01  -.74163E-13   .52360E-01

COSINE OF T,S LATITUDE  'CST'
  .56641E+00   .60876E+00   .64945E+00   .68835E+00   .72537E+00   .76041E+00   .79335E+00   .82413E+00   .85264E+00   .87882E+00
  .90259E+00   .92388E+00   .94264E+00   .95882E+00   .97237E+00   .98325E+00   .99144E+00   .99692E+00   .99966E+00   .99966E+00

COSINE OF U,V LATITUDE  'CS'
  .58779E+00   .62932E+00   .66913E+00   .707[illegible]+00   .74314E+00   .77715E+00   .80902E+00   .83867E+00   .86603E+00   .89101E+00
  .91355E+00   .93358E+00   .95106E+00   .96593E+00   .97815E+00   .98769E+00   .99452E+00   .99863E+00   .10000E+01   .99863E+00

SINE OF U,V LATITUDE  'SINE'
 -.80902E+00  -.77715E+00  -.74314E+00  -.70711E+00  -.66913E+00  -.62932E+00  -.58779E+00  -.54464E+00  -.50000E+00  -.45399E+00
 -.40674E+00  -.35837E+00  -.30902E+00  -.25882E+00  -.20791E+00  -.15643E+00  -.10453E+00  -.52336E-01  -.74163E-13   .52336E-01
```

NUMBER OF LEVELS AT T,S POINTS AND U,V POINTS



SURFACE AREA = 3.122914E+17 VOLUME = 1.518226E+23

```
START AND END INDICES FOR STREAM FUNCTION
J= 20         0    0
J= 19         3   23
J= 18         3   23
J= 17         3   23
J= 16         3   23
J= 15         3   23
J= 14         3   23
J= 13         3   23
J= 12         3   23
J= 11         3   23
J= 10         3   23
J=  9         3   23
J=  8         3   23
J=  7         3   23
J=  6         3   23
J=  5         3   23
J=  4         3   23
J=  3         3   23
J=  2         0    0
J=  1         0    0

&TSPROF
TIMITF = 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000
4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000
0.0349000000000 0.0349000000000 0.0349000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 0.0349000000000
0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000
&END

TS=   1  YEAR=  0.01  DAY=  2.0  ENERGY= 2.833284E-02  DTEMP= 1.764721E-08  DSALT= 9.981131E-09  SCANS= 47
TS=   2  YEAR=  0.01  DAY=  4.0  ENERGY= 2.833289E-02  DTEMP= 1.764708E-08  DSALT= 4.990947E-09  SCANS= 47
TS=   3  YEAR=  0.02  DAY=  6.0  ENERGY= 9.371128E-02  DTEMP= 1.697233E-08  DSALT= 7.376915E-13  SCANS= 47
TS=   4  YEAR=  0.02  DAY=  8.0  ENERGY= 1.963488E-01  DTEMP= 1.631215E-08  DSALT= 7.133956E-13  SCANS= 21
TS=   5  YEAR=  0.03  DAY= 10.0  ENERGY= 2.931513E-01  DTEMP= 1.572470E-08  DSALT= 6.925455E-13  SCANS=  8
TS=   6  YEAR=  0.03  DAY= 12.0  ENERGY= 3.968973E-01  DTEMP= 1.513295E-08  DSALT= 6.716056E-13  SCANS= 10
TS=   7  YEAR=  0.04  DAY= 14.0  ENERGY= 4.647560E-01  DTEMP= 1.460235E-08  DSALT= 6.533705E-13  SCANS= 13
TS=   8  YEAR=  0.04  DAY= 16.0  ENERGY= 5.182046E-01  DTEMP= 1.406741E-08  DSALT= 6.351733E-13  SCANS= 13
TS=   9  YEAR=  0.05  DAY= 18.0  ENERGY= 5.341489E-01  DTEMP= 1.359563E-08  DSALT= 6.193613E-13  SCANS= 13
TS=  10  YEAR=  0.05  DAY= 20.0  ENERGY= 5.335322E-01  DTEMP= 1.317504E-08  DSALT= 6.043908E-13  SCANS= 13
TS=  11  YEAR=  0.06  DAY= 22.0  ENERGY= 5.126568E-01  DTEMP= 1.280925E-08  DSALT= 5.913009E-13  SCANS= 12
TS=  12  YEAR=  0.07  DAY= 24.0  ENERGY= 4.857609E-01  DTEMP= 1.243556E-08  DSALT= 5.780998E-13  SCANS= 10
TS=  13  YEAR=  0.07  DAY= 26.0  ENERGY= 4.558875E-01  DTEMP= 1.213059E-08  DSALT= 5.664715E-13  SCANS=  7
TS=  14  YEAR=  0.08  DAY= 28.0  ENERGY= 4.306144E-01  DTEMP= 1.220638E-08  DSALT= 6.124702E-13  SCANS=  7
TS=  15  YEAR=  0.08  DAY= 30.0  ENERGY= 4.090988E-01  DTEMP= 1.195577E-08  DSALT= 6.029742E-13  SCANS= 10
TS=  16  YEAR=  0.09  DAY= 32.0  ENERGY= 3.958287E-01  DTEMP= 1.141680E-08  DSALT= 5.540379E-13  SCANS=  9
TS=  17  YEAR=  0.09  DAY= 34.0  ENERGY= 3.843840E-01  DTEMP= 1.109962E-08  DSALT= 5.355836E-13  SCANS= 11
TS=  18  YEAR=  0.10  DAY= 36.0  ENERGY= 3.791413E-01  DTEMP= 1.076831E-08  DSALT= 5.173540E-13  SCANS=  8
TS=  19  YEAR=  0.10  DAY= 38.0  ENERGY= 3.721534E-01  DTEMP= 1.048547E-08  DSALT= 5.037062E-13  SCANS= 10
TS=  20  YEAR=  0.11  DAY= 40.0  ENERGY= 3.683395E-01  DTEMP= 1.014964E-08  DSALT= 4.885403E-13  SCANS=  7
TS=  21  YEAR=  0.11  DAY= 42.0  ENERGY= 3.571931E-01  DTEMP= 1.057577E-08  DSALT= 6.511304E-13  SCANS= 17
TS=  22  YEAR=  0.12  DAY= 44.0  ENERGY= 3.570906E-01  DTEMP= 9.704359E-09  DSALT= 4.719267E-13  SCANS= 18
TS=  23  YEAR=  0.13  DAY= 46.0  ENERGY= 3.506689E-01  DTEMP= 9.531531E-09  DSALT= 4.676346E-13  SCANS= 21
TS=  24  YEAR=  0.13  DAY= 48.0  ENERGY= 3.464155E-01  DTEMP= 9.334565E-09  DSALT= 4.583062E-13  SCANS= 17
TS=  25  YEAR=  0.14  DAY= 50.0  ENERGY= 3.414828E-01  DTEMP= 9.204406E-09  DSALT= 4.537062E-13  SCANS= 11
TS=  26  YEAR=  0.14  DAY= 52.0  ENERGY= 3.389993E-01  DTEMP= 9.049442E-09  DSALT= 4.459394E-13  SCANS=  9
TS=  27  YEAR=  0.15  DAY= 54.0  ENERGY= 3.360436E-01  DTEMP= 8.891940E-09  DSALT= 4.414585E-13  SCANS= 10
TS=  28  YEAR=  0.15  DAY= 56.0  ENERGY= 3.353010E-01  DTEMP= 8.698652E-09  DSALT= 4.360309E-13  SCANS=  8
TS=  29  YEAR=  0.16  DAY= 58.0  ENERGY= 3.338595E-01  DTEMP= 8.609062E-09  DSALT= 4.323669E-13  SCANS=  9
TS=  30  YEAR=  0.16  DAY= 60.0  ENERGY= 3.341896E-01  DTEMP= 8.579241E-09  DSALT= 4.280256E-13  SCANS=  7
TS=  31  YEAR=  0.17  DAY= 62.0  ENERGY= 3.337779E-01  DTEMP= 8.511666E-09  DSALT= 4.252794E-13  SCANS=  9
TS=  32  YEAR=  0.18  DAY= 64.0  ENERGY= 3.348775E-01  DTEMP= 8.534793E-09  DSALT= 4.242310E-13  SCANS=  7
TS=  33  YEAR=  0.18  DAY= 66.0  ENERGY= 3.354300E-01  DTEMP= 8.408841E-09  DSALT= 4.221548E-13  SCANS=  9
TS=  34  YEAR=  0.19  DAY= 68.0  ENERGY= 3.374172E-01  DTEMP= 8.214896E-09  DSALT= 4.183493E-13  SCANS=  7
TS=  35  YEAR=  0.19  DAY= 70.0  ENERGY= 3.390089E-01  DTEMP= 8.110642E-09  DSALT= 4.116066E-13  SCANS=  8
TS=  36  YEAR=  0.20  DAY= 72.0  ENERGY= 3.419514E-01  DTEMP= 8.098844E-09  DSALT= 4.032546E-13  SCANS=  7
TS=  37  YEAR=  0.20  DAY= 74.0  ENERGY= 3.444725E-01  DTEMP= 8.089856E-09  DSALT= 3.993563E-13  SCANS=  8
TS=  38  YEAR=  0.21  DAY= 76.0  ENERGY= 3.481850E-01  DTEMP= 8.179109E-09  DSALT= 3.968722E-13  SCANS=  7
TS=  39  YEAR=  0.21  DAY= 78.0  ENERGY= 3.513861E-01  DTEMP= 8.173460E-09  DSALT= 3.953416E-13  SCANS=  9
TS=  40  YEAR=  0.22  DAY= 80.0  ENERGY= 3.556086E-01  DTEMP= 8.091395E-09  DSALT= 3.926051E-13  SCANS=  7
TS=  41  YEAR=  0.22  DAY= 82.0  ENERGY= 3.639250E-01  DTEMP= 8.948876E-09  DSALT= 6.304794E-13  SCANS= 15
TS=  42  YEAR=  0.23  DAY= 84.0  ENERGY= 3.636970E-01  DTEMP= 7.825564E-09  DSALT= 3.917466E-13  SCANS= 15
TS=  43  YEAR=  0.24  DAY= 86.0  ENERGY= 3.680011E-01  DTEMP= 7.468309E-09  DSALT= 3.866975E-13  SCANS= 16
TS=  44  YEAR=  0.24  DAY= 88.0  ENERGY= 3.726818E-01  DTEMP= 7.226034E-09  DSALT= 3.772885E-13  SCANS= 16
TS=  45  YEAR=  0.25  DAY= 90.0  ENERGY= 3.772635E-01  DTEMP= 7.095960E-09  DSALT= 3.729902E-13  SCANS=  9
TS=  46  YEAR=  0.25  DAY= 92.0  ENERGY= 3.821124E-01  DTEMP= 7.420118E-09  DSALT= 3.813201E-13  SCANS=  9
TS=  47  YEAR=  0.26  DAY= 94.0  ENERGY= 3.867683E-01  DTEMP= 7.757427E-09  DSALT= 3.849632E-13  SCANS=  9
TS=  48  YEAR=  0.26  DAY= 96.0  ENERGY= 3.915996E-01  DTEMP= 7.933752E-09  DSALT= 3.806562E-13  SCANS=  9
TS=  49  YEAR=  0.27  DAY= 98.0  ENERGY= 3.961591E-01  DTEMP= 7.866879E-09  DSALT= 3.791283E-13  SCANS=  9
TEMPERATURE FOR J =    7 AT TIMESTEP     50
          2      3      4      5      6      7      8      9     10     11     12     13     14     15     16     17     18     19     20     21

  1   10.06  10.27  10.37  10.39  10.41  10.41  10.40  10.40  10.39  10.38  10.38  10.37  10.37  10.37  10.37  10.37  10.37  10.37  10.37  10.37
  2    5.70   5.80   5.92   5.95   5.96   5.97   5.96   5.95   5.94   5.94   5.94   5.94   5.94   5.93   5.93   5.93   5.93   5.93   5.93   5.93
  3    4.16   4.16   4.21   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22   4.22
  4    4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01   4.01
```

5	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
6	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
7	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
8	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
9	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
10	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
11	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
12	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
13	0.00	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
14	0.00	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
15	0.00	0.00	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00

SALINITY FOR J = 7 AT TIMESTEP 50

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	0.44	0.44	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45
2	0.05	0.06	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
3	-0.09	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08
4	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
5	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
6	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
7	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
8	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
9	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
10	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
11	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
12	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
13	10.00	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
14	10.00	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
15	10.00	10.00	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10

W VELOCITY FOR J = 7 AT TIMESTEP 50

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	-0.08	-0.10	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13	-0.13
3	-0.05	-0.09	-0.13	-0.14	-0.14	-0.14	-0.13	-0.13	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12
4	-0.03	-0.07	-0.13	-0.14	-0.15	-0.15	-0.13	-0.13	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12
5	0.00	-0.05	-0.13	-0.15	-0.17	-0.17	-0.14	-0.13	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12
6	0.00	-0.03	-0.11	-0.16	-0.19	-0.19	-0.15	-0.13	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12	-0.12
7	0.00	0.00	-0.05	-0.17	-0.21	-0.22	-0.15	-0.13	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11
8	0.00	0.00	-0.05	-0.18	-0.25	-0.26	-0.17	-0.13	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11	-0.11
9	0.00	0.00	0.00	-0.18	-0.29	-0.31	-0.18	-0.13	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
10	0.00	0.00	0.00	-0.10	-0.33	-0.36	-0.20	-0.13	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
11	0.00	0.00	0.00	0.00	-0.39	-0.44	-0.22	-0.13	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09
12	0.00	0.00	0.00	0.00	-0.21	-0.50	-0.25	-0.13	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07
13	0.00	0.00	0.00	0.00	0.00	-0.57	-0.28	-0.13	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06
14	0.00	0.00	0.00	0.00	0.00	-0.31	-0.24	-0.12	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04
15	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	-0.12	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

U VELOCITY FOR J = 7 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.00	-0.18	-0.04	-0.06	-0.05	-0.04	-0.08	-0.14	-0.21	-0.26	-0.29	-0.30	-0.29	-0.28	-0.27	-0.26	-0.24	-0.22	-0.21	-0.19
2	0.00	0.09	0.10	0.12	0.14	0.15	0.11	0.05	-0.02	-0.07	-0.10	-0.11	-0.10	-0.10	-0.08	-0.07	-0.05	-0.04	-0.02	0.00
3	0.00	0.05	0.06	0.07	0.09	0.10	0.06	0.00	-0.07	-0.12	-0.15	-0.16	-0.15	-0.14	-0.13	-0.12	-0.10	-0.08	-0.07	-0.05
4	0.00	0.04	0.05	0.06	0.08	0.09	0.06	0.00	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
5	0.00	0.00	0.02	0.06	0.08	0.09	0.06	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
6	0.00	0.00	0.02	0.06	0.08	0.09	0.06	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
7	0.00	0.00	0.00	0.05	0.08	0.09	0.06	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
8	0.00	0.00	0.00	0.05	0.08	0.09	0.06	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
9	0.00	0.00	0.00	0.00	0.07	0.09	0.06	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
10	0.00	0.00	0.00	0.00	0.07	0.09	0.06	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
11	0.00	0.00	0.00	0.00	0.00	0.08	0.06	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
12	0.00	0.00	0.00	0.00	0.00	0.08	0.06	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
13	0.00	0.00	0.00	0.00	0.00	0.00	0.04	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
14	0.00	0.00	0.00	0.00	0.00	0.00	0.04	-0.01	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.03	-0.08	-0.13	-0.15	-0.16	-0.16	-0.15	-0.14	-0.12	-0.11	-0.09	-0.07	-0.06

V VELOCITY FOR J = 7 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.00	-7.06	-6.77	-6.66	-6.82	-7.00	-7.05	-6.89	-6.71	-6.57	-6.47	-6.41	-6.38	-6.36	-6.35	-6.35	-6.35	-6.36	-6.37	-6.37
2	0.00	-0.46	-0.19	-0.12	-0.29	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.10	0.14	0.16	0.16	0.17	0.16	0.16	0.15	0.15
3	0.00	-0.45	-0.16	-0.11	-0.29	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
4	0.00	-0.44	-0.15	-0.11	-0.28	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15

5	0.00	0.00	-0.16	-0.11	-0.28	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
6	0.00	0.00	-0.16	-0.11	-0.28	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
7	0.00	0.00	0.00	-0.11	-0.28	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
8	0.00	0.00	0.00	-0.11	-0.28	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
9	0.00	0.00	0.00	0.00	-0.28	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
10	0.00	0.00	0.00	0.00	-0.28	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
11	0.00	0.00	0.00	0.00	0.00	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
12	0.00	0.00	0.00	0.00	0.00	-0.48	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
13	0.00	0.00	0.00	0.00	0.00	0.00	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
14	0.00	0.00	0.00	0.00	0.00	0.00	-0.53	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.37	-0.20	-0.05	0.05	0.11	0.14	0.16	0.17	0.17	0.17	0.16	0.16	0.15

TEMPERATURE FOR J = 14 AT TIMESTEP 50

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	15.38	15.00	15.18	15.22	15.22	15.23	15.25	15.28	15.29	15.29	15.29	15.30	15.30	15.30	15.30	15.30	15.30	15.30	15.30	15.30
2	7.05	6.56	6.77	6.83	6.83	6.83	6.86	6.89	6.90	6.90	6.90	6.90	6.90	6.90	6.90	6.90	6.90	6.90	6.90	6.90
3	4.27	4.19	4.21	4.22	4.21	4.21	4.22	4.23	4.24	4.24	4.24	4.24	4.24	4.24	4.24	4.24	4.24	4.24	4.24	4.24
4	4.01	4.01	4.01	4.01	4.00	4.00	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01
5	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
6	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
7	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
8	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
9	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
10	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
11	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
12	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
13	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
14	0.00	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
15	0.00	0.00	0.00	0.00	0.00	0.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00

SALINITY FOR J = 14 AT TIMESTEP 50

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	0.16	0.13	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14
2	-0.03	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04
3	-0.09	-0.10	-0.10	-0.10	-0.10	-0.10	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09
4	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
5	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
6	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
7	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
8	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
9	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
10	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
11	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
12	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
13	10.00	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
14	10.00	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
15	10.00	10.00	10.00	10.00	10.00	10.00	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10

W VELOCITY FOR J = 14 AT TIMESTEP 50

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	-0.03	0.28	0.37	0.43	0.45	0.46	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45
3	-0.02	0.16	0.27	0.35	0.40	0.42	0.41	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40
4	-0.01	0.03	0.16	0.30	0.39	0.43	0.41	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38
5	0.00	-0.13	0.03	0.24	0.38	0.45	0.42	0.38	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37
6	0.00	-0.07	-0.05	0.18	0.39	0.48	0.44	0.37	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36
7	0.00	0.00	-0.16	0.10	0.39	0.53	0.46	0.36	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35
8	0.00	0.00	-0.09	0.09	0.41	0.59	0.49	0.35	0.33	0.34	0.34	0.34	0.34	0.34	0.34	0.34	0.34	0.34	0.34	0.34
9	0.00	0.00	0.00	0.09	0.44	0.68	0.53	0.33	0.31	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32
10	0.00	0.00	0.00	0.05	0.55	0.80	0.58	0.31	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
11	0.00	0.00	0.00	0.00	0.67	0.95	0.64	0.29	0.25	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
12	0.00	0.00	0.00	0.00	0.37	1.14	0.74	0.25	0.21	0.22	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23
13	0.00	0.00	0.00	0.00	0.00	1.36	0.85	0.23	0.17	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18
14	0.00	0.00	0.00	0.00	0.00	0.73	0.91	0.21	0.12	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
15	0.00	0.00	0.00	0.00	0.00	0.00	0.97	0.19	0.06	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

U VELOCITY FOR J = 14 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.00	-1.64	-1.88	-2.12	-2.30	-2.39	-2.40	-2.37	-2.34	-2.32	-2.30	-2.29	-2.28	-2.27	-2.26	-2.25	-2.24	-2.22	-2.21	-2.19
2	0.00	0.23	-0.05	-0.27	-0.46	-0.58	-0.59	-0.57	-0.55	-0.53	-0.51	-0.50	-0.49	-0.48	-0.47	-0.46	-0.45	-0.44	-0.42	-0.40
3	0.00	0.42	0.33	0.09	-0.13	-0.26	-0.28	-0.26	-0.24	-0.22	-0.21	-0.20	-0.19	-0.18	-0.17	-0.16	-0.14	-0.13	-0.12	-0.10
4	0.00	0.43	0.39	0.15	-0.08	-0.21	-0.23	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.10	-0.08	-0.07	-0.05

5	0.00	0.00	0.25	0.13	-0.08	-0.21	-0.23	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
6	0.00	0.00	0.25	0.13	-0.08	-0.21	-0.23	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
7	0.00	0.00	0.00	0.05	-0.09	-0.21	-0.23	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
8	0.00	0.00	0.00	0.05	-0.09	-0.21	-0.23	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
9	0.00	0.00	0.00	0.00	-0.11	-0.21	-0.23	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
10	0.00	0.00	0.00	0.00	-0.11	-0.21	-0.23	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
11	0.00	0.00	0.00	0.00	0.00	-0.20	-0.24	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
12	0.00	0.00	0.00	0.00	0.00	-0.20	-0.24	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
13	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
14	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	-0.21	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.17	-0.19	-0.17	-0.16	-0.15	-0.14	-0.13	-0.12	-0.11	-0.09	-0.08	-0.07	-0.05

V VELOCITY FOR J = 14 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.00	-11.26	-11.26	-10.24	-9.60	-9.28	-9.20	-9.37	-9.63	-9.83	-9.94	-9.98	-9.99	-9.98	-9.97	-9.96	-9.95	-9.94	-9.94	-9.94
2	0.00	-0.72	-1.13	-0.57	-0.05	0.27	0.36	0.19	-0.09	-0.30	-0.41	-0.45	-0.46	-0.45	-0.44	-0.43	-0.42	-0.42	-0.41	-0.41
3	0.00	-0.75	-0.76	-0.23	0.28	0.60	0.71	0.53	0.24	0.03	-0.08	-0.13	-0.14	-0.13	-0.12	-0.11	-0.10	-0.09	-0.09	-0.08
4	0.00	-0.75	-0.70	-0.17	0.34	0.66	0.77	0.59	0.30	0.09	-0.03	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
5	0.00	0.00	-0.59	-0.12	0.35	0.67	0.78	0.60	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
6	0.00	0.00	-0.59	-0.12	0.35	0.67	0.78	0.60	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
7	0.00	0.00	0.00	-0.07	0.37	0.67	0.78	0.60	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
8	0.00	0.00	0.00	-0.07	0.37	0.67	0.78	0.60	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
9	0.00	0.00	0.00	0.00	0.40	0.68	0.78	0.60	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
10	0.00	0.00	0.00	0.00	0.40	0.68	0.78	0.60	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
11	0.00	0.00	0.00	0.00	0.00	0.70	0.79	0.60	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
12	0.00	0.00	0.00	0.00	0.00	0.70	0.79	0.60	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
13	0.00	0.00	0.00	0.00	0.00	0.00	0.80	0.61	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.03	-0.02
14	0.00	0.00	0.00	0.00	0.00	0.00	0.80	0.61	0.30	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.02	-0.02
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.62	0.31	0.09	-0.02	-0.07	-0.08	-0.07	-0.06	-0.05	-0.04	-0.03	-0.02	-0.02

TS= 50 YEAR= 0.27 DAY=100.0 ENERGY= 4.008378E-01 DTEMP= 7.759637E-09 DSALT= 3.876320E-13 SCANS= 9

WORK BY:	INTERNAL MODE	EXTERNAL MODE
TIME RATE OF CHANGE	5.009573E-07	1.529970E-07
HORIZONTAL ADVECTION	-7.323394E-09	-4.192204E-10
VERTICAL ADVECTION	7.716057E-09	2.655726E-11
HORIZONTAL FRICTION	-4.498561E-07	-8.228646E-08
VERTICAL FRICTION	-6.101644E-08	4.606451E-24
PRESSURE FORCES	3.319829E-07	1.683767E-08
IMPLICIT EFFECTS	-3.776211E-08	4.353381E-09
WORK BY WIND	7.171671E-07	2.149473E-07
BOTTOM DRAG	4.924038E-11	-4.622055E-10
WORK BY BUOYANCY FORCES	3.488205E-07	

	TEMPERATURE	SALINITY
TIME RATE OF CHANGE	5.393308E-09	3.392059E-13
HORIZONTAL ADVECTION	-1.952206E-20	-6.349940E-25
VERTICAL ADVECTION	-3.712185E-22	-7.883286E-27
HORIZONTAL DIFFUSION	-2.166228E-22	-1.702176E-26
SURFACE FLUX	-2.390461E-23	-8.000985E-28
TRUNCATION ERROR	5.393308E-09	3.392059E-13
CHANGE OF VARIANCE	8.651227E-08	1.081689E-17

ENERGY CONVERSION ERROR -9.554955E-18 NONLINEAR EXCHANGE ERROR -1.173087E-20

NORTHWARD TRANSPORT OF HEAT (X10**15 WATTS)

	X MEAN	X EDDY	Z MEAN	Z EDDY	EKMAN	TOT ADV	DIFFUS	TOTAL
18	-0.205	-0.026	0.011	-0.242	0.000	-0.232	-0.003	-0.234
17	-0.252	-0.006	0.013	0.494	-0.765	-0.258	0.001	-0.257
16	-1.199	0.004	0.003	-0.246	-0.952	-1.195	0.001	-1.194
15	-0.961	0.001	-0.007	0.096	-1.049	-0.960	0.016	-0.943
14	-0.834	0.000	-0.010	0.234	-1.058	-0.834	-0.008	-0.843
13	-1.139	-0.001	-0.010	-0.164	-0.966	-1.139	-0.001	-1.141
12	-0.777	-0.001	-0.010	0.106	-0.874	-0.778	0.000	-0.778
11	-0.674	-0.001	-0.010	0.086	-0.752	-0.676	-0.006	-0.681
10	-0.660	-0.001	-0.008	-0.038	-0.616	-0.662	-0.004	-0.666
9	-0.476	-0.001	-0.007	0.019	-0.490	-0.477	-0.004	-0.481
8	-0.339	-0.001	-0.005	0.037	-0.371	-0.340	-0.005	-0.345
7	-0.261	-0.001	-0.004	0.008	-0.265	-0.261	-0.005	-0.266
6	-0.180	0.000	-0.003	-0.001	-0.176	-0.180	-0.004	-0.184
5	-0.105	0.000	-0.003	0.003	-0.105	-0.105	0.000	-0.105
4	-0.034	0.000	-0.002	0.001	-0.034	-0.034	0.001	-0.033
3	-0.004	0.000	-0.001	0.000	-0.003	-0.005	-0.007	-0.012
2	0.000	0.000	0.000	0.000	0.000	0.000	-0.005	-0.006

NORTHWARD TRANSPORT OF SALT (X10**10 CM**3/SEC)

	X MEAN	X EDDY	Z MEAN	Z EDDY	EKMAN	TOT ADV	DIFFUS	TOTAL
18	-0.035	-0.003	0.002	-0.040	0.000	-0.038	0.003	-0.035
17	-0.058	-0.001	0.003	0.114	-0.175	-0.058	0.004	-0.054
16	-0.351	0.003	0.001	-0.071	-0.278	-0.348	0.004	-0.344
15	-0.353	0.002	-0.003	0.037	-0.386	-0.351	0.012	-0.339
14	-0.380	0.001	-0.005	0.105	-0.482	-0.379	0.004	-0.375
13	-0.640	0.002	-0.006	-0.090	-0.543	-0.638	0.007	-0.631
12	-0.533	0.001	-0.007	0.075	-0.599	-0.532	0.009	-0.522
11	-0.564	0.001	-0.008	0.073	-0.628	-0.563	0.006	-0.557
10	-0.672	0.001	-0.009	-0.036	-0.626	-0.671	0.007	-0.664
9	-0.589	0.000	-0.009	0.025	-0.605	-0.588	0.008	-0.581
8	-0.512	0.000	-0.008	0.057	-0.560	-0.511	0.006	-0.505
7	-0.484	0.000	-0.008	0.015	-0.491	-0.484	0.006	-0.478
6	-0.415	0.000	-0.007	0.000	-0.407	-0.415	0.009	-0.406
5	-0.306	0.000	-0.007	0.009	-0.308	-0.306	0.024	-0.283
4	-0.119	-0.001	-0.007	0.005	-0.117	-0.120	0.033	-0.086
3	-0.022	-0.001	-0.005	-0.001	-0.017	-0.023	0.003	-0.020
2	-0.003	0.000	-0.002	-0.002	0.000	-0.003	0.004	0.001

MERIDIONAL MASS TRANSPORT

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.00	-0.04	-1.41	-3.02	-4.96	-7.23	-9.17	-10.57	-13.47	-17.16	-16.05	-17.57	-24.64	-16.72	-20.13	-24.54	-5.35	-3.53	0.00	0.00
2	0.00	-0.10	-1.48	-3.08	-5.02	-7.28	-9.20	-10.60	-13.44	-17.15	-16.23	-17.14	-24.53	-17.59	-17.51	-26.49	-2.89	-6.99	0.00	0.00
3	0.00	-0.18	-1.56	-3.18	-5.10	-7.33	-9.24	-10.63	-13.43	-17.07	-16.29	-17.00	-24.21	-17.72	-16.85	-26.07	-1.30	-9.03	0.00	0.00
4	0.00	-0.26	-1.67	-3.32	-5.21	-7.40	-9.28	-10.68	-13.42	-17.00	-16.29	-16.93	-23.77	-17.60	-16.56	-25.06	0.99	-11.23	0.00	0.00
5	0.00	-0.33	-1.73	-3.37	-5.22	-7.39	-9.24	-10.65	-13.34	-16.84	-16.18	-16.74	-23.21	-17.17	-15.95	-23.87	3.35	-14.22	0.00	0.00
6	0.00	-0.37	-1.78	-3.43	-5.24	-7.36	-9.19	-10.60	-13.24	-16.62	-16.03	-16.50	-22.45	-16.59	-15.13	-22.12	6.19	-16.98	0.00	0.00
7	0.00	-0.42	-1.80	-3.42	-5.18	-7.25	-9.06	-10.47	-13.02	-16.23	-15.64	-15.93	-21.32	-15.53	-13.86	-20.07	8.73	-19.84	0.00	0.00
8	0.00	-0.44	-1.80	-3.38	-5.10	-7.12	-8.89	-10.30	-12.74	-15.73	-15.13	-15.22	-19.88	-14.18	-12.21	-17.27	11.20	-21.66	0.00	0.00
9	0.30	-0.46	-1.78	-3.29	-4.96	-6.90	-8.62	-9.96	-12.19	-14.86	-14.17	-14.02	-17.94	-12.41	-10.29	-14.43	12.80	-22.98	0.00	0.00
10	0.00	-0.45	-1.72	-3.17	-4.78	-6.64	-8.27	-9.54	-11.32	-13.79	-12.99	-12.54	-15.56	-10.22	-7.91	-10.88	13.82	-22.59	0.00	0.00
11	0.00	-0.44	-1.61	-2.95	-4.45	-6.13	-7.58	-8.66	-10.28	-12.07	-11.18	-10.53	-12.81	-7.98	-5.80	-7.90	13.42	-21.33	0.00	0.00
12	0.00	-0.41	-1.46	-2.70	-4.06	-5.52	-6.75	-7.60	-8.79	-10.00	-9.01	-8.13	-9.51	-5.31	-3.28	-4.52	12.14	-18.03	0.00	0.00
13	0.00	-0.35	-1.17	-2.14	-3.17	-4.24	-5.10	-5.66	-6.41	-7.14	-6.29	-5.52	-6.37	-3.28	-1.77	-2.40	9.23	-13.78	0.00	0.00
14	0.00	-0.25	-0.83	-1.49	-2.13	-2.75	-3.19	-3.40	-3.64	-3.81	-3.13	-2.49	-2.71	-0.94	-0.06	-0.32	5.36	-7.46	0.00	0.00

15 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00

STREAM FUNCTION IN SVERDRUPS, TS= 50

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
20	0.00	-1.47	-2.56	-3.50	-4.43	-5.33	-6.00	-6.28	-6.29	-6.10	-5.80	-5.45	-5.09	-4.73	-4.38	-4.03	-3.67	-3.29	-2.87	-2.39
19	0.00	1.47	2.56	3.50	4.43	5.33	6.00	6.28	6.29	6.10	5.80	5.45	5.09	4.73	4.38	4.03	3.67	3.29	2.87	2.39
18	0.00	2.05	4.12	6.27	8.88	11.79	14.16	15.27	15.47	15.01	14.18	13.18	12.14	11.11	10.11	9.13	8.13	7.08	5.94	4.66
17	0.00	0.64	1.51	3.27	6.64	11.49	16.56	19.55	20.49	20.04	18.86	17.36	15.78	14.23	12.74	11.28	9.83	8.34	6.78	5.11
16	0.00	-0.88	-1.76	-1.38	1.40	6.97	14.43	19.80	21.82	21.63	20.32	18.54	16.63	14.77	13.00	11.31	9.65	7.99	6.31	4.61
15	0.00	-1.17	-2.70	-3.40	-1.94	2.74	10.65	17.38	20.24	20.46	19.17	17.35	15.38	13.45	11.64	9.94	8.32	6.75	5.21	3.71
14	0.00	-0.96	-2.40	-3.62	-3.52	-0.57	6.25	13.18	16.44	16.96	15.99	14.38	12.59	10.84	9.22	7.73	6.36	5.07	3.85	2.70
13	0.00	-0.85	-2.17	-3.69	-4.69	-3.60	1.47	7.77	11.07	11.89	11.31	10.11	8.72	7.36	6.13	5.04	4.07	3.19	2.39	1.66
12	0.00	-0.85	-2.10	-3.75	-5.52	-6.07	-3.13	1.88	4.81	5.80	5.66	4.99	4.16	3.35	2.65	2.07	1.61	1.23	0.91	0.64
11	0.00	-0.80	-1.89	-3.43	-5.56	-7.45	-6.85	-3.75	-1.63	-0.67	-0.44	-0.55	-0.76	-0.94	-1.03	-1.01	-0.91	-0.75	-0.57	-0.38
10	0.00	-0.70	-1.54	-2.79	-4.92	-7.71	-9.38	-8.59	-7.63	-6.97	-6.51	-6.13	-5.73	-5.27	-4.73	-4.11	-3.43	-2.74	-2.05	-1.40
9	0.00	-0.62	-1.27	-2.18	-4.06	-7.28	-10.77	-12.29	-12.72	-12.56	-12.06	-11.33	-10.43	-9.40	-8.28	-7.10	-5.89	-4.68	-3.51	-2.41
8	0.00	-0.58	-1.13	-1.79	-3.29	-6.48	-11.15	-14.60	-16.48	-16.99	-16.63	-15.73	-14.49	-13.04	-11.46	-9.82	-8.15	-6.51	-4.91	-3.40
7	0.00	-0.54	-1.04	-1.52	-2.63	-5.45	-10.61	-15.59	-18.59	-19.80	-19.75	-18.88	-17.51	-15.83	-13.97	-12.03	-10.06	-8.09	-6.17	-4.32
6	0.00	-0.49	-0.94	-1.30	-2.04	-4.27	-9.24	-14.98	-18.81	-20.63	-20.97	-20.31	-19.01	-17.33	-15.43	-13.40	-11.31	-9.21	-7.12	-5.07
5	0.00	-0.41	-0.81	-1.11	-1.56	-3.08	-7.25	-12.90	-16.99	-19.18	-19.86	-19.50	-18.46	-17.01	-15.30	-13.44	-11.50	-9.51	-7.49	-5.45
4	0.00	-0.27	-0.55	-0.80	-1.06	-1.89	-4.79	-9.45	-13.10	-15.20	-16.01	-15.93	-15.26	-14.21	-12.93	-11.51	-10.00	-8.43	-6.79	-5.08
3	0.00	-0.13	-0.25	-0.39	-0.52	-0.79	-2.09	-4.93	-7.20	-8.54	-9.11	-9.14	-8.83	-8.30	-7.63	-6.88	-6.08	-5.23	-4.33	-3.36
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

	22	23	24
20	-1.83	-1.10	0.00
19	1.83	1.10	0.00
18	3.26	1.64	0.00
17	3.36	1.56	0.00
16	2.91	1.29	0.00
15	2.28	0.96	0.00
14	1.64	0.70	0.00
13	1.00	0.43	0.00
12	0.39	0.17	0.00
11	-0.21	-0.08	0.00
10	-0.81	-0.32	0.00
9	-1.42	-0.57	0.00
8	-2.02	-0.82	0.00
7	-2.60	-1.07	0.00
6	-3.10	-1.30	0.00
5	-3.41	-1.47	0.00
4	-3.29	-1.47	0.00
3	-2.29	-1.11	0.00
2	0.00	0.00	0.00
1	0.00	0.00	0.00

