Muir & Hamblin

Environment Canada
Environnement Canada

Canada Centre For Inland Waters
Centre Canadien Des Eaux Intérieures

THE COMPUTATION OF VERTICAL STRUCTURE ASSOCIATED WITH INTERNAL GRAVITY WAVES

by

L. R. MUIR* and P. F. HAMBLIN**

UNPUBLISHED REPORT
RAPPORT NON PUBLIE

THE COMPUTATION OF VERTICAL STRUCTURE ASSOCIATED WITH INTERNAL GRAVITY WAVES

by

L. R. MUIR* and P. F. HAMBLIN**

December, 1977

* Ocean and Aquatic Sciences, Central Region
** National Water Research Institute

ABSTRACT

Two numerical methods are presented for determining the vertical structure of an arbitrarily stratified water column in the presence of internal gravity waves. The development is entirely self-contained and contains listings for Fortran IV computer routines which incorporate the theory presented. The two methods were developed for very different specific applications although they produce identical answers. Method 1 allows the specification of the forcing frequency, but Method 2 assumes a very long period wave, in both cases an arbitrary stable density distribution is allowed to be specified at arbitrary depths.

ACKNOWLEDGMENTS

Karen Beal assisted in the computer programming for Method 2. S.J. Prinsenberg supplied a copy of notes written from M.J. Rattray's class lectures outlining the theory for Method 1.

TABLE OF CONTENTS

CHAPTER 1

1.0 Introduction

In the course of investigation into the properties of internal waves in the oceans and lakes, the authors found no specific references to numerical means of calculating vertical structures and phase speeds of internal waves from actual field data. This report is intended to provide the researcher in this field with a set of tested and documented computer programmes to calculate these vertical properties. Although the theory of internal gravity waves upon which our development is based and also the numerical methods that we have applied to the problems are not novel, we feel that the report will, nonetheless, be of use to those who are faced with the practical problem of calculating internal wave properties from field observations.

In this report, we offer two numerical methods which have been developed for determining the vertical velocity structure of a water column in the presence of internal gravity waves. Although the equations for the vertical velocity structure are developed elsewhere [Phillips, 1966; Roberts, 1975] and the physical interpretations are fairly well known, it was decided to make this report self-contained insofar as the mathematics are concerned. This ensures a consistent notation and allows the clear statement of the assumptions and the manipulations in the development of the equations.

The two methods were derived for very different applications although, from a number of analytical and observational test cases computed by the two methods, the answers produced are identical. Method 1 allows specification of the forcing frequency. Method 2 assumes a very long period wave in comparison to the Brunt-Vaisala period. Although Method 1 could be used to compute dispersion relationships, it should be noted that there exists a computer programme to do this specific task [Bell, 1971]. It should also be noted that Roberts [1975] contains a number of analytical solutions that can be used to test these programmes.

CHAPTER 2

2.0 The Equations of Motion

We assume a right-handed Cartesian coordinate system (x,y,z,t) with the z-axis positive downwards. The three components of instantaneous velocity are u*,v*,w*, respectively, so that the total velocity is given by:

$$\vec{u} = u^*\hat{i} + v^*\hat{j} + w^*\hat{k}$$

where $\hat{i},\hat{j},\hat{k}$ are unit vectors. The pressure, P*, and the density, ρ^*, are functions of x,y,z,t and so, in vector form, the continuity equation is:

$$\frac{D\rho^*}{Dt} + \rho^*(\nabla\cdot\vec{u}) = 0 \qquad (2.1)$$

The momentum equation, or the Navier-Stokes equation, with Coriolis force and arbitrary body force, $\vec{F}$, is given by [Batchelor, 1967, p. 147]:

$$\rho^* \frac{D\vec{u}}{Dt} + 2\rho^*(\vec{\Omega}\times\vec{u}) + \nabla P^* - g\rho^*\hat{k} = \mu\nabla^2\vec{u} + \frac{\mu}{3}\nabla(\nabla\cdot\vec{u}) + \vec{F} \qquad (2.2)$$

where $\vec{\Omega}$ is the angular velocity of the earth and μ is a coefficient of viscosity.

If we exclude sound or compression waves from further consideration, then compressibility plays no part in the continuity equation, and hence:

$$\frac{D\rho^*}{Dt} = 0 \qquad (2.3)$$

so that the continuity equation is:

$$\nabla\cdot\vec{u} = 0 \qquad (2.4)$$

and hence the second term on the righthand side of (2.2) disappears.

We now assume that:

(a) There is no mean flow in any direction, or equivalently,

that, if there is a mean flow, the coordinate system moves with this mean flow.