```
TS=   51  YEAR=  0.28  DAY=102.0  ENERGY= 4.052294E-01  DTEMP= 7.366084E-09  DSALT= 3.797967E-13  SCANS=  8
TS=   52  YEAR=  0.28  DAY=104.0  ENERGY= 4.096870E-01  DTEMP= 7.253038E-09  DSALT= 3.831186E-13  SCANS=  9
TS=   53  YEAR=  0.29  DAY=106.0  ENERGY= 4.139006E-01  DTEMP= 7.274828E-09  DSALT= 3.883550E-13  SCANS=  8
TS=   54  YEAR=  0.30  DAY=108.0  ENERGY= 4.180961E-01  DTEMP= 7.295525E-09  DSALT= 3.759928E-13  SCANS=  9
TS=   55  YEAR=  0.30  DAY=110.0  ENERGY= 4.221459E-01  DTEMP= 7.377924E-09  DSALT= 3.825248E-13  SCANS=  8
TS=   56  YEAR=  0.31  DAY=112.0  ENERGY= 4.260806E-01  DTEMP= 7.271875E-09  DSALT= 3.707919E-13  SCANS=  8
TS=   57  YEAR=  0.31  DAY=114.0  ENERGY= 4.300267E-01  DTEMP= 7.033864E-09  DSALT= 3.594878E-13  SCANS=  8
TS=   58  YEAR=  0.32  DAY=116.0  ENERGY= 4.337822E-01  DTEMP= 6.780761E-09  DSALT= 3.566538E-13  SCANS=  8
TS=   59  YEAR=  0.32  DAY=118.0  ENERGY= 4.377406E-01  DTEMP= 6.581893E-09  DSALT= 3.439638E-13  SCANS=  8
TS=   60  YEAR=  0.33  DAY=120.0  ENERGY= 4.414474E-01  DTEMP= 6.507250E-09  DSALT= 3.456671E-13  SCANS=  7
TS=   61  YEAR=  0.33  DAY=122.0  ENERGY= 4.495636E-01  DTEMP= 7.781031E-09  DSALT= 5.972492E-13  SCANS= 13
TS=   62  YEAR=  0.34  DAY=124.0  ENERGY= 4.493945E-01  DTEMP= 6.619571E-09  DSALT= 3.395764E-13  SCANS= 12
TS=   63  YEAR=  0.34  DAY=126.0  ENERGY= 4.533766E-01  DTEMP= 6.826450E-09  DSALT= 3.445475E-13  SCANS= 15
TS=   64  YEAR=  0.35  DAY=128.0  ENERGY= 4.572278E-01  DTEMP= 6.981250E-09  DSALT= 3.407745E-13  SCANS= 12
TS=   65  YEAR=  0.36  DAY=130.0  ENERGY= 4.610347E-01  DTEMP= 7.046453E-09  DSALT= 3.433195E-13  SCANS=  9
TS=   66  YEAR=  0.36  DAY=132.0  ENERGY= 4.646911E-01  DTEMP= 7.052799E-09  DSALT= 3.429988E-13  SCANS=  9
TS=   67  YEAR=  0.37  DAY=134.0  ENERGY= 4.681890E-01  DTEMP= 6.906529E-09  DSALT= 3.452338E-13  SCANS=  9
TS=   68  YEAR=  0.37  DAY=136.0  ENERGY= 4.716351E-01  DTEMP= 6.667770E-09  DSALT= 3.442434E-13  SCANS=  7
TS=   69  YEAR=  0.38  DAY=138.0  ENERGY= 4.749365E-01  DTEMP= 6.260320E-09  DSALT= 3.377847E-13  SCANS=  9
TS=   70  YEAR=  0.38  DAY=140.0  ENERGY= 4.784568E-01  DTEMP= 5.799264E-09  DSALT= 3.274133E-13  SCANS=  7
TS=   71  YEAR=  0.39  DAY=142.0  ENERGY= 4.819737E-01  DTEMP= 5.300198E-09  DSALT= 3.200142E-13  SCANS=  8
TS=   72  YEAR=  0.39  DAY=144.0  ENERGY= 4.858913E-01  DTEMP= 4.978595E-09  DSALT= 3.146477E-13  SCANS=  7
TS=   73  YEAR=  0.40  DAY=146.0  ENERGY= 4.899279E-01  DTEMP= 4.784163E-09  DSALT= 3.115673E-13  SCANS=  8
TS=   74  YEAR=  0.41  DAY=148.0  ENERGY= 4.944135E-01  DTEMP= 4.820552E-09  DSALT= 3.138085E-13  SCANS=  6
TS=   75  YEAR=  0.41  DAY=150.0  ENERGY= 4.989988E-01  DTEMP= 5.002761E-09  DSALT= 3.200029E-13  SCANS=  7
TS=   76  YEAR=  0.42  DAY=152.0  ENERGY= 5.038975E-01  DTEMP= 5.278165E-09  DSALT= 3.323801E-13  SCANS=  7
TS=   77  YEAR=  0.42  DAY=154.0  ENERGY= 5.088110E-01  DTEMP= 5.457471E-09  DSALT= 3.306724E-13  SCANS=  6
TS=   78  YEAR=  0.43  DAY=156.0  ENERGY= 5.138442E-01  DTEMP= 5.551255E-09  DSALT= 3.258207E-13  SCANS=  8
TS=   79  YEAR=  0.43  DAY=158.0  ENERGY= 5.188403E-01  DTEMP= 5.576802E-09  DSALT= 3.248495E-13  SCANS=  6
TS=   80  YEAR=  0.44  DAY=160.0  ENERGY= 5.238107E-01  DTEMP= 5.559700E-09  DSALT= 3.242970E-13  SCANS=  8
TS=   81  YEAR=  0.44  DAY=162.0  ENERGY= 5.236683E-01  DTEMP= 6.592115E-09  DSALT= 5.804601E-13  SCANS= 13
```

Model 3 is a global model with greatly simplified land masses. It contains an island at its southern extremity and another in the interior. The option line specifies that Fourier filtering is to be done to handle the convergence of meridians at high latitudes, that islands are present, that cyclic continuity exists east-west, and that comments are to be included in the FORTRAN listing. The NAMELIST input file is similar in structure to that of Model 1 although, on line 1, "NA=1" has been added, directing that a restart file be sent to unit 21 at the end of the run.

UPDATE FOR MODEL 3:

```
OPT=F,I,O,K,C
      PARAMETER (IMT=062,JMT=032,KM=15,NT=2,LSEG=3,NISLE=2,LBC=2                000200000
     *,LSEGF=3,JFRST=3,JFT0=8,JFT1=7,JFT2=26,JFU0=8,JFU1=7,JFU2=27              000300000
      SWLDEG=-90.0                                                              111100000
        DYT(J)=6.0                                                              111800000
        DXT(I)=6.0                                                              112900000
      DZ(1)=30.00E2                                                             114400000
      DZ(2)=46.30E2                                                             114400010
      DZ(3)=67.45E2                                                             114400020
      DZ(4)=94.46E2                                                             114400030
      DZ(5)=128.46E2                                                            114400040
      DZ(6)=170.57E2                                                            114400050
      DZ(7)=221.91E2                                                            114400060
      DZ(8)=283.51E2                                                            114400070
      DZ(9)=356.21E2                                                            114400080
      DZ(10)=440.57E2                                                           114400090
      DZ(11)=536.78E2                                                           114400100
      DZ(12)=644.54E2                                                           114400110
      DZ(13)=763.00E2                                                           114400120
      DZ(14)=890.71E2                                                           114400130
      DZ(15)=1025.53E2                                                          114400140
        FKMP(I,J)=KM                                                            133400000
      DO 702 J=10,27                                                            133500010
      DO 702 I=2,10                                                             133500020
        FKMP(I,J)=0                                                             133500030
 702  CONTINUE                                                                  133500040
      DO 704 J=8,27                                                             133500050
      DO 704 I=34,45                                                            133500060
        FKMP(I,J)=0                                                             133500070
 704  CONTINUE                                                                  133500080
      DO 706 J=8,11                                                             133500090
      DO 706 I=15,18                                                            133500100
        FKMP(I,J)=0                                                             133500110
 706  CONTINUE                                                                  133500120
      DO 708 I=11,33                                                            133500130
        FKMP(I,27)=0                                                            133500140
 708  CONTINUE                                                                  133500150
      DO 710 J=10,27                                                            133500160
      DO 710 I=60,61                                                            133500170
        FKMP(I,J)=0                                                             133500180
 710  CONTINUE                                                                  133500190
      DO 712 I=2,IMTM1                                                          133500200
        FKMP(I,2)=0                                                             133500210
 712  CONTINUE                                                                  133500220
      DO 714 J=3,4                                                              133500230
        FKMP(2,J)=0                                                             133500240
 714  CONTINUE                                                                  133500250
        FX=(SIN(3.*PI*FLOAT(J-2)/FLOAT(JMT-4)))*                                151900010
     *       (1.1+COS(2.*PI*FLOAT(J-2)/FLOAT(JMT-4)))*.5                        151900020
          WSX(I)=FX                                                             152400000
          WSY(I)=0.0                                                            152500000
      MTEST=MOD(J,7)                                                            249100000
      IPRT=20                                                                   249400000
C                                                                               420100010
C-----------------------------------------------------------------------       420100020
C  SET NEWTONIAN SURFACE BOUNDARY CONDITION                                     420100030
C-----------------------------------------------------------------------       420100040
C                                                                               420100050
      FXA=25.0                                                                  420100060
      FXB=-15.0                                                                 420100070
      IF(M.EQ.2) THEN                                                           420100080
        FXA=0.0                                                                 420100090
        FXB=+.001                                                               420100100
      ENDIF                                                                     420100110
      DO 846 I=2,IMTM1                                                          420100120
        TA(I,1,M)=TA(I,1,M)+2.315E-7*(FXA+FXB*ABS(PHIT(J))-TB(I,1,M))           420100130
 846  CONTINUE                                                                  420100140
C                                                                               427200010
C  ADD IN CONTRIBUTION FROM NEWTONIAN BOUNDARY CONDITION                        427200020
C                                                                               427200030
          IF(M.EQ.1) TTDTOT(5,1)=TTDTOT(5,1)                                    427200040
     *                     +2.315E-7*(25.0-15.0*ABS(PHIT(J))-TB(I,1,1))         427200050
          IF(M.EQ.2) TTDTOT(5,2)=TTDTOT(5,2)                                    427200060
     *                     +2.315E-7*(      +.001*ABS(PHIT(J))-TB(I,1,2))       427200070
      DATA TO/                                                                  603800000
     *        13.4979166,13.4926065,13.4846595,13.4732852, 8.4679890,           603800010
     *         8.4517059, 5.9421330, 4.4281250, 3.9040599, 2.8802061,           603800020
     *         2.8360754, 2.7780657, 2.7025290, 2.6052822, 2.9625234/           603800030
      DATA SO/                                                                  603900000
     *        -.0022500, -.0022500, -.0022500, -.0022500,  .0001500,            603900010
     *         .0001500, -.0001000, -.0001000, -.0002500, -.0002000,            603900020
     *        -.0002000, -.0002000, -.0002000, -.0002000, -.0002000/           603900030
```

```
      DATA (C( 1,N),N=1,9)/
     *           -.2019152E-03, .7709387E+00,-.4915766E-05,-.2007672E-02,
     *            .4496149E+00, .3652747E-07, .4725372E-02, .3768380E-04,
     *            .6548196E+01/
      DATA (C( 2,N),N=1,9)/
     *           -.2026279E-03, .7706838E+00,-.4903913E-05,-.2004045E-02,
     *            .4497566E+00, .3639690E-07, .4712010E-02, .3761764E-04,
     *            .6546772E+01/
      DATA (C( 3,N),N=1,9)/
     *           -.2036885E-03, .7703042E+00,-.4886267E-05,-.1998629E-02,
     *            .4499598E+00, .3620095E-07, .4692196E-02, .3751913E-04,
     *            .6544648E+01/
      DATA (C( 4,N),N=1,9)/
     *           -.2051946E-03, .7697646E+00,-.4861205E-05,-.1990901E-02,
     *            .4502325E+00, .3591934E-07, .4664216E-02, .3737920E-04,
     *            .6541624E+01/
      DATA (C( 5,N),N=1,9)/
     *           -.1632162E-03, .7810057E+00,-.5277472E-05,-.2322824E-02,
     *            .7159905E+00, .4943163E-07, .5148399E-02, .3859627E-04,
     *            .6612560E+01/
      DATA (C( 6,N),N=1,9)/
     *           -.1664980E-03, .7799287E+00,-.5223968E-05,-.2305658E-02,
     *            .6964292E+00, .4903556E-07, .5108021E-02, .3838043E-04,
     *            .6607319E+01/
      DATA (C( 7,N),N=1,9)/
     *           -.1440378E-03, .7842682E+00,-.5519369E-05,-.2473774E-02,
     *            .5910575E+00, .5793752E-07, .6154590E-02, .3926047E-04,
     *            .6638471E+01/
      DATA (C( 8,N),N=1,9)/
     *           -.1336783E-03, .7860867E+00,-.5672235E-05,-.2557069E-02,
     *            .3528398E+00, .6399606E-07, .7004312E-02, .3962390E-04,
     *            .6655343E+01/
      DATA (C( 9,N),N=1,9)/
     *           -.1357195E-03, .7848000E+00,-.5628947E-05,-.2554850E-02,
     *            .1057585E+00, .6555468E-07, .7215849E-02, .3964243E-04,
     *            .6652038E+01/
      DATA (C(10,N),N=1,9)/
     *           -.1348395E-03, .7842621E+00,-.5657959E-05,-.2578544E-02,
     *           -.6064741E+00, .7002210E-07, .7931478E-02, .3974567E-04,
     *            .6657217E+01/
      DATA (C(11,N),N=1,9)/
     *           -.1472340E-03, .7804559E+00,-.5461105E-05,-.2512906E-02,
     *           -.2829495E+01, .6901463E-07, .7878454E-02, .3931736E-04,
     *            .6640514E+01/
      DATA (C(12,N),N=1,9)/
     *           -.1619146E-03, .7759107E+00,-.5224071E-05,-.2433677E-02,
     *           -.4372853E+01, .6757406E-07, .7827411E-02, .3881627E-04,
     *            .6620347E+01/
      DATA (C(13,N),N=1,9)/
     *           -.1789724E-03, .7705813E+00,-.4943833E-05,-.2339711E-02,
     *           -.6454825E+01, .6554043E-07, .7786706E-02, .3824815E-04,
     *            .6596461E+01/
      DATA (C(14,N),N=1,9)/
     *           -.1984058E-03, .7644511E+00,-.4619129E-05,-.2230396E-02,
     *           -.9215148E+01, .6271601E-07, .7769342E-02, .3762998E-04,
     *            .6568810E+01/
      DATA (C(15,N),N=1,9)/
     *           -.2240840E-03, .7565406E+00,-.4162414E-05,-.2069851E-02,
     *           -.9645579E+01, .5543124E-07, .7290867E-02, .3652558E-04,
     *            .6526380E+01/
```

DATA FOR MODEL 3:

```
&CONTRL NFIRST=1,NLAST=100,NNERGY=50,NTSI=01,NMIX=20,NA=1,&END
&EDDY AMF=1.E9,AHF=2.E7,FKPMF=1.0,FKPHF=1.0,&END
&TSTEPS DTTSF=172800.,DTUVF=7200.,DTSFF=7200.,&END
&PARMS MXSCAN=100,SORF=1.60,CRITF=1.E8,ACORF=.6,&END
&TSPROF TINITF=4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,4.,.0349,.0349,.0349,
.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,.0349,&END
&IBOX ISIS=2,14,IEIS=61,19,JSIS=3,7,JEIS=5,12,&END
```

```
&CONTRL
NFIRST   =  1,
NLAST    =  100,
NNERGY   =  50,
NMIX     =  20,
NWRITE   =  1000000,
NDW      =  1000000,
NTSI     =  1,
NA       =  0,
NB       =  0,
NC       =  0,
&END

&EDDY
AMF      =  .1E+10,
AHF      =  .2E+08,
FKPMF    =  .1E+01,
FKPHF    =  .1E+01,
&END

&TSTEPS
DTTSF    =  .1728E+06,
DTUVF    =  .72E+04,
DTSFF    =  .72E+04,
&END

&PARMS
ACORF    =  .6E+00,
MXSCAN   =  100,
SORF     =  .16E+01,
CRITF    =  .1E+09,
&END

 GRID BOX THICKNESS  'DZ'
   .30000E+04   .46300E+04   .67450E+04   .94460E+04   .12846E+05   .17057E+05   .22191E+05   .28351E+05   .35621E+05   .44057E+05
   .53678E+05   .64454E+05   .76300E+05   .89071E+05   .10255E+06

 GRID POINT SEPARATION  'DZZ'
   .15000E+04   .38150E+04   .56875E+04   .80955E+04   .11146E+05   .14952E+05   .19624E+05   .25271E+05   .31986E+05   .39839E+05
   .48868E+05   .59066E+05   .70377E+05   .82686E+05   .95812E+05   .51277E+05

 DEPTH OF BOX BOTTOM  'ZDZ'
   .30000E+04   .76300E+04   .14375E+05   .23821E+05   .36667E+05   .53724E+05   .75915E+05   .10427E+06   .13989E+06   .18394E+06
   .23762E+06   .30208E+06   .37838E+06   .46745E+06   .57000E+06

 DEPTH OF GRID POINT  'ZDZZ'
   .15000E+04   .53150E+04   .11003E+05   .19098E+05   .30244E+05   .45196E+05   .64820E+05   .90091E+05   .12208E+06   .16192E+06
   .21078E+06   .26985E+06   .34023E+06   .42291E+06   .51872E+06   .57000E+06

 LATITUDE OF T,S POINTS (RADIANS)  'PHIT'
  -.16232E+01  -.15184E+01  -.14137E+01  -.13090E+01  -.12043E+01  -.10996E+01  -.99484E+00  -.89012E+00  -.78540E+00  -.68068E+00
  -.57596E+00  -.47124E+00  -.36652E+00  -.26180E+00  -.15708E+00  -.52360E-01   .52360E-01   .15708E+00   .26180E+00   .36652E+00
   .47124E+00   .57596E+00   .68068E+00   .78540E+00   .89012E+00   .99484E+00   .10996E+01   .12043E+01   .13090E+01   .14137E+01
   .15184E+01   .16232E+01

 LATITUDE OF U,V POINTS (RADIANS)  'PHI'
  -.15708E+01  -.14661E+01  -.13614E+01  -.12566E+01  -.11519E+01  -.10472E+01  -.94248E+00  -.83776E+00  -.73304E+00  -.62832E+00
  -.52360E+00  -.41888E+00  -.31416E+00  -.20944E+00  -.10472E+00  -.91482E-13   .10472E+00   .20944E+00   .31416E+00   .41888E+00
   .52360E+00   .62832E+00   .73304E+00   .83776E+00   .94248E+00   .10472E+01   .11519E+01   .12566E+01   .13614E+01   .14661E+01
   .15708E+01   .16755E+01

 COSINE OF T,S LATITUDE  'CST'
  -.52336E-01   .52336E-01   .15643E+00   .25882E+00   .35837E+00   .45399E+00   .54464E+00   .62932E+00   .70711E+00   .77715E+00
   .83867E+00   .89101E+00   .93358E+00   .96593E+00   .98769E+00   .99863E+00   .99863E+00   .98769E+00   .96593E+00   .93358E+00
   .89101E+00   .83867E+00   .77715E+00   .70711E+00   .62932E+00   .54464E+00   .45399E+00   .35837E+00   .25882E+00   .15643E+00
   .52336E-01  -.52336E-01

 COSINE OF U,V LATITUDE  'CS'
   .13349E-07   .10453E+00   .20791E+00   .30902E+00   .40674E+00   .50000E+00   .58779E+00   .66913E+00   .74314E+00   .80902E+00
   .86603E+00   .91355E+00   .95106E+00   .97815E+00   .99452E+00   .10000E+01   .99452E+00   .97815E+00   .95106E+00   .91355E+00
   .86603E+00   .80902E+00   .74314E+00   .66913E+00   .58779E+00   .50000E+00   .40674E+00   .30902E+00   .20791E+00   .10453E+00
   .13349E-07  -.10453E+00

 SINE OF U,V LATITUDE  'SINE'
  -.10000E+01  -.99452E+00  -.97815E+00  -.95106E+00  -.91355E+00  -.86603E+00  -.80902E+00  -.74314E+00  -.66913E+00  -.58779E+00
  -.50000E+00  -.40674E+00  -.30902E+00  -.20791E+00  -.10453E+00  -.91482E-13   .10453E+00   .20791E+00   .30902E+00   .40674E+00
   .50000E+00   .58779E+00   .66913E+00   .74314E+00   .80902E+00   .86603E+00   .91355E+00   .95106E+00   .97815E+00   .99452E+00
   .10000E+01   .99452E+00
```

START AND END INDICES FOR FOURIER FILTERING

FILTERING INDICES FOR T,S

INDICES FOR ROW 31

1	2	61	0	0	0	0
2	2	61	0	0	0	0
3	2	61	0	0	0	0
4	2	61	0	0	0	0
5	2	61	0	0	0	0
6	2	61	0	0	0	0
7	2	61	0	0	0	0
8	2	61	0	0	0	0
9	2	61	0	0	0	0
10	2	61	0	0	0	0
11	2	61	0	0	0	0
12	2	61	0	0	0	0
13	2	61	0	0	0	0
14	2	61	0	0	0	0
15	2	61	0	0	0	0

INDICES FOR ROW 30

1	2	61	0	0	0	0
2	2	61	0	0	0	0
3	2	61	0	0	0	0
4	2	61	0	0	0	0
5	2	61	0	0	0	0
6	2	61	0	0	0	0
7	2	61	0	0	0	0
8	2	61	0	0	0	0
9	2	61	0	0	0	0
10	2	61	0	0	0	0
11	2	61	0	0	0	0
12	2	61	0	0	0	0
13	2	61	0	0	0	0
14	2	61	0	0	0	0
15	2	61	0	0	0	0

INDICES FOR ROW 29

1	2	61	0	0	0	0
2	2	61	0	0	0	0
3	2	61	0	0	0	0
4	2	61	0	0	0	0
5	2	61	0	0	0	0
6	2	61	0	0	0	0
7	2	61	0	0	0	0
8	2	61	0	0	0	0
9	2	61	0	0	0	0
10	2	61	0	0	0	0
11	2	61	0	0	0	0
12	2	61	0	0	0	0
13	2	61	0	0	0	0
14	2	61	0	0	0	0
15	2	61	0	0	0	0

INDICES FOR ROW 28

1	2	61	0	0	0	0
2	2	61	0	0	0	0
3	2	61	0	0	0	0
4	2	61	0	0	0	0
5	2	61	0	0	0	0
6	2	61	0	0	0	0
7	2	61	0	0	0	0
8	2	61	0	0	0	0
9	2	61	0	0	0	0
10	2	61	0	0	0	0
11	2	61	0	0	0	0
12	2	61	0	0	0	0
13	2	61	0	0	0	0
14	2	61	0	0	0	0
15	2	61	0	0	0	0

INDICES FOR ROW 27

1	46	59	0	0	0	0
2	46	59	0	0	0	0
3	46	59	0	0	0	0
4	46	59	0	0	0	0
5	46	59	0	0	0	0
6	46	59	0	0	0	0
7	46	59	0	0	0	0
8	46	59	0	0	0	0
9	46	59	0	0	0	0

```
   10        46  59    0   0     0   0
   11        46  59    0   0     0   0
   12        46  59    0   0     0   0
   13        46  59    0   0     0   0
   14        46  59    0   0     0   0
   15        46  59    0   0     0   0
INDICES FOR ROW  26
    1        11  33   46  59     0   0
    2        11  33   46  59     0   0
    3        11  33   46  59     0   0
    4        11  33   46  59     0   0
    5        11  33   46  59     0   0
    6        11  33   46  59     0   0
    7        11  33   46  59     0   0
    8        11  33   46  59     0   0
    9        11  33   46  59     0   0
   10        11  33   46  59     0   0
   11        11  33   46  59     0   0
   12        11  33   46  59     0   0
   13        11  33   46  59     0   0
   14        11  33   46  59     0   0
   15        11  33   46  59     0   0
INDICES FOR ROW   7
    1         2  61    0   0     0   0
    2         2  61    0   0     0   0
    3         2  61    0   0     0   0
    4         2  61    0   0     0   0
    5         2  61    0   0     0   0
    6         2  61    0   0     0   0
    7         2  61    0   0     0   0
    8         2  61    0   0     0   0
    9         2  61    0   0     0   0
   10         2  61    0   0     0   0
   11         2  61    0   0     0   0
   12         2  61    0   0     0   0
   13         2  61    0   0     0   0
   14         2  61    0   0     0   0
   15         2  61    0   0     0   0
INDICES FOR ROW   6
    1         2  61    0   0     0   0
    2         2  61    0   0     0   0
    3         2  61    0   0     0   0
    4         2  61    0   0     0   0
    5         2  61    0   0     0   0
    6         2  61    0   0     0   0
    7         2  61    0   0     0   0
    8         2  61    0   0     0   0
    9         2  61    0   0     0   0
   10         2  61    0   0     0   0
   11         2  61    0   0     0   0
   12         2  61    0   0     0   0
   13         2  61    0   0     0   0
   14         2  61    0   0     0   0
   15         2  61    0   0     0   0
INDICES FOR ROW   5
    1         2  61    0   0     0   0
    2         2  61    0   0     0   0
    3         2  61    0   0     0   0
    4         2  61    0   0     0   0
    5         2  61    0   0     0   0
    6         2  61    0   0     0   0
    7         2  61    0   0     0   0
    8         2  61    0   0     0   0
    9         2  61    0   0     0   0
   10         2  61    0   0     0   0
   11         2  61    0   0     0   0
   12         2  61    0   0     0   0
   13         2  61    0   0     0   0
   14         2  61    0   0     0   0
   15         2  61    0   0     0   0
INDICES FOR ROW   4
    1         3  61    0   0     0   0
    2         3  61    0   0     0   0
    3         3  61    0   0     0   0
    4         3  61    0   0     0   0
    5         3  61    0   0     0   0
    6         3  61    0   0     0   0
    7         3  61    0   0     0   0
    8         3  61    0   0     0   0
    9         3  61    0   0     0   0
```

```
        10          3  61    0   0    0   0
        11          3  61    0   0    0   0
        12          3  61    0   0    0   0
        13          3  61    0   0    0   0
        14          3  61    0   0    0   0
        15          3  61    0   0    0   0
INDICES FOR ROW   3
         1          3  61    0   0    0   0
         2          3  61    0   0    0   0
         3          3  61    0   0    0   0
         4          3  61    0   0    0   0
         5          3  61    0   0    0   0
         6          3  61    0   0    0   0
         7          3  61    0   0    0   0
         8          3  61    0   0    0   0
         9          3  61    0   0    0   0
        10          3  61    0   0    0   0
        11          3  61    0   0    0   0
        12          3  61    0   0    0   0
        13          3  61    0   0    0   0
        14          3  61    0   0    0   0
        15          3  61    0   0    0   0

FILTERING INDICES FOR U,V
INDICES FOR ROW  31
         1          0   0    0   0    0   0
         2          0   0    0   0    0   0
         3          0   0    0   0    0   0
         4          0   0    0   0    0   0
         5          0   0    0   0    0   0
         6          0   0    0   0    0   0
         7          0   0    0   0    0   0
         8          0   0    0   0    0   0
         9          0   0    0   0    0   0
        10          0   0    0   0    0   0
        11          0   0    0   0    0   0
        12          0   0    0   0    0   0
        13          0   0    0   0    0   0
        14          0   0    0   0    0   0
        15          0   0    0   0    0   0
INDICES FOR ROW  30
         1          2  61    0   0    0   0
         2          2  61    0   0    0   0
         3          2  61    0   0    0   0
         4          2  61    0   0    0   0
         5          2  61    0   0    0   0
         6          2  61    0   0    0   0
         7          2  61    0   0    0   0
         8          2  61    0   0    0   0
         9          2  61    0   0    0   0
        10          2  61    0   0    0   0
        11          2  61    0   0    0   0
        12          2  61    0   0    0   0
        13          2  61    0   0    0   0
        14          2  61    0   0    0   0
        15          2  61    0   0    0   0
INDICES FOR ROW  29
         1          2  61    0   0    0   0
         2          2  61    0   0    0   0
         3          2  61    0   0    0   0
         4          2  61    0   0    0   0
         5          2  61    0   0    0   0
         6          2  61    0   0    0   0
         7          2  61    0   0    0   0
         8          2  61    0   0    0   0
         9          2  61    0   0    0   0
        10          2  61    0   0    0   0
        11          2  61    0   0    0   0
        12          2  61    0   0    0   0
        13          2  61    0   0    0   0
        14          2  61    0   0    0   0
        15          2  61    0   0    0   0
INDICES FOR ROW  28
         1          2  61    0   0    0   0
         2          2  61    0   0    0   0
         3          2  61    0   0    0   0
         4          2  61    0   0    0   0
         5          2  61    0   0    0   0
```

```
        6         2  61     0   0     0   0
        7         2  61     0   0     0   0
        8         2  61     0   0     0   0
        9         2  61     0   0     0   0
       10         2  61     0   0     0   0
       11         2  61     0   0     0   0
       12         2  61     0   0     0   0
       13         2  61     0   0     0   0
       14         2  61     0   0     0   0
       15         2  61     0   0     0   0
INDICES FOR ROW  27
        1        46  58     0   0     0   0
        2        46  58     0   0     0   0
        3        46  58     0   0     0   0
        4        46  58     0   0     0   0
        5        46  58     0   0     0   0
        6        46  58     0   0     0   0
        7        46  58     0   0     0   0
        8        46  58     0   0     0   0
        9        46  58     0   0     0   0
       10        46  58     0   0     0   0
       11        46  58     0   0     0   0
       12        46  58     0   0     0   0
       13        46  58     0   0     0   0
       14        46  58     0   0     0   0
       15        46  58     0   0     0   0
INDICES FOR ROW   7
        1        46  73    19  32     0   0
        2        46  73    19  32     0   0
        3        46  73    19  32     0   0
        4        46  73    19  32     0   0
        5        46  73    19  32     0   0
        6        46  73    19  32     0   0
        7        46  73    19  32     0   0
        8        46  73    19  32     0   0
        9        46  73    19  32     0   0
       10        46  73    19  32     0   0
       11        46  73    19  32     0   0
       12        46  73    19  32     0   0
       13        46  73    19  32     0   0
       14        46  73    19  32     0   0
       15        46  73    19  32     0   0
INDICES FOR ROW   6
        1         2  61     0   0     0   0
        2         2  61     0   0     0   0
        3         2  61     0   0     0   0
        4         2  61     0   0     0   0
        5         2  61     0   0     0   0
        6         2  61     0   0     0   0
        7         2  61     0   0     0   0
        8         2  61     0   0     0   0
        9         2  61     0   0     0   0
       10         2  61     0   0     0   0
       11         2  61     0   0     0   0
       12         2  61     0   0     0   0
       13         2  61     0   0     0   0
       14         2  61     0   0     0   0
       15         2  61     0   0     0   0
INDICES FOR ROW   5
        1         2  61     0   0     0   0
        2         2  61     0   0     0   0
        3         2  61     0   0     0   0
        4         2  61     0   0     0   0
        5         2  61     0   0     0   0
        6         2  61     0   0     0   0
        7         2  61     0   0     0   0
        8         2  61     0   0     0   0
        9         2  61     0   0     0   0
       10         2  61     0   0     0   0
       11         2  61     0   0     0   0
       12         2  61     0   0     0   0
       13         2  61     0   0     0   0
       14         2  61     0   0     0   0
       15         2  61     0   0     0   0
INDICES FOR ROW   4
        1         3  60     0   0     0   0
        2         3  60     0   0     0   0
        3         3  60     0   0     0   0
        4         3  60     0   0     0   0
        5         3  60     0   0     0   0
```

6	3	60	0	0	0	0
7	3	60	0	0	0	0
8	3	60	0	0	0	0
9	3	60	0	0	0	0
10	3	60	0	0	0	0
11	3	60	0	0	0	0
12	3	60	0	0	0	0
13	3	60	0	0	0	0
14	3	60	0	0	0	0
15	3	60	0	0	0	0

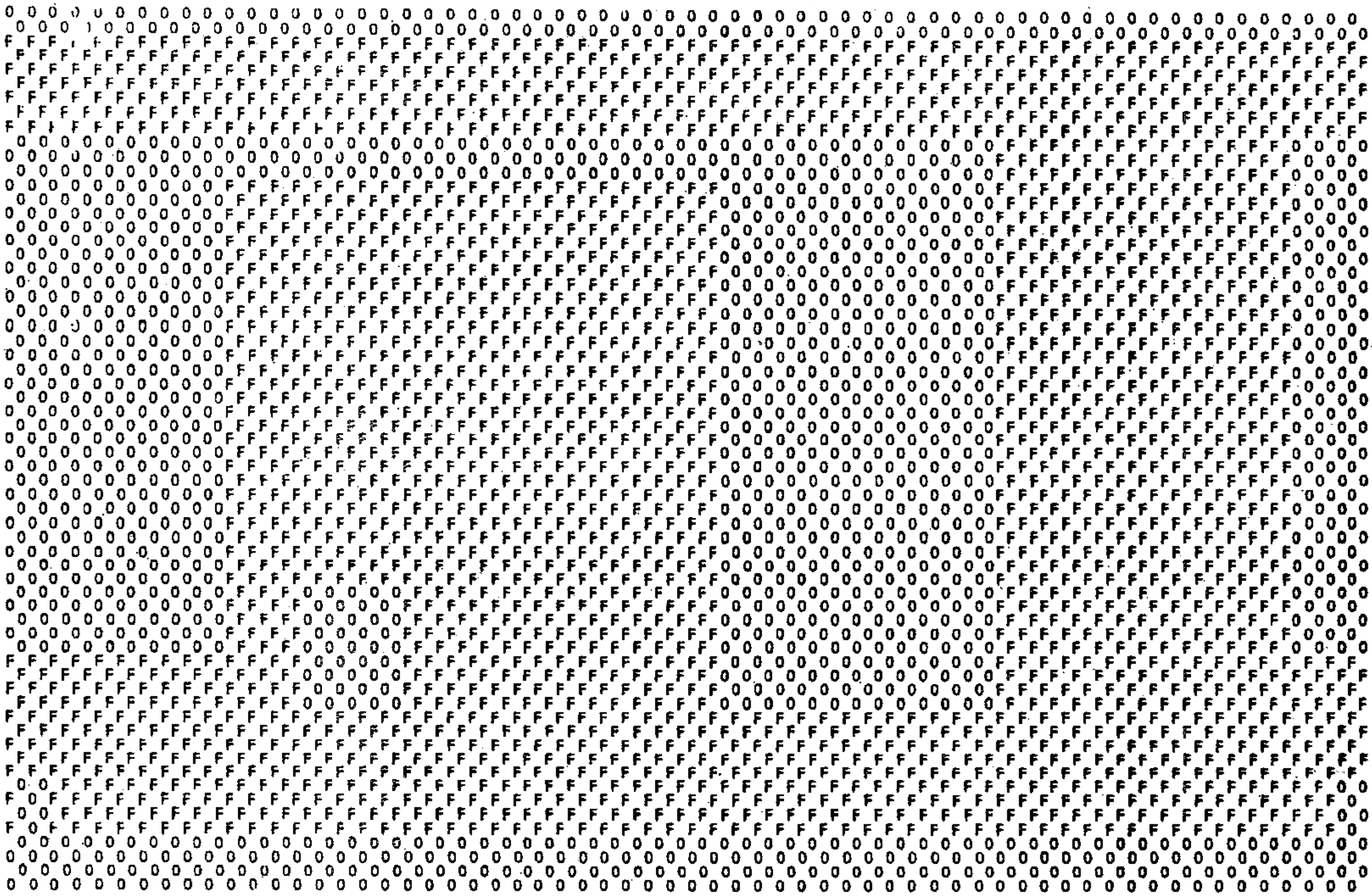
INDICES FOR ROW 3

1	3	60	0	0	0	0
2	3	60	0	0	0	0
3	3	60	0	0	0	0
4	3	60	0	0	0	0
5	3	60	0	0	0	0
6	3	60	0	0	0	0
7	3	60	0	0	0	0
8	3	60	0	0	0	0
9	3	60	0	0	0	0
10	3	60	0	0	0	0
11	3	60	0	0	0	0
12	3	60	0	0	0	0
13	3	60	0	0	0	0
14	3	60	0	0	0	0
15	3	60	0	0	0	0

FILTERING INDICES FOR STREAM FUNCTION

31	0	0	0	0	0	0
30	2	61	0	0	0	0
29	2	61	0	0	0	0
28	47	58	0	0	0	0
27	47	58	0	0	0	0
7	47	73	20	32	0	0
6	2	61	0	0	0	0
5	4	60	0	0	0	0
4	4	60	0	0	0	0
3	0	0	0	0	0	0

NUMBER OF LEVELS AT T,S POINTS AND U,V POINTS



SURFACE AREA = 3.367699E+18   VOLUME = 1.919588E+24

START AND END INDICES FOR STREAM FUNCTION

J= 32	0	0	0	0	0	0
J= 31	0	0	0	0	0	0
J= 30	2	61	0	0	0	0
J= 29	2	61	0	0	0	0
J= 28	47	58	0	0	0	0
J= 27	47	58	0	0	0	0
J= 26	47	58	0	0	0	0
J= 25	12	32	47	58	0	0
J= 24	12	32	47	58	0	0
J= 23	12	32	47	58	0	0
J= 22	12	32	47	58	0	0
J= 21	12	32	47	58	0	0
J= 20	12	32	47	58	0	0
J= 19	12	32	47	58	0	0
J= 18	12	32	47	58	0	0
J= 17	12	32	47	58	0	0
J= 16	12	32	47	58	0	0
J= 15	12	32	47	58	0	0
J= 14	12	32	47	58	0	0
J= 13	12	32	47	58	0	0
J= 12	12	13	20	32	47	58
J= 11	12	13	20	32	47	58
J= 10	12	13	20	32	47	58
J= 9	12	13	20	32	47	58
J= 8	2	13	20	32	47	61
J= 7	2	13	20	32	47	61
J= 6	2	61	0	0	0	0
J= 5	4	60	0	0	0	0
J= 4	4	60	0	0	0	0
J= 3	0	0	0	0	0	0
J= 2	0	0	0	0	0	0
J= 1	0	0	0	0	0	0