(b) There are no arbitrary body forces such as friction acting on the fluid, so that $\vec{F} = 0$.

(c) The fluid is inviscid so that $\mu = 0$. This is equivalent to an assumption that the Reynolds number is large.

(d) Both the pressure and the density are composed of a mean value plus a fluctuating part so that:

$$P^* = \bar{P}(z) + P(x,y,z,t)$$
$$\rho^* = \bar{\rho}(z) + \rho(x,y,z,t)$$

Then the momentum equation becomes:

$$(\bar{\rho}+\rho)\frac{D\vec{u}}{Dt} + 2(\bar{\rho}+\rho)(\vec{\Omega}\times\vec{u}) + \nabla(\bar{P}+P) - g(\bar{\rho}+\rho)\hat{k} = 0$$

We make the further assumptions that:

(a) We may neglect the fluctuation in density in all terms except the gravitational term. This is the Boussinesq approximation and has been discussed extensively in Long [1965] and elsewhere. This assumption if consistently made, eliminates the possibility of some solitary wave solutions.

(b) We neglect all nonlinear terms of the form $\vec{u}\frac{\partial\vec{u}}{\partial x}$ and so any waves must be small-amplitude waves.

(c) The horizontal components of the Coriolis force may be neglected and $2\vec{\Omega} = (o,o,f)$.

Then the linear, inviscid, Boussinesq equations of motion in component form are:

$$\frac{\partial u}{\partial t} - fv + \frac{1}{\rho}\frac{\partial P}{\partial x} = 0 \qquad (2.5)$$

$$\frac{\partial v}{\partial t} + fu + \frac{1}{\rho}\frac{\partial P}{\partial y} = 0 \qquad (2.6)$$

$$\frac{\partial w}{\partial t} + \frac{1}{\bar{\rho}} \frac{\partial P}{\partial z} - \frac{g\rho}{\bar{\rho}} = 0 \tag{2.7}$$

$$\frac{\partial \rho}{\partial t} + w \frac{\partial \bar{\rho}}{\partial z} = 0 \tag{2.8}$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \tag{2.9}$$

Two alternate methods will now be given for the derivation of an eigenvalue problem.

CHAPTER 3

3.0 Method 1

The development of the vertical structure equation uses classical arguments [Phillips, 1966] to derive the vertical velocity equation. The finite difference scheme and iteration process is initially from a hand written manuscript by P. Kovaala supplied by S.J. Prinsenberg although it is discussed in a more general fashion and made more rigorous here. The problem is also discussed in Keller [1968] in a more general form.

3.1 The Vertical Structure Equation

From the linearized, inviscid, Boussinesq equations, we first differentiate (2.5) and (2.6) with respect to z and add them to get:

$$\frac{\partial^2 u}{\partial z \partial t} + f\frac{\partial u}{\partial z} + \frac{1}{\bar{\rho}}\frac{\partial^2 P}{\partial x \partial z} + \frac{\partial^2 v}{\partial z \partial t} - f\frac{\partial v}{\partial z} + \frac{1}{\bar{\rho}}\frac{\partial^2 P}{\partial y \partial z} = 0 \qquad (3.1)$$

Then, differentiating (2.7) with respect to x and (2.8) with respect to y and subtracting the two equations successively from (3.1), we have eliminated the pressure to get:

$$\frac{\partial^2 u}{\partial z \partial t} + \frac{\partial^2 v}{\partial z \partial t} - \frac{\partial^2 w}{\partial x \partial t} - \frac{\partial^2 w}{\partial y \partial t} + \frac{g}{\bar{\rho}}\frac{\partial \rho}{\partial x} + \frac{g}{\bar{\rho}}\frac{\partial \rho}{\partial y} - f\left(\frac{\partial v}{\partial z} - \frac{\partial u}{\partial z}\right) = 0 \qquad (3.2)$$

We now differentiate this equation with respect to time, use equation (2.8) and take the horizontal divergence $\nabla_h = \frac{\partial}{\partial x} + \frac{\partial}{\partial y}$ of the resulting equation to get:

$$\frac{\partial^2}{\partial t^2}(\nabla^2 w) + N^2 \nabla_h^2 w + f\nabla_h \frac{\partial}{\partial t}\left(\frac{\partial v}{\partial z} - \frac{\partial u}{\partial z}\right) = \frac{\partial^2}{\partial t^2}\nabla_h\left(\frac{\partial u}{\partial z} + \frac{\partial v}{\partial z}\right)$$

where $N^2 = \frac{g}{\rho}\frac{\partial \rho}{\partial z}$ is the square of the Brunt-Vaisala frequency.