```
&TSPROF
TINITF = 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000
4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000
0.0349000000000 0.0349000000000 4.0000000000000 4.0000000000000 4.0000000000000 4.0000000000000 0.0349000000000 0.0349000000000 0.0349000000000
0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000 0.0349000000000
&END

&IBOX
ISIS = 2 14
IEIS = 61 19
JSIS = 3 7
JEIS = 5 12
&END
```

TS=	1	YEAR=	0.01	DAY=	2.0	ENERGY= 1.830949E-03	DTEMP=	1.440491E-08	DSALT=	2.979225E-08	SCANS=	100
TS=	2	YEAR=	0.01	DAY=	4.0	ENERGY= 1.830961E-03	DTEMP=	1.440492E-08	DSALT=	1.489656E-08	SCANS=	100
TS=	3	YEAR=	0.02	DAY=	6.0	ENERGY= 4.512782E-03	DTEMP=	1.385325E-08	DSALT=	8.436415E-13	SCANS=	100
TS=	4	YEAR=	0.02	DAY=	8.0	ENERGY= 8.125579E-03	DTEMP=	1.330269E-08	DSALT=	8.179236E-13	SCANS=	100
TS=	5	YEAR=	0.03	DAY=	10.0	ENERGY= 9.180787E-03	DTEMP=	1.281694E-08	DSALT=	7.966034E-13	SCANS=	100
TS=	6	YEAR=	0.03	DAY=	12.0	ENERGY= 8.780523E-03	DTEMP=	1.233161E-08	DSALT=	7.753330E-13	SCANS=	100
TS=	7	YEAR=	0.04	DAY=	14.0	ENERGY= 7.052399E-03	DTEMP=	1.190115E-08	DSALT=	7.570214E-13	SCANS=	100
TS=	8	YEAR=	0.04	DAY=	16.0	ENERGY= 4.613504E-03	DTEMP=	1.147078E-08	DSALT=	7.387211E-13	SCANS=	100
TS=	9	YEAR=	0.05	DAY=	18.0	ENERGY= 3.468817E-03	DTEMP=	1.108807E-08	DSALT=	7.227303E-13	SCANS=	100
TS=	10	YEAR=	0.05	DAY=	20.0	ENERGY= 3.287694E-03	DTEMP=	1.070533E-08	DSALT=	7.067474E-13	SCANS=	100
TS=	11	YEAR=	0.06	DAY=	22.0	ENERGY= 4.271193E-03	DTEMP=	1.036432E-08	DSALT=	6.926709E-13	SCANS=	100
TS=	12	YEAR=	0.07	DAY=	24.0	ENERGY= 6.061125E-03	DTEMP=	1.002329E-08	DSALT=	6.786050E-13	SCANS=	100
TS=	13	YEAR=	0.07	DAY=	26.0	ENERGY= 7.336437E-03	DTEMP=	9.719095E-09	DSALT=	6.661522E-13	SCANS=	100
TS=	14	YEAR=	0.08	DAY=	28.0	ENERGY= 8.251972E-03	DTEMP=	9.414955E-09	DSALT=	6.537078E-13	SCANS=	25
TS=	15	YEAR=	0.08	DAY=	30.0	ENERGY= 8.354130E-03	DTEMP=	9.143369E-09	DSALT=	6.426440E-13	SCANS=	100
TS=	16	YEAR=	0.09	DAY=	32.0	ENERGY= 7.868166E-03	DTEMP=	8.871876E-09	DSALT=	6.315820E-13	SCANS=	100
TS=	17	YEAR=	0.09	DAY=	34.0	ENERGY= 7.545134E-03	DTEMP=	8.629160E-09	DSALT=	6.217255E-13	SCANS=	100
TS=	18	YEAR=	0.10	DAY=	36.0	ENERGY= 7.255357E-03	DTEMP=	8.389649E-09	DSALT=	6.123026E-13	SCANS=	30
TS=	19	YEAR=	0.10	DAY=	38.0	ENERGY= 7.559796E-03	DTEMP=	8.172571E-09	DSALT=	6.035242E-13	SCANS=	100
TS=	20	YEAR=	0.11	DAY=	40.0	ENERGY= 8.153329E-03	DTEMP=	7.961561E-09	DSALT=	5.956410E-13	SCANS=	100
TS=	21	YEAR=	0.11	DAY=	42.0	ENERGY= 9.670873E-03	DTEMP=	7.660943E-09	DSALT=	9.671667E-13	SCANS=	35
TS=	22	YEAR=	0.12	DAY=	44.0	ENERGY= 9.666635E-03	DTEMP=	7.566002E-09	DSALT=	5.790599E-13	SCANS=	35
TS=	23	YEAR=	0.13	DAY=	46.0	ENERGY= 1.017377E-02	DTEMP=	7.399728E-09	DSALT=	5.760476E-13	SCANS=	43
TS=	24	YEAR=	0.13	DAY=	48.0	ENERGY= 1.052778E-02	DTEMP=	7.211548E-09	DSALT=	5.643693E-13	SCANS=	24
TS=	25	YEAR=	0.14	DAY=	50.0	ENERGY= 1.079861E-02	DTEMP=	7.061599E-09	DSALT=	5.606338E-13	SCANS=	23
TS=	26	YEAR=	0.14	DAY=	52.0	ENERGY= 1.103873E-02	DTEMP=	6.899513E-09	DSALT=	5.520496E-13	SCANS=	23
TS=	27	YEAR=	0.15	DAY=	54.0	ENERGY= 1.139864E-02	DTEMP=	6.763769E-09	DSALT=	5.482285E-13	SCANS=	36
TS=	28	YEAR=	0.15	DAY=	56.0	ENERGY= 1.185845E-02	DTEMP=	6.613832E-09	DSALT=	5.402747E-13	SCANS=	43
TS=	29	YEAR=	0.16	DAY=	58.0	ENERGY= 1.237791E-02	DTEMP=	6.488842E-09	DSALT=	5.362419E-13	SCANS=	31
TS=	30	YEAR=	0.16	DAY=	60.0	ENERGY= 1.294981E-02	DTEMP=	6.357191E-09	DSALT=	5.297991E-13	SCANS=	21
TS=	31	YEAR=	0.17	DAY=	62.0	ENERGY= 1.346911E-02	DTEMP=	6.244119E-09	DSALT=	5.260467E-13	SCANS=	28
TS=	32	YEAR=	0.18	DAY=	64.0	ENERGY= 1.396341E-02	DTEMP=	6.126119E-09	DSALT=	5.204534E-13	SCANS=	18
TS=	33	YEAR=	0.18	DAY=	66.0	ENERGY= 1.442480E-02	DTEMP=	6.023796E-09	DSALT=	5.169874E-13	SCANS=	22
TS=	34	YEAR=	0.19	DAY=	68.0	ENERGY= 1.487493E-02	DTEMP=	5.919190E-09	DSALT=	5.123183E-13	SCANS=	18
TS=	35	YEAR=	0.19	DAY=	70.0	ENERGY= 1.535555E-02	DTEMP=	5.827305E-09	DSALT=	5.092561E-13	SCANS=	24
TS=	36	YEAR=	0.20	DAY=	72.0	ENERGY= 1.586774E-02	DTEMP=	5.730917E-09	DSALT=	5.046438E-13	SCANS=	22
TS=	37	YEAR=	0.20	DAY=	74.0	ENERGY= 1.641082E-02	DTEMP=	5.643640E-09	DSALT=	5.014629E-13	SCANS=	27

TS=	YEAR=	DAY=	ENERGY=	DTEMP=	DSALT=	SCANS=
39	0.21	76.0	1.698616E-02	5.554837E-09	4.972951E-13	23
40	0.21	78.0	1.756352E-02	5.476956E-09	4.944463E-13	27
41	0.22	80.0	1.815381E-02	5.396839E-09	4.907934E-13	23
42	0.22	82.0	1.934687E-02	6.668673E-09	9.990785E-13	34
43	0.23	84.0	1.934696E-02	5.252951E-09	4.849203E-13	34
44	0.24	86.0	1.996211E-02	5.200543E-09	4.869968E-13	33
45	0.24	88.0	2.059532E-02	5.121245E-09	4.795308E-13	28
46	0.25	90.0	2.123867E-02	5.068548E-09	4.798518E-13	23
47	0.25	92.0	2.189634E-02	5.000327E-09	4.745638E-13	19
48	0.26	94.0	2.256099E-02	4.950009E-09	4.741935E-13	22
49	0.26	96.0	2.323486E-02	4.889091E-09	4.699794E-13	17
	0.27	98.0	2.391518E-02	4.841694E-09	4.692750E-13	27

TEMPERATURE FOR J = 7 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
[illegible]	4.80	4.80	4.80	4.80	4.80	4.80	4.80	4.80	4.80	4.80	4.80	4.80	4.80	4.82	4.84	4.84	4.84	4.84	4.82	4.79
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	4.56	4.56	4.56	4.57	[illegible]	[illegible]	4.58	4.58	4.57	4.55
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	4.56	4.56	4.56	4.57	[illegible]	[illegible]	4.58	4.58	4.57	4.55
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	4.29	4.29	4.29	4.29	4.30	4.31	4.31	4.31	4.31	4.30	4.29
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	4.29	4.29	4.29	4.29	4.29	4.30	4.31	4.31	4.31	4.31	4.30	4.29
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	4.17	4.17	4.17	4.17	4.17	4.18	4.18	4.19	4.19	4.18	4.18	4.17
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	4.17	4.17	4.17	4.17	4.17	4.18	4.18	4.19	4.19	4.18	4.18	4.17
8	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.14	4.14	4.14	4.14	4.14	4.14	4.13
9	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.14	4.14	4.14	4.14	4.14	4.14	4.13
10	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
11	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
12	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
13	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
14	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
15	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00

SALINITY FOR J = 7 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.00	0.00
3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.00	0.00
4	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.04	-0.04	-0.04	-0.04	-0.05	-0.05
5	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.04	-0.04	-0.04	-0.04	-0.05	-0.05
6	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07
7	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07
8	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08
9	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08
10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
11	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
12	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
13	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
14	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
15	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10

W VELOCITY FOR J = 7 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.02	0.01	0.02	0.01	0.01	0.01	-0.05	-0.09	-0.09	-0.09	-0.09	-0.03	0.01
3	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.02	0.01	0.02	0.01	0.01	0.01	-0.05	-0.09	-0.09	-0.09	-0.09	-0.03	0.01
4	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.02	0.01	0.02	0.01	0.01	0.01	-0.05	-0.09	-0.09	-0.09	-0.09	-0.02	0.01
5	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.01	0.02	0.01	0.01	0.01	-0.06	-0.09	-0.09	-0.09	-0.09	-0.01	0.01
6	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.01	0.02	0.01	0.01	0.01	-0.06	-0.08	-0.08	-0.08	-0.08	-0.01	0.01
7	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.01	0.01	-0.07	-0.08	-0.08	-0.08	-0.08	0.00	0.01
8	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	-0.07	-0.08	-0.08	-0.08	-0.08	0.00	0.01
9	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	-0.07	-0.07	-0.07	-0.07	-0.07	0.01	0.01
10	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	-0.06	-0.07	-0.07	-0.07	-0.07	0.01	0.01
11	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	-0.05	-0.06	-0.06	-0.06	-0.06	0.01	0.01
12	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	-0.05	-0.05	-0.05	-0.05	-0.05	0.01	0.01
13	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	-0.04	-0.04	-0.04	-0.04	-0.04	0.01	0.01
14	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	-0.03	-0.03	-0.03	-0.03	-0.03	0.00	0.01
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.01	-0.02	-0.02	-0.02	-0.02	0.00	0.00
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

U VELOCITY FOR J = 7 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.53	0.49	0.48	0.46	0.42	0.43	0.38	0.40	0.35	0.36	0.31	0.28	0.20	0.00	0.00	0.00	0.00	0.00	0.26	0.42
2	0.45	0.43	0.41	0.39	0.37	0.35	0.33	0.31	0.29	0.28	0.25	0.21	0.13	0.00	0.00	0.00	0.00	0.00	0.19	0.34
3	0.43	0.41	0.39	0.37	0.35	0.33	0.31	0.29	0.28	0.26	0.23	0.19	0.11	0.00	0.00	0.00	0.00	0.00	0.17	0.32
4	0.42	0.40	0.38	0.36	0.33	0.31	0.30	0.28	0.26	0.24	0.22	0.18	0.09	0.00	0.00	0.00	0.00	0.00	0.16	0.31
5	0.40	0.39	0.36	0.34	0.32	0.30	0.28	0.27	0.25	0.23	0.21	0.16	0.08	0.00	0.00	0.00	0.00	0.00	0.15	0.29
6	0.39	0.37	0.35	0.33	0.31	0.29	0.27	0.25	0.24	0.22	0.19	0.15	0.07	0.00	0.00	0.00	0.00	0.00	0.13	0.28

7	0.38	0.36	0.34	0.32	0.30	0.28	0.26	0.24	0.23	0.21	0.18	0.14	0.06	0.00	0.00	0.00	0.00	0.00	0.12	0.27
8	0.37	0.35	0.33	0.31	0.29	0.27	0.25	0.23	0.22	0.20	0.18	0.13	0.05	0.00	0.00	0.00	0.00	0.00	0.11	0.26
9	0.37	0.35	0.33	0.31	0.28	0.26	0.25	0.23	0.21	0.19	0.17	0.13	0.05	0.00	0.00	0.00	0.00	0.00	0.11	0.26
10	0.37	0.35	0.33	0.30	0.28	0.26	0.25	0.23	0.21	0.19	0.17	0.12	0.04	0.00	0.00	0.00	0.00	0.00	0.11	0.25
11	0.37	0.35	0.33	0.30	0.28	0.26	0.25	0.23	0.21	0.19	0.17	0.12	0.04	0.00	0.00	0.00	0.00	0.00	0.11	0.25
12	0.37	0.35	0.33	0.30	0.28	0.26	0.25	0.23	0.21	0.19	0.17	0.12	0.04	0.00	0.00	0.00	0.00	0.00	0.11	0.25
13	0.37	0.35	0.33	0.30	0.28	0.26	0.25	0.23	0.21	0.19	0.17	0.12	0.04	0.00	0.00	0.00	0.00	0.00	0.11	0.25
14	0.37	0.35	0.33	0.30	0.28	0.26	0.25	0.23	0.21	0.19	0.17	0.12	0.04	0.00	0.00	0.00	0.00	0.00	0.11	0.25
15	0.37	0.35	0.33	0.30	0.28	0.26	0.25	0.23	0.21	0.19	0.17	0.12	0.04	0.00	0.00	0.00	0.00	0.00	0.11	0.25

V VELOCITY FOR J = 7 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	2.13	2.12	2.12	2.12	2.11	2.13	2.10	2.13	2.08	2.12	2.07	2.12	2.09	0.00	0.00	0.00	0.00	0.00	2.56	2.38
2	-0.06	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.09	-0.09	-0.08	-0.08	0.00	0.00	0.00	0.00	0.00	0.40	0.16
3	-0.06	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.09	-0.09	-0.08	-0.08	0.00	0.00	0.00	0.00	0.00	0.40	0.16
4	-0.06	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.09	-0.09	-0.08	-0.08	0.00	0.00	0.00	0.00	0.00	0.40	0.16
5	-0.06	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.09	-0.09	-0.08	-0.08	0.00	0.00	0.00	0.00	0.00	0.40	0.16
6	-0.06	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.09	-0.09	-0.08	-0.08	0.00	0.00	0.00	0.00	0.00	0.40	0.16
7	-0.06	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16
8	-0.06	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16
9	-0.05	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16
10	-0.05	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16
11	-0.05	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16
12	-0.05	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16
13	-0.05	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16
14	-0.05	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16
15	-0.05	-0.06	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.08	-0.08	-0.09	-0.08	-0.07	0.00	0.00	0.00	0.00	0.00	0.40	0.16

TEMPERATURE FOR J = 14 AT TIMESTEP 50

	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	16.18	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31	16.31
2	6.99	7.12	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13	7.13
3	4.24	4.26	4.25	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26	4.26
4	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01
5	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
6	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
7	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
8	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
9	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
10	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
11	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
12	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
13	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
14	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
15	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00

SALINITY FOR J = 14 AT TIMESTEP 50

	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16
2	-0.04	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03
3	-0.10	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09
4	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
5	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
6	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
7	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
8	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
9	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
11	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
12	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
13	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
14	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
15	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10

W VELOCITY FOR J = 14 AT TIMESTEP 50

	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
3	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
7	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

8	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
11	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
14	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

U VELOCITY FOR J = 14 AT TIMESTEP 50

	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
1	0.00	0.84	0.93	0.93	0.93	0.92	0.91	0.90	0.90	0.90	0.90	0.91	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.93
2	0.00	0.10	0.13	0.13	0.12	0.12	0.11	0.10	0.09	0.09	0.10	0.11	0.11	0.11	0.11	0.11	0.11	0.12	0.12	0.12
3	0.00	0.00	0.00	0.00	0.00	-0.01	-0.02	-0.02	-0.03	-0.03	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	0.00	0.00
4	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
5	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
6	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
7	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
8	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
9	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
10	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
11	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
12	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
13	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
14	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01
15	0.00	-0.01	-0.02	-0.01	-0.01	-0.02	-0.03	-0.04	-0.04	-0.04	-0.04	-0.03	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.01

V VELOCITY FOR J = 14 AT TIMESTEP 50

	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
1	0.00	-1.15	-0.96	-0.93	-0.92	-0.93	-0.94	-0.94	-0.94	-0.94	-0.93	-0.94	-0.94	-0.94	-0.94	-0.94	-0.94	-0.94	-0.94	-0.95
2	0.00	-0.10	0.02	0.05	0.06	0.05	0.04	0.04	0.04	0.04	0.05	0.04	0.03	0.04	0.04	0.04	0.04	0.04	0.04	0.03
3	0.00	-0.08	-0.01	0.03	0.03	0.03	0.02	0.01	0.01	0.02	0.02	0.01	0.01	0.01	0.01	0.02	0.02	0.01	0.01	0.01
4	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.01	0.00
5	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
6	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
7	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
8	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
9	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
10	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
11	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
12	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
13	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
14	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00
15	0.00	-0.07	-0.01	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00

TEMPERATURE FOR J = 21 AT TIMESTEP 50

	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	13.96	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03	14.03
2	6.48	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57	6.57
3	4.20	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22	4.22
4	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01
5	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
6	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
7	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
8	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
9	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
10	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
11	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
12	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
13	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
14	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
15	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00

SALINITY FOR J = 21 AT TIMESTEP 50

	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31
2	0.00	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09
4	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
5	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
6	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
7	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10

8	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
9	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
11	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
12	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
13	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
14	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
15	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10

W VELOCITY FOR J = 21 AT TIMESTEP 50

	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
3	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
4	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
5	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
6	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
7	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
8	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
9	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
10	0.00	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
11	0.00	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
12	0.00	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
13	0.00	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
14	0.00	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

U VELOCITY FOR J = 21 AT TIMESTEP 50

	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
1	0.00	0.12	0.07	0.05	0.06	0.07	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.09	0.09	0.10	0.11	0.11	0.12	0.13
2	0.00	0.01	-0.04	-0.06	-0.06	-0.05	-0.04	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.02	-0.01	-0.01	0.00	0.01	0.01
3	0.00	-0.03	-0.07	-0.09	-0.09	-0.08	-0.07	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.05	-0.05	-0.04	-0.03	-0.03	-0.02
4	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.05	-0.05	-0.04	-0.04	-0.03	-0.02
5	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
6	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
7	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
8	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
9	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
10	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
11	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
12	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
13	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
14	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02
15	0.00	-0.03	-0.07	-0.10	-0.09	-0.08	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.06	-0.06	-0.06	-0.05	-0.04	-0.04	-0.03	-0.02

V VELOCITY FOR J = 21 AT TIMESTEP 50

	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
1	0.00	0.30	-0.19	-0.26	-0.22	-0.19	-0.17	-0.17	-0.17	-0.18	-0.18	-0.17	-0.17	-0.17	-0.17	-0.17	-0.17	-0.17	-0.18	-0.18
2	0.00	0.46	-0.03	-0.10	-0.06	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	0.00	0.00	-0.01	-0.01	-0.01	-0.01	-0.01
3	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
4	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
5	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
6	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
7	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
8	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
9	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
10	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
11	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
12	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
13	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
14	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02
15	0.00	0.45	-0.03	-0.10	-0.06	-0.03	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02

TEMPERATURE FOR J = 28 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39	4.39
2	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27
3	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27
4	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13
5	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13	4.13
6	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07
7	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07	4.07

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
8	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03
9	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03	4.03
10	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02
11	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02	4.02
12	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01
13	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01
14	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01
15	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01	4.01

SALINITY FOR J = 28 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08
2	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
3	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
4	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04
5	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04
6	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07
7	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07	-0.07
8	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09
9	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09
10	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09
11	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09	-0.09
12	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
13	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
14	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10
15	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10	-0.10

W VELOCITY FOR J = 28 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06
3	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06
4	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06
5	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
6	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
7	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
8	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
9	-0.05	-0.04	-0.04	-0.05	-0.04	-0.04	-0.04	-0.04	-0.04	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
10	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04
11	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04
12	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04	-0.04
13	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03	-0.03
14	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02	-0.02
15	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

U VELOCITY FOR J = 28 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.40	0.39	0.39	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38
2	0.37	0.37	0.37	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36
3	0.36	0.36	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35
4	0.35	0.34	0.34	0.34	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
5	0.33	0.33	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32	0.32
6	0.32	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.30	0.30	0.31	0.31	0.31	0.31	0.31	0.31	0.31
7	0.31	0.30	0.30	0.30	0.30	0.30	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
8	0.30	0.29	0.29	0.29	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28
9	0.29	0.28	0.28	0.28	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
10	0.28	0.27	0.27	0.27	0.27	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
11	0.27	0.26	0.26	0.26	0.26	0.26	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
12	0.26	0.26	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
13	0.25	0.25	0.25	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24
14	0.25	0.24	0.24	0.24	0.24	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23
15	0.24	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23

V VELOCITY FOR J = 28 AT TIMESTEP 50

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	-1.47	-1.47	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48	-1.48
2	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
4	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
5	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
6	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
7	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01

8	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
9	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
10	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
11	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
12	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
13	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
14	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
15	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01

TS= 50 YEAR= 0.27 DAY=100.0 ENERGY= 2.460674E-02 DTEMP= 4.786561E-09 DSALT= 4.657365E-13 SCANS= 20

WORK BY:	INTERNAL MODE	EXTERNAL MODE
TIME RATE OF CHANGE	8.387710E-09	9.124575E-08
HORIZONTAL ADVECTION	-3.083105E-11	-1.279995E-13
VERTICAL ADVECTION	3.095905E-11	-8.095728E-26
HORIZONTAL FRICTION	-4.805119E-09	-7.330825E-09
VERTICAL FRICTION	-1.269721E-09	-3.990400E-26
PRESSURE FORCES	-3.733157E-08	7.419375E-21
IMPLICIT EFFECTS	-4.533570E-13	4.836073E-10
WORK BY WIND	5.224735E-08	9.809310E-08
BOTTOM DRAG	0.000000E+00	0.000000E+00
WORK BY BUOYANCY FORCES	-3.733157E-08	

	TEMPERATURE	SALINITY
TIME RATE OF CHANGE	4.763982E-09	4.657089E-13
HORIZONTAL ADVECTION	-1.229421E-21	-8.147096E-26
VERTICAL ADVECTION	-1.953332E-23	-4.230222E-28
HORIZONTAL DIFFUSION	-1.283268E-22	-1.213918E-26
SURFACE FLUX	-2.244262E-23	-6.521671E-28
TRUNCATION ERROR	4.763982E-09	4.657089E-13
CHANGE OF VARIANCE	7.639183E-08	-1.252264E-17

ENERGY CONVERSION ERROR -1.131432E-18 NONLINEAR EXCHANGE ERROR -8.399940E-23

NORTHWARD TRANSPORT OF HEAT (X10**15 WATTS)

	X MEAN	X EDDY	Z MEAN	Z EDDY	EKMAN	TOT ADV	DIFFUS	TOTAL
30	0.000	0.000	0.000	0.000	0.000	0.000	0.004	0.004
29	-0.001	0.000	0.000	0.003	-0.001	-0.001	0.009	0.008
28	-0.006	0.000	0.000	0.003	-0.006	-0.006	0.013	0.007
27	-0.004	0.000	0.000	0.000	-0.004	-0.004	0.004	0.000
26	-0.008	0.000	0.000	0.001	-0.009	-0.008	0.004	-0.004
25	-0.111	0.000	0.000	0.005	-0.116	-0.111	-0.014	-0.125
24	-0.206	0.000	0.000	0.016	-0.222	-0.206	0.011	-0.195
23	-0.204	0.000	0.000	0.013	-0.217	-0.204	0.012	-0.192
22	-0.147	0.000	0.000	0.006	-0.153	-0.147	0.013	-0.134
21	-0.038	0.000	0.000	0.005	-0.043	-0.038	0.013	-0.025
20	0.078	0.000	0.000	-0.010	0.089	0.078	0.013	0.092
19	0.275	0.000	0.000	0.060	0.215	0.275	0.015	0.290
18	0.347	0.000	0.000	0.016	0.331	0.347	0.016	0.362
17	0.136	0.000	0.000	-0.406	0.542	0.136	-0.002	0.133
16	0.000	0.000	0.000	*********	*********	0.000	0.000	0.000
15	-0.136	0.000	0.000	0.406	-0.542	-0.136	0.002	-0.133
14	-0.347	0.000	0.000	-0.016	-0.331	-0.347	-0.016	-0.362
13	-0.275	0.000	0.000	-0.060	-0.215	-0.275	-0.015	-0.290
12	-0.078	0.000	0.000	0.010	-0.089	-0.078	-0.013	-0.092
11	0.032	0.000	0.000	-0.005	0.038	0.032	-0.012	0.021
10	0.126	0.000	0.000	-0.011	0.136	0.126	-0.011	0.115
9	0.175	-0.001	0.000	-0.018	0.193	0.175	-0.011	0.164
8	0.248	0.000	0.000	-0.016	0.265	0.248	-0.015	0.234
7	0.134	0.000	0.000	-0.005	0.138	0.133	0.017	0.150
6	0.036	0.000	0.000	0.000	0.037	0.036	-0.019	0.018
5	0.018	0.000	0.000	0.000	0.019	0.018	-0.017	0.002
4	0.006	0.000	0.000	0.000	0.006	0.006	-0.013	-0.007
3	0.001	0.000	0.000	0.000	0.001	0.001	-0.008	-0.008
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

NORTHWARD TRANSPORT OF SALT (X10**10 CM**3/SEC)

X MEAN	X EDDY	Z MEAN	Z EDDY	EKMAN	TOT ADV	DIFFUS	TOTAL
0.001	0.000	0.000	0.001	0.000	0.001	-0.007	-0.007
-0.035	0.000	0.000	0.001	-0.036	-0.035	-0.014	-0.049
-0.092	0.000	0.000	0.002	-0.094	-0.092	-0.017	-0.109
-0.032	0.000	0.000	0.003	-0.036	-0.032	-0.006	-0.038
-0.041	0.000	0.000	0.004	-0.045	-0.041	-0.010	-0.051
-0.364	0.000	0.000	0.017	-0.361	-0.364	-0.111	-0.475
-0.553	0.000	0.000	0.042	-0.595	-0.553	-0.012	-0.565
-0.411	0.000	0.000	0.025	-0.436	-0.411	-0.017	-0.428
-0.222	0.000	0.000	0.009	-0.232	-0.223	-0.020	-0.242
-0.043	0.000	0.000	0.006	-0.049	-0.043	-0.022	-0.065
0.066	0.000	0.000	-0.009	0.074	0.066	-0.023	0.043
0.165	0.000	0.000	0.036	0.129	0.165	-0.023	0.142
0.141	0.000	0.000	0.006	0.135	0.141	-0.024	0.118
0.034	0.000	0.000	-0.101	0.135	0.034	-0.026	0.007
0.000	0.000	0.000	*********	*********	0.000	0.000	0.000
-0.034	0.000	0.000	0.101	-0.135	-0.034	0.026	-0.007
-0.141	0.000	0.000	-0.006	-0.135	-0.141	0.024	-0.118
-0.165	0.000	0.000	-0.036	-0.129	-0.165	0.023	-0.142
-0.066	0.000	0.000	0.009	-0.074	-0.066	0.023	-0.043
0.037	0.000	0.000	-0.007	0.043	0.037	0.019	0.056
0.190	0.000	0.000	-0.016	0.206	0.191	0.018	0.208
0.353	-0.001	0.000	-0.037	0.389	0.352	0.015	0.368
0.666	0.000	0.000	-0.044	0.710	0.666	0.012	0.677
0.438	-0.001	0.000	-0.016	0.453	0.438	0.135	0.572
0.190	0.000	0.000	-0.001	0.192	0.190	0.038	0.229
0.149	0.000	0.000	-0.003	0.152	0.149	0.026	0.175
0.089	0.000	0.000	-0.004	0.093	0.089	0.020	0.109
0.034	0.000	0.000	-0.002	0.036	0.034	0.013	0.047
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

MERIDIONAL MASS TRANSPORT

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.00	0.00	1.89	5.29	9.60	13.27	10.72	10.08	5.87	3.63	0.83	-1.76	-5.58	-6.52	-2.80	0.00	2.80	6.52	5.58	1.76
2	0.00	0.00	1.87	5.23	9.50	13.14	10.62	9.99	5.81	3.59	0.80	-1.79	-5.67	-6.19	-0.99	0.00	0.99	6.19	5.67	1.79
3	0.00	0.00	1.83	5.15	9.35	12.95	10.47	9.86	5.74	3.55	0.78	-1.78	-5.62	-6.07	-0.57	0.00	0.57	6.07	5.62	1.78
4	0.00	0.00	1.78	5.03	9.15	12.68	10.26	9.70	5.64	3.48	0.77	-1.75	-5.53	-5.96	-0.52	0.00	0.52	5.96	5.53	1.75
5	0.00	0.00	1.72	4.87	8.88	12.32	9.99	9.47	5.51	3.40	0.75	-1.71	-5.40	-5.82	-0.51	0.00	0.51	5.82	5.40	1.71
6	0.00	0.00	1.64	4.67	8.53	11.85	9.63	9.17	5.34	3.29	0.73	-1.66	-5.23	-5.63	-0.49	0.00	0.49	5.63	5.23	1.65
7	0.00	0.00	1.53	4.41	8.09	11.25	9.19	8.77	5.11	3.15	0.70	-1.58	-5.00	-5.39	-0.47	0.00	0.47	5.39	5.00	1.58
8	0.00	0.00	1.41	4.10	7.54	10.51	8.65	8.27	4.81	2.97	0.66	-1.49	-4.71	-5.08	-0.45	0.00	0.45	5.08	4.71	1.49
9	0.00	0.00	1.26	3.72	6.88	9.61	7.98	7.64	4.44	2.74	0.61	-1.38	-4.35	-4.69	-0.41	0.00	0.41	4.69	4.35	1.38
10	0.00	0.00	1.09	3.27	6.09	8.54	7.16	6.85	3.99	2.46	0.54	-1.24	-3.91	-4.21	-0.37	0.00	0.37	4.21	3.91	1.24
11	0.00	0.00	0.90	2.75	5.15	7.29	6.17	5.90	3.43	2.12	0.47	-1.07	-3.36	-3.63	-0.32	0.00	0.32	3.63	3.36	1.06
12	0.00	0.00	0.69	2.16	4.07	5.84	4.97	4.76	2.77	1.71	0.38	-0.86	-2.71	-2.92	-0.26	0.00	0.26	2.92	2.71	0.86
13	0.00	0.00	0.47	1.49	2.84	4.17	3.56	3.40	1.98	1.22	0.27	-0.61	-1.94	-2.09	-0.18	0.00	0.18	2.09	1.94	0.61
14	0.00	0.00	0.23	0.77	1.49	2.23	1.90	1.82	1.06	0.65	0.14	-0.33	-1.04	-1.12	-0.10	0.00	0.10	1.12	1.04	0.33
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

	21	22	23	24	25	26	27	28	29	30	31	32
1	-0.97	-4.24	-6.85	-8.40	-8.93	-2.87	-2.07	-5.49	-1.98	0.00	0.00	0.00
2	-0.94	-4.19	-6.79	-8.32	-8.85	-2.85	-2.05	-5.43	-1.95	0.01	0.00	0.00
3	-0.92	-4.14	-6.70	-8.22	-8.72	-2.81	-2.02	-5.34	-1.91	0.02	0.00	0.00
4	-0.90	-4.07	-6.59	-8.08	-8.55	-2.75	-1.97	-5.21	-1.86	0.03	0.00	0.00
5	-0.88	-3.97	-6.43	-7.89	-8.32	-2.67	-1.91	-5.05	-1.79	0.04	0.00	0.00
6	-0.85	-3.85	-6.23	-7.64	-8.03	-2.57	-1.84	-4.84	-1.70	0.05	0.00	0.00
7	-0.82	-3.68	-5.96	-7.31	-7.66	-2.44	-1.74	-4.57	-1.59	0.07	0.00	0.00

8	-0.77	-3.47	-5.62	-6.89	-7.21	-2.28	-1.62	-4.25	-1.46	0.08	0.00	0.00
9	-0.71	-3.20	-5.19	-6.37	-6.65	-2.08	-1.48	-3.85	-1.30	0.10	0.00	0.00
10	-0.64	-2.88	-4.66	-5.71	-5.97	-1.85	-1.31	-3.39	-1.13	0.11	0.00	0.00
11	-0.55	-2.48	-4.01	-4.92	-5.14	-1.56	-1.11	-2.85	-0.93	0.12	0.00	0.00
12	-0.44	-2.00	-3.23	-3.97	-4.15	-1.26	-0.87	-2.23	-0.71	0.11	0.00	0.00
13	-0.32	-1.43	-2.31	-2.84	-2.96	-0.90	-0.61	-1.55	-0.48	0.10	0.00	0.00
14	-0.17	-0.76	-1.24	-1.52	-1.59	-0.48	-0.32	-0.80	-0.23	0.06	0.00	0.00
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

STREAM FUNCTION IN SVERDRUPS, TS= 50

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
32	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
31	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
30	-12.24	-12.67	-11.94	-11.84	-11.76	-11.70	-11.65	-11.62	-11.59	-11.58	-11.57	-11.56	-11.55	-11.55	-11.56	-11.56	-11.57	-11.58	-11.59	-11.60
29	-10.26	-10.10	-10.02	-9.98	-9.93	-9.89	-9.86	-9.85	-9.86	-9.85	-9.84	-9.83	-9.84	-9.85	-9.85	-9.85	-9.65	-9.86	-9.68	-9.88
28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
27	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
26	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	5.97	7.55	7.07	6.21	5.53	5.08	4.79	4.56	4.35	4.14
24	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	11.67	13.66	12.20	10.51	9.36	8.67	8.23	7.86	7.50	7.13
23	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	15.34	16.46	13.86	11.65	10.38	9.70	9.25	8.82	8.37	7.91
22	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	16.03	15.59	12.22	9.98	8.94	8.44	8.05	7.60	7.10	6.61
21	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	13.87	11.96	8.48	6.69	6.10	5.85	5.53	5.06	4.52	4.05
20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9.94	7.16	4.21	3.17	3.10	3.10	2.84	2.35	1.80	1.34
19	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	5.62	2.70	0.68	0.47	0.88	1.09	0.90	0.42	-0.13	-0.55
18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2.02	-0.51	-1.50	-1.00	-0.22	0.17	0.08	-0.32	-0.81	-1.20
17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.41	-2.32	-2.44	-1.46	-0.46	0.06	0.11	-0.13	-0.47	-0.77
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-1.83	-3.13	-2.68	-1.51	-0.50	0.06	0.25	0.24	0.16	0.03
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-2.71	-3.47	-2.77	-1.71	-0.90	-0.47	-0.27	-0.11	0.10	0.30
14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-3.34	-3.63	-2.86	-2.12	-1.67	-1.54	-1.58	-1.58	-1.30	-0.68
13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-3.70	-3.42	-2.50	-2.12	-2.02	-2.16	-2.54	-3.21	-3.97	-3.01
12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-3.70	-2.66	-0.87	-0.87	-0.87	-0.87	-0.87	-0.87	-6.89	-5.89
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-3.75	-2.64	-0.87	-0.87	-0.87	-0.87	-0.87	-0.87	-8.03	-7.63
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-3.50	-2.63	-0.87	-0.87	-0.87	-0.87	-0.87	-0.87	-6.59	-6.60
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-2.68	-2.31	-0.87	-0.87	-0.87	-0.87	-0.87	-0.87	-2.32	-1.82
8	8.55	7.77	6.95	6.11	5.27	4.42	3.50	2.51	1.41	0.16	-1.04	-1.44	-0.87	-0.87	-0.87	-0.87	-0.87	-0.87	4.07	6.63
7	22.56	21.00	19.30	17.60	15.97	14.39	12.81	11.18	9.39	7.41	5.12	2.43	-0.87	-0.87	-0.87	-0.87	-0.67	-0.87	12.84	18.09
6	41.99	39.82	37.08	34.32	31.80	29.78	28.06	26.52	25.17	23.93	22.88	22.20	21.98	22.44	23.25	24.46	26.29	28.61	31.52	33.34
5	67.29	67.29	59.56	54.05	50.23	47.73	46.03	44.75	43.84	43.30	43.04	43.04	43.34	43.93	44.72	45.63	46.64	47.64	48.45	46.90
4	67.29	67.29	66.31	64.77	63.12	61.71	60.67	59.98	59.51	59.20	59.05	59.07	59.26	59.55	59.88	60.21	60.54	60.84	61.06	61.14
3	67.29	67.29	67.29	57.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
32	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
31	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
30	-11.62	-11.64	-11.66	-11.69	-11.72	-11.76	-11.80	-11.84	-11.90	-11.96	-12.04	-12.12	-12.22	-12.33	-12.46	-12.61	-12.77	-12.96	-13.17	-13.41
29	-9.88	-9.89	-9.92	-9.94	-9.96	-9.97	-10.00	-10.04	-10.08	-10.12	-10.16	-10.23	-10.32	-10.41	-10.51	-10.63	-10.79	-11.02	-11.27	-11.55
28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
27	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
26	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
25	3.91	3.67	3.41	3.12	2.81	2.45	2.07	1.65	1.22	0.78	0.36	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
24	6.74	6.34	5.92	5.46	4.94	4.36	3.71	3.00	2.25	1.48	0.71	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
23	7.46	7.03	6.59	6.11	5.57	4.94	4.23	3.44	2.59	1.71	0.82	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
22	6.18	5.82	5.49	5.14	4.74	4.26	3.67	3.00	2.26	1.49	0.72	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
21	3.70	3.47	3.33	3.22	3.06	2.82	2.48	2.05	1.56	1.03	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20	1.08	1.00	1.07	1.19	1.29	1.31	1.23	1.06	0.83	0.55	0.27	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
19	-0.76	-0.72	-0.51	-0.21	0.08	0.29	0.40	0.41	0.34	0.23	0.12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18	-1.37	-1.31	-1.03	-0.65	-0.27	0.02	0.19	0.24	0.21	0.14	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
17	-0.94	-0.90	-0.66	-0.29	0.07	0.35	0.48	0.48	0.38	0.24	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
16	-0.09	-0.11	0.05	0.34	0.65	0.87	0.94	0.86	0.66	0.41	0.17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15	0.32	0.28	0.35	0.56	0.83	1.03	1.10	1.01	0.78	0.49	0.21	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
14	-0.36	-0.33	-0.30	-0.12	0.15	0.43	0.61	0.65	0.55	0.36	0.16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13	-2.26	-2.06	-2.00	-1.82	-1.48	-1.05	-0.65	-0.33	-0.14	-0.05	-0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	-4.76	-4.36	-4.25	-4.07	-3.66	-3.08	-2.42	-1.77	-1.19	-0.70	-0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
11	-6.51	-6.07	-6.00	-5.87	-5.50	-4.89	-4.10	-3.22	-2.31	-1.43	-0.64	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	-5.85	-5.64	-5.77	-5.88	-5.77	-5.39	-4.76	-3.94	-2.99	-1.96	-0.92	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	-1.44	-1.64	-2.18	-2.74	-3.14	-3.30	-3.22	-2.94	-2.46	-1.80	-0.94	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	7.16	6.64	5.66	4.57	3.54	2.62	1.83	1.13	0.52	0.01	-0.27	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
7	19.29	18.83	17.66	16.26	14.77	13.27	11.75	10.10	8.28	6.09	3.38	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	34.00	33.58	32.54	31.31	29.93	28.48	27.08	25.69	24.32	23.05	22.02	21.39	21.30	21.44	21.84	22.29	22.75	23.39	24.14	24.98
5	48.95	48.61	47.96	47.15	46.26	45.35	44.47	43.69	43.04	42.55	42.26	42.21	42.37	42.71	43.20	43.82	44.56	45.40	46.40	47.56
4	61.07	60.88	60.61	60.28	59.92	59.53	59.17	58.87	58.66	58.54	58.50	58.57	58.74	59.02	59.40	59.85	60.36	60.96	61.64	62.39
3	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61
32	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
31	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
30	-13.67	-13.95	-14.24	-14.54	-14.82	-15.07	-15.26	-15.39	-15.42	-15.36	-15.22	-14.99	-14.70	-14.37	-14.01	-13.64	-13.29	-12.97	-12.69	-12.44
29	-11.90	-12.41	-13.11	-13.99	-15.03	-16.31	-17.43	-16.24	-18.68	-18.70	-18.33	-17.62	-16.68	-15.60	-14.46	-13.35	-12.34	-11.50	-10.93	-10.53
28	0.00	0.00	0.00	0.00	0.00	-7.28	-12.14	-15.01	-16.20	-16.14	-15.35	-14.15	-12.62	-10.68	-8.32	-5.68	-2.92	0.00	0.00	0.00
27	0.00	0.00	0.00	0.00	0.00	-3.28	-8.96	-10.79	-11.38	-11.12	-10.30	-9.12	-7.70	-6.12	-4.46	-2.77	-1.15	0.00	0.00	0.00
26	0.00	0.00	0.00	0.00	0.00	-2.13	-3.73	-4.73	-5.12	-4.97	-4.45	-3.72	-2.90	-2.09	-1.35	-0.72	-0.25	0.00	0.00	0.00
25	0.00	0.00	0.00	0.00	0.00	2.44	2.38	1.46	0.78	0.52	0.59	0.80	0.99	1.07	1.00	0.78	0.43	0.00	0.00	0.00
24	0.00	0.00	0.00	0.00	0.00	6.83	7.27	5.83	4.64	4.03	3.79	3.63	3.38	2.95	2.35	1.61	0.80	0.00	0.00	0.00
23	0.00	0.00	0.00	0.00	0.00	9.56	9.47	7.34	5.86	5.18	4.88	4.60	4.15	3.51	2.70	1.79	0.86	0.00	0.00	0.00
22	0.00	0.00	0.00	0.00	0.00	9.87	8.77	6.24	4.85	4.39	4.26	4.06	3.64	3.02	2.27	1.47	0.69	0.00	0.00	0.00
21	0.00	0.00	0.00	0.00	0.00	8.04	6.08	3.65	2.67	2.59	2.71	2.68	2.41	1.96	1.44	0.91	0.42	0.00	0.00	0.00
20	0.00	0.00	0.00	0.00	0.00	5.09	2.80	0.90	0.46	0.75	1.09	1.21	1.10	0.86	0.59	0.34	0.15	0.00	0.00	0.00
19	0.00	0.00	0.00	0.00	0.00	2.26	0.23	-0.96	-0.91	-0.40	0.03	0.21	0.20	0.09	0.00	-0.05	-0.04	0.00	0.00	0.00
18	0.00	0.00	0.00	0.00	0.00	0.39	-1.01	-1.50	-1.14	-0.59	-0.21	-0.07	-0.10	-0.17	-0.21	-0.19	-0.11	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00	-0.41	-1.04	-0.95	-0.48	-0.03	0.19	0.21	0.11	-0.01	-0.10	-0.12	-0.08	0.00	0.00	0.00
16	0.00	0.00	0.00	0.00	0.00	-0.68	-0.55	-0.04	0.43	0.70	0.74	0.63	0.45	0.27	0.13	0.04	-0.01	0.00	0.00	0.00
15	0.00	0.00	0.00	0.00	0.00	-1.18	-0.44	0.40	0.83	0.92	0.82	0.65	0.47	0.32	0.20	0.10	0.03	0.00	0.00	0.00
14	0.00	0.00	0.00	0.00	0.00	-2.39	-1.26	-0.17	0.23	0.21	0.05	-0.10	-0.17	-0.17	-0.12	-0.07	-0.03	0.00	0.00	0.00
13	0.00	0.00	0.00	0.00	0.00	-4.12	-2.92	-1.66	-1.28	-1.37	-1.55	-1.61	-1.51	-1.26	-0.92	-0.56	-0.24	0.00	0.00	0.00
12	0.00	0.00	0.00	0.00	0.00	-5.47	-4.51	-3.29	-2.94	-3.10	-3.31	-3.34	-3.11	-2.66	-2.03	-1.31	-0.59	0.00	0.00	0.00
11	0.00	0.00	0.00	0.00	0.00	-5.21	-4.53	-3.53	-3.34	-3.64	-3.99	-4.13	-3.99	-3.55	-2.87	-1.99	-0.98	0.00	0.00	0.00
10	0.00	0.00	0.00	0.00	0.00	-2.41	-1.51	-0.76	-0.83	-1.37	-1.98	-2.42	-2.62	-2.56	-2.28	-1.79	-1.06	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	3.03	5.32	6.16	5.91	5.14	4.24	3.43	2.78	2.26	1.81	1.33	0.75	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	10.25	15.55	17.23	17.16	16.36	15.35	14.37	13.49	12.72	12.03	11.38	10.77	10.27	9.78	9.23
7	0.00	0.00	0.00	0.00	0.00	19.73	28.28	31.22	31.75	31.27	30.47	29.60	28.81	28.10	27.48	26.91	26.34	25.71	24.92	23.88
6	26.20	27.66	29.51	32.39	36.17	41.02	44.61	46.76	47.43	47.32	47.03	46.53	46.04	45.73	45.40	45.16	45.05	44.78	44.33	43.57
5	48.89	50.41	52.14	54.05	56.02	57.89	59.41	60.44	61.00	61.23	61.28	61.22	61.17	61.23	61.42	61.79	62.46	63.56	65.12	67.29
4	63.19	64.03	64.90	65.78	66.61	67.36	67.97	68.43	68.77	68.98	69.10	69.14	69.13	69.11	69.07	68.99	68.82	68.52	68.05	67.29
3	67.29	67.29	67.29	57.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29	67.29
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

TS= 51	YEAR= 0.28	DAY=102.0	ENERGY= 2.530550E-02	DTEMP= 4.742182E-09	DSALT= 4.648661E-13	SCANS= 29
TS= 52	YEAR= 0.28	DAY=104.0	ENERGY= 2.601716E-02	DTEMP= 4.691837E-09	DSALT= 4.617949E-13	SCANS= 22
TS= 53	YEAR= 0.29	DAY=106.0	ENERGY= 2.674C07E-02	DTEMP= 4.650400E-09	DSALT= 4.608487E-13	SCANS= 30
TS= 54	YEAR= 0.30	DAY=108.0	ENERGY= 2.747494E-02	DTEMP= 4.604165E-09	DSALT= 4.581309E-13	SCANS= 23
TS= 55	YEAR= 0.30	DAY=110.0	ENERGY= 2.822876E-02	DTEMP= 4.565438E-09	DSALT= 4.571380E-13	SCANS= 28
TS= 56	YEAR= 0.31	DAY=112.0	ENERGY= 2.898839E-02	DTEMP= 4.522854E-09	DSALT= 4.547111E-13	SCANS= 22
TS= 57	YEAR= 0.31	DAY=114.0	ENERGY= 2.979392E-02	DTEMP= 4.486779E-09	DSALT= 4.537161E-13	SCANS= 25
TS= 58	YEAR= 0.32	DAY=116.0	ENERGY= 3.061395E-02	DTEMP= 4.447382E-09	DSALT= 4.515255E-13	SCANS= 20
TS= 59	YEAR= 0.32	DAY=118.0	ENERGY= 3.146348E-02	DTEMP= 4.413560E-09	DSALT= 4.505249E-13	SCANS= 21
TS= 60	YEAR= 0.33	DAY=120.0	ENERGY= 3.234246E-02	DTEMP= 4.376923E-09	DSALT= 4.485255E-13	SCANS= 18
TS= 61	YEAR= 0.33	DAY=122.0	ENERGY= 3.418989E-02	DTEMP= 5.801546E-09	DSALT= 9.950718E-13	SCANS= 26
TS= 62	YEAR= 0.34	DAY=124.0	ENERGY= 3.419267E-02	DTEMP= 4.310763E-09	DSALT= 4.457602E-13	SCANS= 26
TS= 63	YEAR= 0.34	DAY=126.0	ENERGY= 3.515872E-02	DTEMP= 4.317552E-09	DSALT= 4.632885E-13	SCANS= 42
TS= 64	YEAR= 0.35	DAY=128.0	ENERGY= 3.614829E-02	DTEMP= 4.248005E-09	DSALT= 4.430678E-13	SCANS= 22
TS= 65	YEAR= 0.36	DAY=130.0	ENERGY= 3.716023E-02	DTEMP= 4.226473E-09	DSALT= 4.446818E-13	SCANS= 23
TS= 66	YEAR= 0.36	DAY=132.0	ENERGY= 3.818270E-02	DTEMP= 4.189409E-09	DSALT= 4.406161E-13	SCANS= 22
TS= 67	YEAR= 0.37	DAY=134.0	ENERGY= 3.922738E-02	DTEMP= 4.166954E-09	DSALT= 4.414077E-13	SCANS= 33
TS= 68	YEAR= 0.37	DAY=136.0	ENERGY= 4.027598E-02	DTEMP= 4.133820E-09	DSALT= 4.382737E-13	SCANS= 17
TS= 69	YEAR= 0.38	DAY=138.0	ENERGY= 4.133140E-02	DTEMP= 4.111586E-09	DSALT= 4.386565E-13	SCANS= 32
TS= 70	YEAR= 0.38	DAY=140.0	ENERGY= 4.239009E-02	DTEMP= 4.081314E-09	DSALT= 4.360925E-13	SCANS= 17
TS= 71	YEAR= 0.39	DAY=142.0	ENERGY= 4.343800E-02	DTEMP= 4.059506E-09	DSALT= 4.362119E-13	SCANS= 28
TS= 72	YEAR= 0.39	DAY=144.0	ENERGY= 4.448277E-02	DTEMP= 4.031377E-09	DSALT= 4.340033E-13	SCANS= 16
TS= 73	YEAR= 0.40	DAY=146.0	ENERGY= 4.551372E-02	DTEMP= 4.010193E-09	DSALT= 4.339521E-13	SCANS= 23
TS= 74	YEAR= 0.41	DAY=148.0	ENERGY= 4.653212E-02	DTEMP= 3.984123E-09	DSALT= 4.320479E-13	SCANS= 12
TS= 75	YEAR= 0.41	DAY=150.0	ENERGY= 4.754101E-02	DTEMP= 3.963642E-09	DSALT= 4.318680E-13	SCANS= 19
TS= 76	YEAR= 0.42	DAY=152.0	ENERGY= 4.853654E-02	DTEMP= 3.939185E-09	DSALT= 4.301644E-13	SCANS= 12
TS= 77	YEAR= 0.42	DAY=154.0	ENERGY= 4.952277E-02	DTEMP= 3.919444E-09	DSALT= 4.298965E-13	SCANS= 22
TS= 78	YEAR= 0.43	DAY=156.0	ENERGY= 5.050274E-02	DTEMP= 3.896437E-09	DSALT= 4.283566E-13	SCANS= 15
TS= 79	YEAR= 0.43	DAY=158.0	ENERGY= 5.147391E-02	DTEMP= 3.877457E-09	DSALT= 4.280238E-13	SCANS= 25
TS= 80	YEAR= 0.44	DAY=160.0	ENERGY= 5.244397E-02	DTEMP= 3.855792E-09	DSALT= 4.266236E-13	SCANS= 15
TS= 81	YEAR= 0.44	DAY=162.0	ENERGY= 5.438156E-02	DTEMP= 5.323382E-09	DSALT= 9.856514E-13	SCANS= 20
TS= 82	YEAR= 0.45	DAY=164.0	ENERGY= 5.438473E-02	DTEMP= 3.817024E-09	DSALT= 4.249635E-13	SCANS= 20
TS= 83	YEAR= 0.45	DAY=166.0	ENERGY= 5.536250E-02	DTEMP= 3.851657E-09	DSALT= 4.515094E-13	SCANS= 32
TS= 84	YEAR= 0.46	DAY=168.0	ENERGY= 5.633883E-02	DTEMP= 3.780093E-09	DSALT= 4.233836E-13	SCANS= 20
TS= 85	YEAR= 0.47	DAY=170.0	ENERGY= 5.733210E-02	DTEMP= 3.770184E-09	DSALT= 4.254527E-13	SCANS= 19
TS= 86	YEAR= 0.47	DAY=172.0	ENERGY= 5.831257E-02	DTEMP= 3.744343E-09	DSALT= 4.217962E-13	SCANS= 20
TS= 87	YEAR= 0.48	DAY=174.0	ENERGY= 5.932801E-02	DTEMP= 3.732562E-09	DSALT= 4.230173E-13	SCANS= 15
TS= 88	YEAR= 0.48	DAY=176.0	ENERGY= 6.033170E-02	DTEMP= 3.710017E-09	DSALT= 4.202777E-13	SCANS= 17
TS= 89	YEAR= 0.49	DAY=178.0	ENERGY= 6.136440E-02	DTEMP= 3.697532E-09	DSALT= 4.210354E-13	SCANS= 13
TS= 90	YEAR= 0.49	DAY=180.0	ENERGY= 6.239805E-02	DTEMP= 3.677003E-09	DSALT= 4.188217E-13	SCANS= 19
TS= 91	YEAR= 0.50	DAY=182.0	ENERGY= 6.344897E-02	DTEMP= 3.664148E-09	DSALT= 4.192757E-13	SCANS= 15
TS= 92	YEAR= 0.50	DAY=184.0	ENERGY= 6.450293E-02	DTEMP= 3.643057E-09	DSALT= 4.174036E-13	SCANS= 21
TS= 93	YEAR= 0.51	DAY=186.0	ENERGY= 6.557865E-02	DTEMP= 3.632082E-09	DSALT= 4.176544E-13	SCANS= 24
TS= 94	YEAR= 0.51	DAY=188.0	ENERGY= 6.664950E-02	DTEMP= 3.614298E-09	DSALT= 4.160472E-13	SCANS= 17
TS= 95	YEAR= 0.52	DAY=190.0	ENERGY= 6.774970E-02	DTEMP= 3.601247E-09	DSALT= 4.161374E-13	SCANS= 20
TS= 96	YEAR= 0.53	DAY=192.0	ENERGY= 6.884376E-02	DTEMP= 3.584221E-09	DSALT= 4.146953E-13	SCANS= 18
TS= 97	YEAR= 0.53	DAY=194.0	ENERGY= 6.996130E-02	DTEMP= 3.571400E-09	DSALT= 4.146942E-13	SCANS= 20

Answers after Circ. 10 - L 41 - 4/22/88

M. Restarting procedure

All data that is needed to restart a run from a previous run is included in units 11-15. When using disc I/O, if permanent disc files are available, no action is necessary at the end of each run. If the disc files are not reserved after the end of the run, they must be saved to tape and then recopied to disc when restarting.

When running in core contained mode, the virtual disc, "BIG" must be saved to tape at the end of each run. This is designated by setting NA=1 in the "CONTRL" NAMELIST file, wherupon unit 21 will receive the restart data. For safety, it may be wise to save this file periodically through the run. This can be done by setting NWRITE to the interval, in timesteps, for which this is to be done, the data going to unit 22.

In both the disc I/O and core contained modes, restarting is indicated by setting NFIRST=0 on the "CONTRL" NAMELIST file, whereupon the run will restart at whatever timestep the data on the disc (or virtual disc) is defined.

GFDL OCEAN MODEL CIRCULAR NO. 1

Sept. 28, 1984

Toward the goal of keeping all users and potential users of the GFDL ocean model abreast of the latest happenings with respect to the model code, this is the first in a series of circulars to be distributed occasionally. It is anticipated that these notes will include updates to the code of both PERMANENT and GLOBAL types (see sec. H of the manual), clarifications and corrections to the manual, and various other communications which may be helpful. It is suggested that they be kept in the binder with the manual. All persons who have requested the documentation manual will receive the circulars unless they indicate otherwise.

The following update should be made permanently to the base code. Some of these changes have already been indicated to some users on the introductory note accompanying the tape.

PERMANENT UPDATE:

```
      IF(J.EQ.JMTM1) THEN                                  ...  345500000 S
        DO 382 I=1,IMT                                     ...  345510000 S
          ZVN(I)=0.                                        ...  345520000 S
 382    CONTINUE                                           ...  345530000 S
      ENDIF                                                ...  345540000 S

      PARAMETER(N11=20,N12=IMT*JMT*NKFLDS,                 ...  640800000 K

      PARAMETER (IMP1X2=IMTP1*2)                           ...  703200000 F
      DIMENSION COSSAV(LQMSUM),DENMSV(LHSUM),COSNPI(IMT)   ...  703800000 F
      DIMENSION DIFF(IMP1X2)                               ...  704400000 F
      LOGICAL INITDN                                       ...  704900000 F
      DATA INITDN/.FALSE./                                 ...  704910000 F
      IF(INITDN) GOTO 90                                   ...  706100000 F
      DENMSV(IBASE+I)=1.0/(1.0-COS(PI*FLOAT(I)*FIMR))      ...  708700000 F
      INITDN=.TRUE.                                        ...  710700000 F
      DENOM(I)=FXA*DENMSV(IBASE+I)                         ...  719100000 F

 3000 DO 3010 I=1,IM                                       ...  729400000 F
      SPRIME(I)=0.                                         ...  729410000 F
 3010 CONTINUE                                             ...  729420000 F

      DO 3110 I=1,IM                                       ...  730100000 F
      SPRIME(I)=FNORM*SPRIME(I)                            ...  730110000 F
 3110 CONTINUE                                             ...  730120000 F
```

Explanatory remarks:

345500000-345540000:

This is a replacement of CDC 205 specific code by equivalent standard FORTRAN. No change in answers or running speed.

(over)

640800000:

This is a minor bug. N11 is changed from 5 to 20. No change in answers unless the auxiliary array AKNTRL, normally unused, was brought into use specifically by the local programmer.

703200000-719100000:

Same as 345500000-345540000 (.. >6 characters in variable name & implied data statement not standard)

729400000-729420000:

Same as 345500000-345540000

730100000-730120000:

Same as 345500000-345540000

Manual correction:

On p. I1, the formula for MEM(BIG) should read (corrections underlined):

```
MEM(BIG)=20+NKFLDS*IMT*JMT+3*((NT+2)*IMT*KM+LBC*IMT)*JMT
```

END

GFDL OCEAN MODEL CIRCULAR NO. 2

Oct. 28, 1984

Minor alterations are needed to get the model running on various systems. For systems which require all FORTRAN I/O units to be specified ahead of time, for instance on the PROGRAM statement, units 10,20,22,24, and 25 must be so specified before running the UPDOC utility.

Some systems require header cards preceding each routine in the FORTRAN source code and others do not. Such cards (*DECK) are included in the source code supplied to you originally, and are the separators required by the CDC system at GFDL. If your system does not use separators between routines, an alteration to the UPDOC utility, suggested by David Webb of IOS, England can be made to eliminate them. Immediately after the statement labelled 322 in UPDOC, place the following line:

```
      IF(CD(1).EQ.'*' .AND. CD(2).EQ.'D') GO TO 320
```

An additional line can be inserted at this point if an alternate separator is needed.

The following update should be made <u>permanently to the base code</u>.

<u>PERMANENT UPDATE:</u>

```
            ISM1=IS-1                                                     376810000 F
            IEA=IE                                                        376820000 F
            IF(IE.GE.IMU) IEA=IMUM1                                       376830000 F
            DO 700 I=IS,IEA                                               376900000 F
-377900000
              UDIF(I-ISM1,K)=-FX*UA(I,K)*SPSIN(I)-VA(I,K)*SPCOS(I)        377100000 F
              VDIF(I-ISM1,K)= FX*UA(I,K)*SPCOS(I)-VA(I,K)*SPSIN(I)        377200000 F
            IF(IE.GE.IMU) THEN                                            377310000 F
              IEB=IE-IMUM2                                                377320000 F
              II=IMUM1-IS                                                 377330000 F
              DO 702 I=2,IEB                                              377340000 F
                UDIF(I+II,K)=-FX*UA(I,K)*SPSIN(I)-VA(I,K)*SPCOS(I)        377350000 F
                VDIF(I+II,K)= FX*UA(I,K)*SPCOS(I)-VA(I,K)*SPSIN(I)        377360000 F
  702         CONTINUE                                                    377370000 F
            ENDIF                                                         377380000 F
            DO 720 I=IS,IEA                                               377900000 F
-378000000
              UA(I,K)=FX*(-UDIF(I-ISM1,K)*SPSIN(I)                        378100000 F
     *                    +VDIF(I-ISM1,K)*SPCOS(I))                       378200000 F
              VA(I,K)=-UDIF(I-ISM1,K)*SPCOS(I)-VDIF(I-ISM1,K)*SPSIN(I)    378300000 F
            IF(IE.GE.IMT) THEN                                            378410000 F
              DO 722 I=2,IEB                                              378420000 F
                UA(I,K)=FX*(-UDIF(I+II,K)*SPSIN(I)                        378430000 F
     *                      +VDIF(I+II,K)*SPCOS(I))                       378440000 F
                VA(I,K)=-UDIF(I+II,K)*SPCOS(I)-VDIF(I+II,K)*SPSIN(I)      378450000 F
  722         CONTINUE                                                    378460000 F
            ENDIF                                                         378470000 F
              ISM1=IS-1                                                   437710000 F
              IEA=IE                                                      437720000 F
              IF(IE.GE.IMT) IEA=IMTM1                                     437730000 F
              DO 1100 I=IS,IEA                                            437800000 F
-437900000
                TDIF(I-ISM1,K,1)=TA(I,K,MM)                               438000000 F
              IF(IE.GE.IMT) THEN                                          438110000 F
                IEB=IE-IMTM2                                              438120000 F
                II=IMTM1-IS                                               438130000 F
                DO 1102 I=2,IEB                                           438140000 F
                  TDIF(I+II,K,1)=TA(I,K,MM)                               438150000 F
 1102           CONTINUE                                                  438160000 F
              ENDIF                                                       438170000 F
              DO 1120 I=IS,IEA                                            438500000 F
-438600000
                TA(I,K,MM)=TDIF(I-ISM1,K,1)                               438700000 F
              IF(IE.GE.IMT) THEN                                          438810000 F
                DO 1122 I=2,IEB                                           438820000 F
                  TA(I,K,MM)=TDIF(I+II,K,1)                               438830000 F
 1122           CONTINUE                                                  438840000 F
              ENDIF                                                       438850000 F
```

(over)

Explanatory remarks:

376810000-438850000:

This is an alteration to enable vectorization of the section of the code which sets up the strip of velocities (1st part) and tracers (2nd part) which are to be filtered when the "F" option is used. The previous code contained a MOD function which prohibits vectorization on some machines. No change in answers, but a speed increase of 10-20% depending upon the model.

END

April 30, 1985

Please advise me of address changes. (Mike Cox, Box 308, Princeton, NJ 08540)

There are two pages accompanying this circular which should replace the corresponding pages of the GFDL ocean model documentation, along with a new table of contents which should replace the old one. An error in the indexing of W is corrected on p. G 3, and the error in the memory requirement formula (first reported in Circular 1) is corrected on p. I1.

The following update should be made permanently to the base code.

PERMANENT UPDATE:

```
      BIT KEVENBV(IMT,KM),KODDBV(IMT,KM),KALTBV(IMT,KM,2),CONBV(IMT,KM)          009500000
C  CONVERT HEAT TRANSPORT TO PETAWATTS, SALT TRNSPT TO 10**10 CM**3/SEC          274200000 C
      DO 967 N=1,NCON                                                            429400000
      LCB=0                                                                      429410000 W
        CONBV(1,KS;LN)=KALTBV(1,KS,KS;LN)                                        431310000 W
     *                   .AND. TEMPA(1,KS;LN).GT.TEMPA(1,KS+1;LN)                431320000 W
        LCA=Q8SCNT(CONBV(1,KS;LN))                                               431330000 W
        LCB=LCB+LCA                                                              431340000 W
        IF(LCA.EQ.0) GO TO 965                                                   431350000 W

          WHERE (CONBV(1,KS;LN))
-431700000                                                                       431600000 W
      IF(LCB.EQ.0) GO TO 968
  967 CONTINUE                                                                   432310000 W
  968 CONTINUE                                                                   432320000
C               MM =1 (COSINE SERIES, DERIV AT BNDRY PTS=0)                      432330000 W
C                  =2 (  SINE SERIES,          BNDRY PTS=0)          ========== 701000000 FC
                                                                     ========== 701100000 FC
```

Explanatory remarks:

009500000 & 429400000-4323300000

This is an improvement in the 205 optimized code to eliminate passes through the convection loop (when NCON>1) when static stability has already been achieved and further passes would accomplish nothing. No change in answers, but a minor speed increase depending on model.

274200000

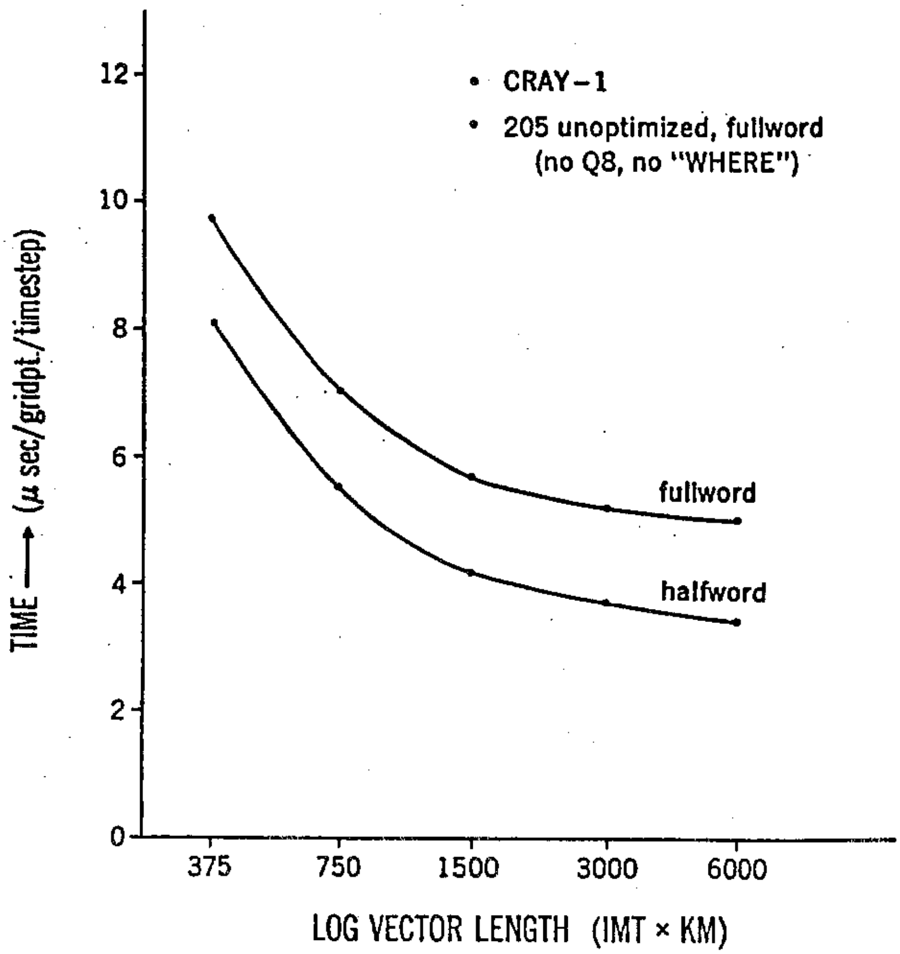
Spelling correction on comment line.

701000000-701100000

Correction of erroneous comments.

(over)

There have been several inquiries concerning the running speed of the model. A convenient method of representing the speed is the quantity, time per gridpoint per timestep. There are several factors in the model which affect this quantity. For vectorizing computers, one of the most important factors affecting rate of computation is vector length. The curves plotted below show the results of several test runs of the model on the CYBER 205 with "Q" and "W" optimization specified (see p. H 1 of documentation), and for fullword (64 bit) and halfword (32 bit) mode as functions of the most predominant vector length in the model, IMT*KM. There is an appreciable increase in computation rate between vector length O(100) and O(1000) on the 205. The computation rate for the model run on the 205 with no "Q" or "W" optimization, as well as the rate using the CRAY-1 at NCAR is indicated for the vector length 1500 case by lone dots. Note that the computation rate of the CRAY-1 is not as sensitive to vector length, for vectors more than several multiples of 64, and its curve is probably nearly horizontal through most of the region depicted here.



Obviously, there are many other factors affecting the computation rate, some of which are listed below with their values in the tests above:

# of relaxation scans to convergence (MSCAN)	20
# of convection passes (NCON)	2
frequency of Euler backward timestep (EB, NMIX)	0
frequency of energy/printout timestep (NNERGY)	0
# of tracers computed (NT)	2
use of Fourier filtering at high latitudes	0
# of islands computed (NISLE)	0

END

GFDL OCEAN MODEL CIRCULAR NO. 4

Feb. 12, 1986

If my current address for you is not up to date, please advise.
(Mike Cox, Box 308, Princeton, NJ 08540 - Tel. (609) 452-6531)

Since the last circular, there has been a series of suggestions for correction and improvement to the model. These have been included on pages 2 and 3 as a permanent update. In each case, the alterations produce either a speed increase, prevention of machine bomb for certain configurations of the model, or simply an aesthetic improvement to the code. None of them produce significant (other than roundoff) changes in answers. Thanks to those indicated for their suggestions.

Some additional work has been done on the UPDOC utility recently and a factor of two increase in running speed has been achieved. A listing of the new version is included on pages 4 and 5.

A "global" update (see p. H3 of the documentation) for utilization of a Robert time filter on prognostic variables has been submitted for general use by Malcolm MacVean of the U.K. Met. Office and is included on page 6.

```
*************************************************************************
*                                                                       *
*       Anyone wishing to have their program tape updated to the current *
*   version (through this circular, including the new UPDOC) may do so by *
*   returning the original tape or sending a new one to me at any time.  I *
*   will update it and return it promptly.                              *
*                                                                       *
*************************************************************************
```

(over)

The following update should be made <u>permanently to the base code</u>.

PERMANENT UPDATE:

```
   *,NIEVEN=2*((NISLE+1)/2)                                              002210000
   *,ISMASK(IMT,JMT),COF(IMT,JMT),COFIS(NIEVEN),                         009100000 I
C                                                                         166600000 KC
C-----------------------------------------------------------------------  166700000 KC
C  SAVE RESTART DATA EVERY NWRITE TIMESTEPS, AND AT END OF RUN IF NA=1    166800000 KC
C-----------------------------------------------------------------------  166900000 KC
C                                                                         167000000 KC
C                                                                         211710000 C
C-----------------------------------------------------------------------  211720000 C
C  RESET KM+1 BOUNDARY VELOCITY TO ZERO                                   211730000 C
C-----------------------------------------------------------------------  211740000 C
C                                                                         211750000 C
      DO 122 I=1,IMT                                                      211760000
        UUNDER(I)=0.                                                      211770000
        VUNDER(I)=0.                                                      211780000
 122  CONTINUE                                                            211790000
-260600000,260900000
        DO 8136 K=1,KM                                                    261000000
          IF(KMT(I).GE.K.AND.KMTP(I).GE.K) THEN                           261010000
            VBRZ=VBRZ+(V(I,K)*DXU(I)+V(I-1,K)*DXU(I-1))*DZ(K)             261100000
            TBRZ=TBRZ+(T(I,K,M)+TP(I,K,M))*DZ(K)                          261200000
            TOTDZ=TOTDZ+DZ(K)                                             261300000
          ENDIF                                                           261310000
        IF(TOTDZ.EQ.0.) GO TO 8140                                        261410000
        IF(KZ.EQ.0) GO TO 524                                             355710000
        IF(KZ.EQ.0) GO TO 520                                             356910000
      DO 521 K=1,KZ                                                       357000000
 521  CONTINUE                                                            358210000
            N=NINT(IM*CS(J)*CSR(JFUO))                                    376000000 F-O
               N=NINT(IM*CS(J)*CSR(JFUO))                                 376300000 FO
               N=NINT(IM*CS(J)*CSR(JFUO)*0.5)                             376600000 FO
        N=NINT(IM*CST(J)*CSR(JFUO))                                       382600000 F-O
           N=NINT(IM*CST(J)*CSR(JFUO))                                    382900000 FO
           N=NINT(IM*CST(J)*CSR(JFUO)*0.5)                                383200000 FO
            N=NINT(IM*CST(J)*CSTR(JFTO))                                  436700000 F-O
               N=NINT(IM*CST(J)*CSTR(JFTO))                               437000000 FO
               N=NINT(IM*CST(J)*CSTR(JFTO)*0.5)                           437300000 FO
C            --COEFFICIENTS FOR THE LOWER OF THE 2 LEVELS ARE USED  ===========  608600000 C
      REWIND LO                                                           647810000 K
      DIMENSION ICBASE(IMTP1),IDBASE(IMTP1),IND(IMTX8),TEMP(IMTX4)        703700000 F
      DO 10 I=1,IMTX8                                                     707300000 F
      C1=0.5*FLOAT(NMAX)+0.25                                             712410000 F
      C2=FLOAT(NMAX)+0.5                                                  712420000 F
-717600000
      DO 701 I=1,LQM                                                      717710000 F
 701  COSINE(LH-I)=-COSSAV(JBASE+I)                                       717720000 F
      DO 710 I=1,LH                                                       718300000 F-O
      COSINE(LH+I)=-COSINE(I)                                             718400000 F-O
 710  CONTINUE                                                            718500000 F-O
      COSINE(LH+1;LH)=-COSINE(1;LH)                                       718510000 FO
      TEMP(I)=DENOM(LH-I)                                                 719500000 F
      DO 722 I=1,LHM1                                                     719610000 F
 722  DENOM(LH+I)=TEMP(I)                                                 719620000 F
      DO 730 I=LCYP1,IMX4                                                 719900000 F-O
      DENOM(I)=DENOM(I-LCY)                                               720000000 F-O
 730  CONTINUE                                                            720100000 F-O
      DENOM(LCYP1;IMX4-LCY)=DENOM(1;IMX4-LCY)                             720110000 FO
      INDX(I)=IND(I)*FACT1                                                721500000 F
-721600000
      DO 741 I=1,IMX4                                                     721710000 F
 741  INDX(IMX4+I)=IND(I)*FACT2                                           721720000 F
      DO 810 J=1,IMX8                                                     723800000 F-O
      COF(J)=COSINE(INDX(J))                                              723900000 F-O
 810  CONTINUE                                                            724000000 F-O
      COF(1;IMX8)=Q8VGATHR(COSINE(1;IMTX8),INDX(1;IMX8);COF(1;IMX8))      724010000 FO
-724200000
      IF(MM.EQ.1) THEN                                                    724610000 F
-725600000
      DO 1201 J=1,IM                                                      725710000 F
 1201 FTARR(J*IMTP1-IMT)=FTARR(J*IMTP1-IMT)+C1                            725720000 F
      ELSE IF(MM.EQ.2) THEN                                               725800000 F
-726900000
      DO 1501 J=1,IM                                                      727010000 F
 1501 FTARR(J*IMTP1-IMT)=FTARR(J*IMTP1-IMT)+C1                            727020000 F
      ELSE IF(MM.EQ.3) THEN                                               727100000 F
-728800000
      DO 1801 J=1,IM                                                      728910000 F
 1801 FTARR(J*IMTP1-IMT)=FTARR(J*IMTP1-IMT)+C2                            728920000 F
      ENDIF                                                               729000000 F
```

(over)

Explanatory remarks:

002210000-0091000000

This gets around a potential problem which can arise when using half-precision and islands. Machine bomb is prevented for these cases, no change in answers otherwise. (A. Rosati, GFDL)

166600000-167000000

The "C" option identifier was inadvertantly left off these comments. No change in answers.

211710000-211790000

Initialization which is needed for certain configurations of the model. Machine bomb is prevented for these models, no change in answers otherwise. (H. Cattle, U.K. Met. Office)

260600000-261410000

Correction of a minor bug in the code to compute northward transport of heat, salt, tracer due to vertically averaged flow. No change in answers for prognostic variables.

355710000-358210000

New code to prevent execution of a possible null DO loop. No change in answers. (M. MacVean, U.K. Met. Office)

376000000-437300000

Replacement of "IFIX" function by "NINT" function which is more appropriate for this section of code. No change in answers. (M. MacVean, U.K. Met. Office)

608600000

Correction of erroneous comment.

647810000

This causes the restart file to be overwritten each time in core-contained mode. Whether overwriting or cumulative writing is done is at the option of the programmer by keeping or deleting this rewind. No change in answers.

703700000-729000000

These are various updates to FILTER which enable vectorization of code previously unvectorizable on the CDC-205. There are insignificant changes in answers due to some reordering of computation, and a modest speed increase. (M. MacVean, U.K.M.O)

(over)

```
      PROGRAM UPDOC
C
C-----------------------------------------------------------------------
C   OCEAN UPDATING PROGRAM
C   / IN COL 1 TO START NEW UPDATE MODULE
C   : IN COL 1 FOR COMMENT (IGNORED BY UPDOC)
C   SEE GFDL OCEAN GROUP TECH REPORT #1, SECTION H FOR DOCUMENTATION
C-----------------------------------------------------------------------
C
      CHARACTER CD*80,CDUPD*80,OPTIN*19,OPTCD*9,OPTUPD*9,NAME*6,NAMEC*6
      CHARACTER LN*100
      DIMENSION NAME(70)
      LOGICAL LOPT,LMATCH
      REWIND 10
      REWIND 20
      REWIND 22
      REWIND 24
      OPTIN=' '
C
C-----------------------------------------------------------------------
C   READ IN CODE OPTIONS
C-----------------------------------------------------------------------
C
      READ(10,9998,END=400) CD
      PRINT 9984
      PRINT 9994,CD
      IF(CD(1:4).NE.'OPT=') GO TO 400
      OPTIN=CD(5:23)
C
C-----------------------------------------------------------------------
C   PROCESS CODE UPDATES
C-----------------------------------------------------------------------
C
      LR=20
      LW=22
      MAXSEQ=999999999
 100  MUPDB=0
      READ(LR,9998,END=105) CD,M,OPTCD
      GO TO 110
 105  M=MAXSEQ
 110  READ(10,9998,END=190) CDUPD,MUPD,OPTUPD
      PRINT 8998,CDUPD,MUPD,OPTUPD
      IF(CDUPD(1:1).EQ.':') GO TO 110
      IF(CDUPD(1:1).EQ.'/') GO TO 190
      CDUPD(73:73)='*'
      IF(CDUPD(1:1).EQ.'-') GO TO 180
    — IF(MUPD.LT.MUPDB) GO TO 600
      MUPDB=MUPD
 120  IF(M.GT.MUPD) GO TO 160
      IF(M.EQ.MUPD) GO TO 140
      MX=M/1000
      IF(MOD(M,100000).EQ.0) GO TO 126
      MX=MOD(M,1000000)
      WRITE(LW,9989) CD(1:73),MX,M,OPTCD
      GO TO 128
 126  WRITE(LW,9990) CD(1:73),MX,M,OPTCD
 128  READ(LR,9998,END=130) CD,M,OPTCD
      GO TO 120
 130  M=MAXSEQ
      GO TO 120
C
C  REPLACE LINE
C
 140  CD=CDUPD
      M=MUPD
      OPTCD=OPTUPD
      GO TO 110
C
C  INSERT LINE
C
 160  MX=MUPD/1000
      IF(MOD(MUPD,100000).EQ.0) GO TO 166
      MX=MOD(MUPD,1000000)
      WRITE(LW,9989) CDUPD(1:73),MX,MUPD,OPTUPD
      GO TO 110
 166  WRITE(LW,9990) CDUPD(1:73),MX,MUPD,OPTUPD
      GO TO 110
C
C  DELETE LINE(S)
C
 180  READ(CDUPD,9995)MS,ME
      IF(MS.LE.MUPDB) GO TO 600
      MUPDB=MS
      IF(ME.EQ.0) ME=MS
 182  IF(M.LT.MS) GO TO 186
 184  IF(M.GT.ME) GO TO 110
      READ(LR,9998,END=185) CD,M,OPTCD
      GO TO 184
 185  M=MAXSEQ
      GO TO 184
 186  MX=M/1000
      IF(MOD(M,100000).EQ.0) GO TO 187
      MX=MOD(M,1000000)
      WRITE(LW,9989) CD(1:73),MX,M,OPTCD
      GO TO 189
 187  WRITE(LW,9990) CD(1:73),MX,M,OPTCD
 189  READ(LR,9998,END=188) CD,M,OPTCD
      GO TO 182
 188  M=MAXSEQ
      GO TO 182
 190  CONTINUE
      IF(M.EQ.MAXSEQ) GO TO 200
```

(over)

```
      MX=M/1000
      IF(MOD(M,100000).EQ.0) GO TO 196
      MX=MOD(M,1000000)
      WRITE(LW,9989) CD(1:73),MX,M,OPTCD
      GO TO 192
196   WRITE(LW,9990) CD(1:73),MX,M,OPTCD
192   READ(LR,9998,END=200) CD,M,OPTCD
      MX=M/1000
      IF(MOD(M,100000).EQ.0) GO TO 197
      MX=MOD(M,1000000)
      WRITE(LW,9989) CD(1:73),MX,M,OPTCD
      GO TO 192
197   WRITE(LW,9990) CD(1:73),MX,M,OPTCD
      GO TO 192
200   REWIND LW
      IF(CDUPD(1:1).NE.'/') GO TO 220
C
C  PREPARE TO PROCESS ANOTHER UPDATE SET
C
      LS=LR
      LR=LW
      LW=LS
      REWIND LR
      REWIND LW
      CDUPD(1:1)=' '
      GO TO 100
C
C-----------------------------------------------------------------------
C  PROCESS CODE OPTIONS
C-----------------------------------------------------------------------
C
220   READ(LW,9998,END=300) CD,M,OPTCD
      LOPT=.TRUE.
      DO 250 N=1,9
        IF(OPTCD(N:N).EQ.' ') GO TO 260
        IF(OPTCD(N:N).EQ.'-') THEN
          LOPT=.FALSE.
          GO TO 250
        ENDIF
        LMATCH=.FALSE.
        DO 230 NN=1,19,2
          IF(OPTCD(N:N).NE.OPTIN(NN:NN)) GO TO 230
          LMATCH=.TRUE.
          GO TO 240
230     CONTINUE
240     IF((LMATCH.AND..NOT.LOPT) .OR. (.NOT.LMATCH.AND.LOPT)) GO TO 220
        LOPT=.TRUE.
250   CONTINUE
260   WRITE(24,9998) CD,M,OPTCD
      GO TO 220
300   CONTINUE
      REWIND 24
C
C-----------------------------------------------------------------------
C  PLACE COMDECKS
C-----------------------------------------------------------------------
C
      REWIND 20
      DO 310 N=41,70
        NAME(N)='
310   CONTINUE
      L=40
320   READ(24,9997,END=390) LN
322   IF(LN(1:1).EQ.'*' .AND. LN(2:2).EQ.'C') GO TO 340
      WRITE(20,9997) LN
      GO TO 320
340   IF(LN(3:3).EQ.'A') GO TO 360
      L=L+1
      NAME(L)=LN(10:15)
342   READ(24,9997,END=390) LN
      IF(LN(1:1).EQ.'*') GO TO 322
      WRITE(L ,9997) LN
      GO TO 342
360   NAMEC=LN(7:12)
      L=40
362   L=L+1
      IF(NAME(L).EQ.'      ') GO TO 500
      IF(NAME(L).NE.NAMEC) GO TO 362
      REWIND L
364   READ(L ,9997,END=368) LN
      WRITE(20,9997) LN
      GO TO 364
368   REWIND L
      GO TO 320
390   CONTINUE
      STOP
400   PRINT 9996
      STOP
500   PRINT 9992,NAMEC
      STOP
600   PRINT 9991
      STOP
9998  FORMAT(A80,I10,1X,A9)
9997  FORMAT(A100)
9996  FORMAT(' ABNORMAL END -- NO OPTION RECORD PROVIDED')
9995  FORMAT(2(1X,I9))
9994  FORMAT(1X,A80)
9992  FORMAT(' ABNORMAL END -- COMDECK ',A6,' NOT PROVIDED')
9991  FORMAT(' ABNORMAL END -- UPDATES NOT IN ASCENDING ORDER')
9990  FORMAT(A73,I6,'>',I10,1X,A9)
9989  FORMAT(A73,'<',I6,I10,1X,A9)
9984  FORMAT(/)
8998  FORMAT(1X,A80,I10,1X,A9)
      END
```

(over)

GLOBAL UPDATE: Robert time filter

Submitted by:
(& all inquiries to)

Malcolm MacVean
Meteorological Office
Bracknell, Berkshire
RG12 2SZ
ENGLAND

Notes:

1. Tested only under K option. Due to the additional DPUT at 248627000 which interferes with the normal I/O flow of the model, severe reduction in wall-clock efficiency may be experienced when running in disc I/O mode.

```
     X NDISKS,PNU,PNU2M,                                                003350000
     X,TF(IMT,KM,NT),UF(IMT,KM),VF(IMT,KM),FBCON(IMT,LBC)                007110000
     X,FBCONP(IMT,LBC)                                                   007120000
     X,SSFUB(IMT),SSFVB(IMT),SSFUBP(IMT),SSFVBP(IMT)                     007130000
      REAL PNUF                                                          104710000
     X,EB,PNUF                                                           105010000
      PNU=PNUF                                                           116710000
      PNU2M=1.-2.*PNU                                                    116720000
      NDISKS=NDISKA                                                      163250000
C                                                                        216750000 C
C-----------------------------------------------------------------       216751000 C
C     SAVE INCIDENTAL QUANTITIES FOR USE IN TIME FILTER                  216752000 C
C-----------------------------------------------------------------       216753000 C
C                                                                        216754000 C
      DO 210 N=1,NSWICH                                                  216755000
      FBCONP(N,1)=FKMTP(N)                                               216756000
  210 CONTINUE                                                           216757000
C                                                                        229550000 C
C-----------------------------------------------------------------       229551000 C
C     SAVE EXTERNAL MODE FOR USE IN TIME FILTER                          229552000 C
C-----------------------------------------------------------------       229553000 C
C                                                                        229554000 C
      DO 290 I=1,IMT                                                     229555000
      SSFUBP(I)=SFUB(I)                                                  229556000
      SSFVBP(I)=SFVB(I)                                                  229557000
  290 CONTINUE                                                           229558000
C     TOGETHER WITH QUANTITIES SAVED FOR TIME FILTER                     235250000 C
      DO 321 I=1,IMT                                                     236050000
      SSFUB(I)=SSFUBP(I)                                                 236051000
      SSFVB(I)=SSFVBP(I)                                                 236052000
  321 CONTINUE                                                           236053000
      DO 322 N=1,NSWICH                                                  236054000
      FBCON(N,1)=FBCONP(N,1)                                             236055000
  322 CONTINUE                                                           236056000
C     SAVE INCIDENTAL QUANTITIES FOR TIME FILTER                         236350000 C
      DO 323 N=1,NSWICH                                                  236850000
      FBCONP(N,1)=FKMTP(N)                                               236851000
  323 CONTINUE                                                           236852000
C                                                                        248601000 C
C-----------------------------------------------------------------       248602000 C
C     TIME FILTER THE TRACERS AND BAROCLINIC VELOCITIES TO PRODUCE       248603000 C
C     NEW VALUES AT TIME LEVEL TAU. THE UNFILTERED TIME LEVEL TAU        248604000 C
C     VALUES ON 'DISK' ARE THEN OVERWRITTEN WITH THE NEW SLAB (TF)       248605000 C
C-----------------------------------------------------------------       248606000 C
C                                                                        248607000 C
      DO 346 M=1,NT                                                      248608000
      DO 346 K=1,KM                                                      248609000
      DO 346 I=1,IMT                                                     248610000
      TF(I,K,M)=PNU2M*T(I,K,M)+PNU*(TA(I,K,M)+TB(I,K,M))                 248611000
  346 CONTINUE                                                           248612000
      IF(J.NE.JMTM1)THEN                                                 248613000
      DO 349 K=1,KM                                                      248613500
      DO 349 I=1,IMT                                                     248613510
      UF(I,K)=GM(I,K)*SSFUB(I)                                           248613520
      VF(I,K)=GM(I,K)*SSFVB(I)                                           248613530
  349 CONTINUE                                                           248613540
      DO 350 K=1,KM                                                      248613550
      DO 350 I=1,IMT                                                     248613560
      UF(I,K)=PNU*(UB(I,K)-UF(I,K)+UA(I,K))                              248613570
      VF(I,K)=PNU*(VB(I,K)-VF(I,K)+VA(I,K))                              248613580
  350 CONTINUE                                                           248613590
      DO 347 K=1,KM                                                      248614000
      DO 347 I=1,IMT                                                     248615000
      UF(I,K)=PNU2M*USAV(I,K)+UF(I,K)                                    248616000
      VF(I,K)=PNU2M*VSAV(I,K)+VF(I,K)                                    248617000
  347 CONTINUE                                                           248618000
      ELSE                                                               248619000
      FX=0.                                                              248620000
      DO 348 K=1,KM                                                      248621000
      DO 348 I=1,IMT                                                     248622000
      UF(I,K)=FX                                                         248623000
      VF(I,K)=FX                                                         248624000
  348 CONTINUE                                                           248625000
      ENDIF                                                              248626000
      CALL DPUT(LABS(NDISK),NSLAB,(J-1)*NSLAB+1,TF)                      248627000
      NDISKS=NDISKA                                                      281450000
C                                                                        311750000 C
C-----------------------------------------------------------------       311751000 C
C     SAVE EXTERNAL MODE FOR USE IN TIME FILTER                          311752000 C
C-----------------------------------------------------------------       311753000 C
C                                                                        311754000 C
      DO 156 I=1,IMT                                                     311755000
      SSFUBP(I)=SFUB(I)                                                  311756000
      SSFVBP(I)=SFVB(I)                                                  311757000
  156 CONTINUE                                                           311758000
C     TIME FILTERING THE STREAMFUNCTION AT TIME LEVEL TAU                560950000 C
C     THE UNFILTERED FIELD HELD ON 'DISK' IS OVERWRITTEN WITH THE        560951000 C
C     VALUES FROM THE FILTERED FIELD PB                                  560952000 C
      PB(I,J)=PNU2M*P(I,J)+PNU*(PP(I,J)+PTDD(I,J))                       561500000
      CALL OPUT(KFLDS,NWDS,(NDISKS-1)*NWDS+1,PB)                         561750000
```

END

Apr. 25, 1986

If my current address for you is not up to date, please advise. Also, anyone wishing to receive a current version (through this circular) of the model on tape, please notify me. (Mike Cox, Box 308, Princeton, NJ 08540; 609-924-7530)

The main feature of this circular is an update which allows those running in core-contained mode to reduce their memory requirements significantly. Please replace 2 pages of your manual with the enclosed 2 sheets which reflect this.

The following update should be made permanently to the base code.

PERMANENT UPDATE:

```
 -156400000
        NDISKB=MOD(ITT+2,3)+1                                          205900000 -K
        NDISK =MOD(ITT  ,3)+1                                          206000000 -K
        NDISKA=MOD(ITT+1,3)+1                                          206100000 -K
        NDISKB=MOD(ITT+1,2)+1                                          206110000 K
        NDISK =MOD(ITT  ,2)+1                                          206120000 K
        NDISKA=NDISKB                                                  206130000 K
C   UNUSED UNIT ("NDISKB") AND TRANSFERRED TO THE PROPER UNIT ("NDISKA")  278100000 C-K
C   LATER.  RETURN TO THE TOP OF "STEP" TO DO THE 2ND PASS.            278200000 C-K
C   "NDISKA" UNIT.  RETURN TO THE TOP OF "STEP" TO DO THE 2ND PASS.    278210000 CK
C                                                                      279400000 C-K
C----------------------------------------------------------------------- 279500000 C-K
C   IF THIS IS THE END OF THE 2ND PASS OF AN EULER BACKWARD TIMESTEP,  279600000 C-K
C   TRANSFER THE DATA WRITTEN TEMPORARILY TO "NDISKX" TO ITS FINAL     279700000 C-K
C   DESTINATION (THE ORIGINAL "NDISKA").                               279800000 C-K
C----------------------------------------------------------------------- 279900000 C-K
C                                                                      280000000 C-K
      IF(MXP.EQ.1) THEN                                                280100000 -K
        NDISKA=NDISK                                                   280200000 -K
        NDISK=NDISKB                                                   280300000 -K
        DO 394 J=2,JMTM1                                               280400000 -K
          CALL DGET(LABS(NDISKX),NSLAB,(J-1)*NSLAB+1,TA)               280500000 -K
          CALL OPUT(LABS(NDISKA),NSLAB,(J-1)*NSLAB+1,TA)               280600000 -K
 394    CONTINUE                                                       280700000 -K
      ENDIF                                                            280800000 -K
      CALL OPUT(KFLDS,NWDS,MOD(ITT+1,3)*NWDS+1,P)                      281500000
C       SET VERTICAL VELOCITY AT KMP1 TO ZERO                          320510000 C
        W(I,KMP1)=FX                                                   320910000
      DO 250 K=1,KMM1                                                  321400000
      DO 255 K=1,KMM1                                                  322200000
      DO 70 JND=1,2                                                    613500000
         IF (JND.EQ.1) THEN                                            613700000
            TDIO(I,K,JND)=TDQ(I,KREF)                                  614400000
            SDIO(I,K,JND)=SDQ(I,KREF)                                  614500000
            CIO(I,K,N,JND)=CQ(I,KREF,N)                                614900000
         IF (JND.EQ.2) THEN                                            615200000
            TDIO(I,K,JND)=TDQ(I,KREF)                                  615900000
            SDIO(I,K,JND)=SDQ(I,KREF)                                  616000000
            CIO(I,K,N,JND)=CQ(I,KREF,N)                                616400000
     * NSL=((NT+2)*IMTKM+LBC*IMT)*JMT,NTB=N11+N12+2*NSL)               640900000 K
      IF(LU.GE.13) NS=NSKP+(2*((NFRST-1)/NWRS)+(LU-13))*NWRS+1         644100000 K
      IF(LU.GE.13) NS=NSKP+(2*((NFRST-1)/NWRS)+(LU-13))*NWRS+1         645700000 K
```

Explanatory remarks:

156400000-281500000 & 640900000-645700000

When running in core-contained (K option) mode, it is not necessary to save all three time levels of the prognostic variables in separate areas of the virtual disk "BIG". This update causes the tau+1 data to be written over the tau-1 area, thereby reducing the size of "BIG" by almost one third. No change in answers. Note that when running in disk mode, this procedure would cause I/O channel interference which could result in severe reduction in wall-clock efficiency. It is therefore coded for K option only. (Thanks to R. Slater, GFDL)

320510000-322200000

The vertical velocity is computed to be zero at K=KMP1 only within the limits of the precision of the machine. It is later multiplied by U and V at KMP1, addresses which are outside their dimensions. The possibility therefore exists for erroneous answers under certain circumstances. This update eliminates that possibility. Unless the trailing neighbor array of V was changed by the local programmer, it is very unlikely that this change will result in a significant change in answers. No change for the three sample programs. (Thanks to M. MacVean, U.K. Met. Office)

(over)

613500000-616400000

Using the variable "IND" in entry "STINIT" as a DO LOOP index and in entry "STATEC" as an argument is not acceptable to some compilers. This update eliminates that potential problem. (Thanks to M. MacVean, U.K. Met. Office)

GLOBAL UPDATE: Update to accelerate convergence of lower levels.
Submitted by M. Cox, GFDL

Notes: 1. This update allows one to use longer time-steps at lower levels to speed convergence. It should only be used when the final solution will not be altered by this procedure.

2. In its present configuration, the multiplicative factor increases from 1 at the surface to 60 at 5000 meters on an exponential curve and remains at this value below 5000 m. Lines 119400010-50 may be changed to effect different configurations.

```
/  UPDATE TO ACCELERATE LOWER LEVELS ********************************
      *,DTX(KM)                                                          005400070
     * DTXO    (IMT,KM),DZXO    (IMT,KM),ZZ2RXO   (IMT,KM),              008600070
      X5K=60.                                                            119400010
      BX=LOG(X5K)/5.E5                                                   119400020
      DO 12 K=1,KM                                                       119400030
        DTX(K)=MIN(EXP(BX*ZDZZ(K)),X5K)                                  119400040
 12   CONTINUE                                                           119400050
        DTXR=1./DTX(K)                                                   208800070
        DTXO    (I,K)=DTX(K)                                             210100070
        DZXO    (I,K)=DZ(K)*DTXR                                         210100072
      FX=1.                                                              210200070
      DO 185 K=2,KM                                                      210200072
        ZZ2RXO(*,K)=FX/(DZXO(*,K-1)+DZXO(*,K))                           210200074
 185  CONTINUE                                                           210200076
        TA(I,K,M)=(TB(I,K,M)+C2DTTS*DTXO(I,K)*TA(I,K,M))*FM(I,K)         420900000 -H
        TFULL(I,K)=TB(I,K,M)+EXTEND(C2DTTS)*DTXO(I,K)*TA(I,K,M)          421300000 H
            TA(I,K,M)=(DZXO(I,K)*TA(I,K,M)+DZXO(I,K+1)*TA(I,K+1,M))*     430700000 -W
     *                                                   ZZ2RXO(I,K+1)   430701000 -W
          TEMPB(1,1;IMTKM)=DZXO(1,1;IMTKM)*TA(1,1,M;IMTKM)               431500000 W
     *                     *ZZ2RXO(1,KS+1;LN)                            431900000 W
          TDIF(I,K,M)=TDIF(I,K,M)/DTXO(I,K)                              440300070
```

GLOBAL UPDATE: Update for a single state variable.
Submitted by M. Cox, GFDL

Notes: 1. This update allows one to run with an "apparent temperature", with changes to the "STATE" subroutine to yield an equation of state of the type:

rho = 1.0 - 2.5E-4 * (apparent temperature)

```
/  UPDATE FOR A SINGLE STATE VARIABLE  ******************************
      * XXXXXX
-8800000                                                                 8700000
-600200000,601900000
-602100000,602200000
-603000000,604000000
-604700000,605400000
        FXA=-2.5E-4
          RHO(I,K)=FXA*TX(I,K)                                           605400010
-605800000,606200000                                                     605700000
-606900000,608800000
        FXA=-2.5E-4
          RHO(I,L)=FXA*TX(I,L)                                           609000010
-609400000,609800000                                                     609300000
-610500000,616600000
```

Present distribution of this circular: 46

Australia	2	Italy	1
Belgium	1	Japan	2
Canada	4	Switzerland	1
Denmark	1	USA	22
England	5	West Germany	4
France	3		

END

GFDL OCEAN MODEL CIRCULAR NO. 6

Dec. 15, 1986

If my current address for you is not up to date, please advise. Also, anyone wishing to receive a current version (through this circular) of the model on tape, please notify me. (Mike Cox, Box 308, Princeton, NJ 08540; 609-924-7530)

This circular contains one permanent update that corrects a minor bug in the base code, a minor correction to the documentation, two optional ("global") updates and other comments that may be helpful.

The following update should be made <u>permanently to the base code</u>.

PERMANENT UPDATE:

```
      LENV=(JSCAN-2)*IMT                                          550210000 Q
      RESMAX=MAX(ABS(Q8SMAX(RES(1,3;LENV))),                      550300000 Q
     *           ABS(Q8SMIN(RES(1,3;LENV))))                      550400000 Q
```

Explanatory remarks:

xxxxxxxxx-xxxxxxxxx

This corrects a bug in the base code when running under both Q and I options (at the same time), and rows 2 or JMTM1 contain island neighboring ocean points. Before this fix, it is possible that, under the above circumstances, the convergence of the relaxation in RELAX could have been measured erroneously, resulting in the execution of more scans than necessary to reach the prescribed convergence criterion.

Manual correction:

On p. C7 of the documentation there should be an m^2 immediately following the equal sign in Eq. 54.

GLOBAL UPDATES: There are two new update modules available which may be useful for some applications. They are both of considerable length and will not be listed here, but anyone wishing to receive them may do so by contacting me as directed above.

1. Update for biharmonic mixing:
   This module replaces the previous LaPlacian type mixing on velocity and tracers with mixing of biharmonic form, useful in eddy-resolving studies. It contains code to keep 5 instead of 3 rows in memory at once.

2. Update for wind-forced mixing:
   This module is an implementation of the Kraus-Turner bulk mixed-layer model. It takes a given KE input and converts it to PE via mixing within the upper layers.

(over)

List of global update modules available:

Update	Circular
Robert time filter	4
Accelerate convergence of lower levels	5
Single state variable	5
Biharmonic mixing	6
Wind-forced mixing	6

Most of these modules are available either on floppy disc or tape from me at the address above. Anyone who has developed a module of this type that may be of use to others in the modelling community, and is willing to make it available is invited to do so through this circular.

For those non-CDC users who don't want the *DECK lines to be passed by UPDOC to the compiler, you can get rid of them by placing the following line in the UPDOC code after the statement labelled 322:

```
IF(LN(1:5).EQ.'*DECK') GO TO 320
```

```
**********************
*                    *
*                    *
*   HAPPY HOLIDAYS   *
*                    *
*                    *
**********************
```

END

GFDL OCEAN MODEL CIRCULAR NO. 7

Jan. 15, 1987

If my current address for you is not up to date, please advise. Also, anyone wishing to receive a current version (through this circular) of the model on tape, please notify me. (Mike Cox, Box 308, Princeton, NJ 08540; 609-452-6531)

The purpose of this circular is to announce the availability of an update module which implements mixing of all "T" quantities (temperature, salinity, passive tracers) along isopycnals in the model. The theoretical basis for this algorithm is presented in:

Redi (1982): Oceanic isopycnal mixing by coordinate rotation. J. Phys. Oceanogr., 12, 1154-1158.

The mixing tensor derived there is simplified somewhat using scale analysis. For numerical stability, a constraint, read in through namelist input data, is imposed upon the maximum slope of the mixing vector. To relax this constraint somewhat, the "zz" term of the tensor is calculated fully implicitly. For most applications, mixing vector slopes of at least ρ_x/ρ_z=1/100 should be attainable without reducing the timestep.

This module has been in use at GFDL for several months on an experimental basis with promising results. Due to its computational complexity, a significant decrease in running speed is realized in the overall model. However, the improvement in the solution in various aspects may more than justify the additional investment.

Anyone wishing to obtain this (approx. 400 line) module may do so by contacting me at the above address. Standard ASCII text file on a PC-DOS floppy disc is the preferrable mode of transfer, although 9-track tape is possible as well. Please specify.

List of global update modules available:

Update	Circular
Robert time filter	4
Accelerate convergence of lower levels	5
Single state variable	5
Biharmonic mixing	6
Wind-forced mixing	6
Isopycnal mixing	7

A note on using update modules under "UPDOC". Update modules may be included in the unit 10 input data to UPDOC as separate entities. After the initial block of updates with the "OPT=" card leading, additional modules may be included separately, with "/" cards leading:

```
OPT=I,K,C
           ......                              #########
           ......                              #########
           ......                              #########
/ UPDATE FOR XXXXX
           ......                              #########
           ......                              #########
/ UPDATE FOR YYYYY
           ......                              #########
           ......                              #########
etc.
```

Care should be taken that there is no interference between update modules.

END

GFDL OCEAN MODEL CIRCULAR NO. 8

Sept. 9, 1987

If my current address for you is not up to date, please let me know. Anyone wishing to receive a current version (through this circular) of the model on tape, please notify me. (Mike Cox, Box 308, Princeton, NJ 08540; 609-452-6531)

On pages B2-3 of the documentation, convection within a statically unstable water column is described as a special case of the vertical diffusion process, wherein the coefficient goes to infinity. In the model, this is accomplished by a two-pass process in which first odd levels are mixed with their lower neighbor if unstable, and then even levels also conditionally mixed with their lower neighbor. This process can, obviously, leave residual instability behind for odd relative to even levels, and to bring the process toward completion, it is necessary to perform multiple passes (NCON>1). In practice, particularly for KM>10 or so, it is difficult for this algorithm to mix the levels vigorously enough to maintain physically reasonable static stability. In addition, the resultant degree of stabilization is a function of timestep length with shorter timesteps resulting in relatively more complete stabilization. The following module replaces the above system with an algorithm which performs the process described in Eq (12) more literally. Convection is integrated into the diffusive process by adopting a large coefficient of mixing in statically unstable regions.

To accomodate the large coefficient without exceeding numerical constraints, it is necessary to evaluate the vertical diffusion term implicitly. The scheme described in Richtmyer and Morton, 1967 is used for this purpose. A coefficient of 10^4 cm^2 s^{-1} is found to maintain stability quite well, although this value can be adjusted for preference.

Besides maintaining stability better, the new system conducts convection in a more continuous, less episodic manner resulting in a generally lower noise level within the solution in some cases. Although some additional memory is required for the implicit treatment, the new scheme is generally faster than a multi-pass application of the old scheme. This is primarily because the equation of state must be calculated only once per grid point, whereas the old scheme requires a new call to STATEC for each double pass. I recommend trying the new module in any model in which convection may be a critically important process. A copy on PC-DOS floppy disk is available through me.

```
/ UPDATE FOR IMPLICIT VERTICAL DIFFUSION & CONVECTION
     * EEMQ  (IMT,KM),FFMQ  (IMT,KM),FK3(IMT,KM,3:3),                  8600000
-209800000
-210000000
      REAL AA(IMT,KM),BB(IMT,KM),CC(IMT,KM),                           402610000
     *     EE(IMT,KMP1),FF(IMT,KMP1,NT),EFDR(IMT)                      402611000
      DIMENSION RZ(IMT,KMP1)                                           402612000
      EQUIVALENCE (EE,TDIF),(AA,RZ)                                    402613000
C  ELIMINATE ROUNDOFF ERROR IN W AT THE BOTTOM                         407800000 C
-408100000,409200000
-409700000,409900000
-418300000,420100000
C                                                                      422120200 C
C-----------------------------------------------------------------     422120400 C
C   COMPUTE VERTICAL DIFFUSION OF TRACERS IMPLICITLY                   422120600 C
C     & STABILIZE THE WATER COLUMN VERTICALLY                          422120800 C
C-----------------------------------------------------------------     422121000 C
C                                                                      422121200 C
C   1ST, COMPUTE THE VERTICAL DENSITY GRADIENT OF THE WATER COLUMN     422121400 C
C                                                                      422121600 C
      CALL STATEC(TB ,TB (1,1,2),TEMPA,TDIF,TDIF(1,1,2),1)             422121900
      CALL STATEC(TB ,TB (1,1,2),TEMPB,TDIF,TDIF(1,1,2),2)             422122000
      DO 860 K=2,KM,2                                                  422122200
      DO 860 I=1,IMT                                                   422122400
        RZ (I,K)=TEMPA(I,K-1)-TEMPA(I,K)                               422122600
        RZ (I,K+1)=TEMPB(I,K)-TEMPB(I,K+1)                             422122800
 860  CONTINUE                                                         422123000
C                                                                      422123200 C
C   2ND, SET THE COEFFICIENT OF VERTICAL DIFFUSIVITY                   422123400 C
C        (SET LARGE IN CASE OF STATIC INSTABILITY)                     422123600 C
C                                                                      422123800 C
      DO 862 K=2,KM                                                    422124000
      DO 862 I=1,IMT                                                   422124200
        FK3(I,K,3)=FKPH                                                422124400
 862  CONTINUE                                                         422124600
      FXA=10000.                                                       422124800
      DO 863 K=2,KM                                                    422125000 -W
      DO 863 I=1,IMT                                                   422125200 -W
        IF(RZ(I,K).GE.0.0) FK3(I,K,3)=FXA                              422125400 -W
 863  CONTINUE                                                         422125600 -W
      WHERE(RZ(1,2;IMT*KMM1).GE.0.0)FK3(1,2,3;IMT*KMM1)=FXA            422125800 W
C                                                                      422126000 C
C   3RD, SOLVE THE IMPLICIT DIFFUSION EQUATIONS                        422126200 C
C        (AS IN RICHTMYER AND MORTON, 1967, PAGE 198... )              422126400 C
C                                                                      422126600 C
      FXA=4.0                                                          422126800
      FXB=1.0                                                          422127000
      FXC=0.0                                                          422127200
```

(over)

```
      DO 865 I=1,IMT                                              422127400
        AA(I,    1)=FXC                                           422127600
        CC(I,   KM)=FXC                                           422127800
        EE(I,KM+1)=FXC                                            422128000
865   CONTINUE                                                    422128200
      DO 866 M=1,NT                                               422128400
      DO 866 I=1,IMT                                              422128600
        FF(I,KM+1,M)=FXC                                          422128800
866   CONTINUE                                                    422129000
      DO 867 K=1,KM                                               422129200
      DO 867 I=1,IMT                                              422129400
        TEMPA(I,K)=C2DTTS*DZ2RO(I,K)*FXA*FM(I,K)                  422129600
867   CONTINUE                                                    422129800
      DO 868 K=2,KM                                               422130000
      DO 868 I=1,IMT                                              422130200
        TEMPB(I,K)=FK3(I,K,3)*DZZ2RO(I,K)                         422130400
868   CONTINUE                                                    422130600
      DO 869 K=2,KM                                               422130800
      DO 869 I=1,IMT                                              422131000
        AA(I,K)=TEMPA(I,K)*TEMPB(I,K)                             422131200
869   CONTINUE                                                    422131400
      DO 870 K=1,KMM1                                             422131600
      DO 870 I=1,IMT                                              422131800
        CC(I,K)=TEMPA(I,K)*TEMPB(I,K+1)*FM(I,K+1)                 422132000
870   CONTINUE                                                    422132200
      DO 871 K=1,KM                                               422132400
      DO 871 I=1,IMT                                              422132600
        BB(I,K)=AA(I,K)+CC(I,K)+FXB                               422132800
871   CONTINUE                                                    422133000
      DO 874 K=KM,1,-1                                            422133200
        DO 872 I=2,IMTM1                                          422133400
          EFDR(I)=FXB/(BB(I,K)-CC(I,K)*EE(I,K+1))                 422133600
872     CONTINUE                                                  422133800
        DO 873 I=2,IMTM1                                          422134000
          EE(I,K)=AA(I,K)*EFDR(I)                                 422134200
873     CONTINUE                                                  422134400
      DO 874 M=1,NT                                               422134600
      DO 874 I=2,IMTM1                                            422134800
        FF(I,K,M)=(TA(I,K,M)+CC(I,K)*FF(I,K+1,M))*EFDR(I)         422135000
874   CONTINUE                                                    422135200
      DO 877 M=1,NT                                               422135400
        DO 875 I=2,IMTM1                                          422135600
          BB(I,1)=FF(I,1,M)                                       422135800
875     CONTINUE                                                  422136000
        DO 876 K=2,KM                                             422136200
        DO 876 I=2,IMTM1                                          422136400
          BB(I,K)=EE(I,K)*BB(I,K-1)+FF(I,K,M)                     422136600
876     CONTINUE                                                  422136800
        DO 877 K=1,KM                                             422137000
        DO 877 I=1,IMT                                            422137200
          TA(I,K,M)=BB(I,K)*FM(I,K)                               422137400
877   CONTINUE                                                    422137600
-422200000,423500000
-426700000,427200000
-428400000,432330000
```

There are the following interactions with modules covered in earlier circ.:

```
1. NEEDED WHEN RUNNING MODULE TO ACCELERATE LOWER LEVELS ***********************************
   *       *DTXO(I,K)
                                                                  422129610
2. NEEDED WHEN RUNNING WIND FORCED MIXING MODULE *******************************************
C                                                                 422125810
C  (DO NOT APPLY DIFFUSION WITHIN WIND MIXED LAYER)               422125815
C                                                                 422125820
      FXA=2.0                                                     422125825
      FXB=1.0                                                     422125830
      FXC=0.0                                                     422125835
      DO 8631 K=2,KMWFM                                           422125840
      DO 8631 I=2,IMTM1                                           422125845
        TEMPA(I,K)=FXB-(DML(I,J)-ZDZ(K-1))*DZ2R(K)*FXA            422125850
8631  CONTINUE                                                    422125855
      DO 8632 K=2,KMWFM                                           422125860 -W
      DO 8632 I=2,IMTM1                                           422125865 -W
        IF(TEMPA(I,K).GT.FXB) TEMPA(I,K)=FXB                      422125870 -W
        IF(TEMPA(I,K).LT.FXC) TEMPA(I,K)=FXC                      422125875 -W
8632  CONTINUE                                                    422125880 -W
      WHERE(TEMPA(1,2;IMT*(KMWFM-1)).GT.FXB)TEMPA(1,2;IMT*(KMWFM-1))=FXB   422125885 W
      WHERE(TEMPA(1,2;IMT*(KMWFM-1)).LT.FXC)TEMPA(1,2;IMT*(KMWFM-1))=FXC   422125890 W
      DO 8633 K=2,KMWFM                                           422125895
      DO 8633 I=1,IMT                                             422125900
        FK3(I,K,3)=TEMPA(I,K)*FK3(I,K,3)                          422125905
8633  CONTINUE                                                    422125910
```

Those of you using the isopycnal diffusion module may adopt this scheme by inserting the 863 loop of the module above after the 250 loop of that module, and adding the -428400000,4323300000 directive.

Manual corrections:

1. On p. C7 of the documentation, line 2 of Eq (55), the leading sign should be +, and both "a"s should be squared.

2. The integrated contribution from the Newtonian boundary condition in the update modules for model examples 2 and 3 should be outside of the K loop instead of inside as positioned at 427200010... Also, they should be multiplied by DZ(1). This affects only the "SURFACE FLUX" and "TRUNCATION ERROR" diagnostics on p. L21, L39 and no other answers, and is a bug in the update modules only. Thanks to the two users who pointed this out.

END

GFDL OCEAN MODEL CIRCULAR NO. 9

Feb. 18, 1988

If my current address for you is not up to date, please let me know. Anyone wishing to receive a current version (through this circular) of the model on tape, please notify me. (Mike Cox, Box 308, Princeton, NJ 08540; 609-452-6531)

The elliptic equation relating the mass transport stream function to the vorticity calculated in CLINIC is currently solved in RELAX using the method of successive over relaxation. Since, in this scheme, updating of the field being relaxed is done immediately when the residual is calculated, the western or I-1 segment of the relaxation star cannot be vectorized. This presents a serious problem for the overall efficiency of the code. In some model runs where many (O(100)) scans are needed for convergence, the relaxation process may take a considerable part of the total cp seconds per timestep, with much of it tied up in the scalar I-1 component of the star. One alternative is simultaneous relaxation in which updating of the field being relaxed is done after all residuals are calculated. This has the advantage of being completely vectorizable, but takes so many more scans to convergence that its speed advantage is more than cancelled by additional computation.

A new scheme has been developed by John Derber at GFDL, using the technique of conjugate gradients, which has the advantage of being completely vectorizable but generally does not require more scans for convergence to equal accuracy. Preliminary tests indicate that 10-20% and more can be saved in overall model running time with the use of this scheme, depending on the number of scans being used presently. Tests also indicate that convergence using the conjugate gradient technique is somewhat more adversely affected by sharply changing bottom topography (particularly shallow areas), or implicit treatment of external beta waves (ACOR>0.) The advantage of the scheme may be reduced if such conditions exist.

An update module in UPDOC format has been prepared and is available to implement the scheme. Since it is roughly 200 lines in length, I have not listed it here but will make it available on PC-DOS floppy disk to anyone requesting it. J. Derber has a short manuscript describing the technique and will provide it upon request. Thanks to Ron Pacanowski who helped expand the update to UPDOC format for general use.

There are various other major code supplements currently being developed by different users, some I am aware of and possibly others I am not. One involves a major change in the treatment of the external mode in which the rigid-lid constraint is relaxed, and the free surface height is calculated by means of multiple sub-timesteps within the main timestep. This technique will probably become more attractive as we enter the realm of eddy resolution where, from experience so far, the relaxation scan count can become large. Carrying surface elevation as a prognostic variable also has the advantage of facilitating the utilization of altimetry data.

Another effort involves the implementation of multi-tasking in the model for use on multi-processing machines. There are various ways in which multi-tasking can be implemented, and opinion currently appears to be in a state of flux as to what is the most appropriate. How will charging algorithms be written? What mix of multi-processing/multi-programming will facility managers prefer? What degree of auto-tasking will compilers provide? Is there any prospect of having a universal multi-tasking protocol? How much redundant calculation or supplementary memory is justified to achieve a given level of multi-tasking in terms of the optimization of total machine usage? The answers to these questions will largely determine the optimum multi-tasking approach. Anyone who has been or will be engaged in efforts to multi-task the code is invited to make their update modules available through this circular. I know of at least one such effort.

Likewise, anyone who has developed a code supplement of any type which may be of use to other modelers is invited to make it known through this circular.

END

GFDL OCEAN MODEL CIRCULAR NO. 10

April 22, 1988

If my current address for you is not up to date, please notify me. Also, anyone wishing to receive a current version (through this circular) of the model on tape, please notify me. (Mike Cox, Box 308, Princeton, NJ 08540; 609-452-6531)

This circular contains three permanent updates, two of which correct minor bugs in the base code, and one that generalizes the timing code somewhat.

The following updates should be made <u>permanently to the base code</u>.

<u>PERMANENT UPDATE 1</u>:

```
      IF(MXP.EQ.0) THEN                                    561110000
      ELSE                                                 561710000
      DO 342 J=1,JMT                                       561720000
      DO 342 I=1,IMT                                       561730000
        P(I,J)=PB(I,J)+PTD(I,J)                            561740000
342   CONTINUE                                             561750000
      ENDIF                                                561760000
```

<u>Explanatory remarks</u>:

This corrects a bug in the base code when running with EB=.TRUE. (Euler backward mixing timestep.) In the uncorrected code, the stream function at tau-1 (PB) is updated on both passes of the Euler backward timestep. This update corrects the code so that PB is updated only on the first pass, as is required in the Euler backward formulation. The error occurs only on mixing timesteps and is found, with recent test runs, to be generally insignificant except when the stream function is changing rapidly in time. Such conditions may exist when P contains high frequency waves, or at the beginning of a spin-up when P is developing rapidly. For this reason, the answers in Examples 2 and 3 of the documentation are changed somewhat. Solutions for experiments which have been brought to reasonably steady state with only temporal variations remaining which are well resolved by the timestep used should not be significantly affected by this change. Since removal of the error causes P to evolve more smoothly across mixing timesteps, the scan count tends to remain somewhat lower in the corrected code. Thanks to C. Wübber at AWI, Bremerhaven, FRG for finding this bug and making it known.

<u>PERMANENT UPDATE 2</u>:

```
      DO 168 M=1,NTMIN2                                    165300000
```

<u>Explanatory remarks</u>:

This corrects a bug in the base code when running with NT=1 where an "uninitialized variable" error may occur due to initializing only half of the TDIF array. No change in answers.

(over)

PERMANENT UPDATE 3:

```
      T3=SECOND()                                                 106700000 T
      TO=SECOND()                                                 165800000 T
      TIME(1)=TO-T3                                               165810000 T
      T1=SECOND()                                                 168300000 T
-203200000
      TO=SECOND()                                                 203800000 T
-203900000
      PRINT 110,TO                                                204000000 T
 110  FORMAT (' ACC TIME(SEC)=',F10.2)                            204100000 T
      TO=SECOND()                                                 246900000 T
      T1=SECOND()                                                 247900000 T
      TO=SECOND()                                                 248400000 T
      TO=SECOND()                                                 276600000 T
      T1=SECOND()                                                 277400000 T
      TO=SECOND()                                                 525900000 T
      T1=SECOND()                                                 560500000 T
      TO=SECOND()                                                 604500000 T
      T1=SECOND()                                                 606400000 T
      TO=SECOND()                                                 609000000 T
      T1=SECOND()                                                 610000000 T
      TO=SECOND()                                                 705700000 FT
      T1=SECOND()                                                 731800000 FT
```

Explanatory remarks:

This replaces the non-standard timing call "GETIME" with the more standard function "SECOND" making the use of option T possible for more users. No change in answers.

END