For small disturbances in the absence of a mean shear, the right side of this equation is zero since the term in brackets is the mean shear. In addition, the last term on the lefthand side

$$\frac{\partial}{\partial t}\left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right) = f\frac{\partial w}{\partial z}$$

from the vertical component of the vorticity equation if we assume that the fluid is irrotational except for vorticity produced by the Coriolis acceleration. Thus, we have our final equation:

$$\frac{\partial^2}{\partial t^2}\nabla^2 w + N^2\nabla_h^2 w + f^2\frac{\partial^2 w}{\partial z^2} = 0 \qquad (3.3)$$

We assume a separable solution for (3.3) in the form:

$$w(x,y,z,t) = \phi(z)F(x,y)\exp(i\sigma t)$$

and substitute this into (3.3), then after neglecting the nonlinear terms, we get:

$$\frac{1}{\sigma^2-f^2}\frac{\nabla_h \cdot F}{F} = \frac{1}{N^2-\sigma^2}\frac{\phi_{zz}}{\phi} = \frac{-k_h^2}{\sigma^2-f^2}$$

where the separation constant has been chosen from the recognition that the field equation is a non-homogeneous form of LaPlace's equation, and hence the horizontal solutions will be wavelike. The constant, k_h, is a horizontal wave number. The vertical structure equation is, then:

$$\frac{d^2\phi}{dz^2} + \left(\frac{N^2-\sigma^2}{\sigma^2-f^2}\right)k_h^2\,\phi = 0 \qquad (3.4)$$

For a sinusoidal surface disturbance with constant atmospheric pressure, and which has the same wave number as the internal solution, the surface boundary condition is; [Fjeldstad, 1933]

$$\frac{d\phi}{dz} + \left(\frac{k_h^2}{\sigma^2-f^2}\right)g\,\phi = 0 \qquad \text{at } z = 0 \qquad (3.5)$$

At the bottom, we have:

$$\phi = 0 \qquad \text{at } z = H \qquad (3.6)$$

If the relative displacement of an isopycnal is given by $\xi(x,y,z,t)$, then by definition:

$$w(x,y,z,t) = \phi(z)F(x,y)\exp(i\sigma t) = \frac{\partial \xi}{\partial t} \simeq \frac{d\xi}{dt}$$

so that $\xi = \frac{1}{i\sigma} F(x,y)\exp(i\sigma t)\phi(z)$; so that, at a given depth,

$\xi(x,y,z_o,t)$ is proportional to $\phi(z_o)$

If we can assume that in a small region about the origin there is no y-dependence in the motion, so that $F(x,y) = F(x)$ and, if we assume

$$w = F(x)\ \phi(z)\ \exp(i\sigma t)$$
$$u = G(x)\ U(z)\ \exp(i\sigma t)$$

Then, from the continuity equation $\frac{\partial u}{\partial x} = -\frac{\partial w}{\partial z}$, we have:

$$U(z) \int_o^x \frac{dG}{dx}\, dx = -\frac{d\phi}{dz} \int_o^x F(x)dx$$

Thus, at the point x_o,

$$U(z)G(x)\Big|_o^{x_o} = -\frac{d\phi}{dz} F(x)\Big|_o^{x_o}$$

and so the horizontal velocity is proportional to $\frac{d\phi}{dz}$.

3.2 The Eigenvalue Problem

The system of equations (3.4), (3.5), and (3.6) provide a description of the vertical structure of a water column with arbitrary density structure. The system may be non-dimensionalized by defining the following quantities:

$$\bar{z} = z/H$$

$$\bar{\phi}(\bar{z}) = \phi(z)$$

$$\bar{c}^2 = \frac{\sigma^2 - f^2}{K_h^2 g H}$$

Then we have:

$$\frac{d}{dz} = \frac{d}{d\bar{z}} \frac{d\bar{z}}{dz} = \frac{1}{H} \frac{d}{d\bar{z}}$$

$$\frac{d\phi}{dz} = \frac{1}{H} \frac{d\bar{\phi}(\bar{z})}{d\bar{z}}$$

$$\lambda = \frac{1}{c^2} = \frac{1}{gH\bar{c}^2} = \frac{\bar{\lambda}}{gH}$$

$$N^2 = \frac{g}{\rho_o} \frac{d\rho}{dz} = \frac{g}{H\rho_o} \frac{d\rho}{d\bar{z}} = \frac{g}{H} \bar{N}^2$$

$$\sigma^2 = \frac{g}{H} \bar{\sigma}^2$$

$$M = N^2 - \sigma^2 = \frac{g}{H} (\bar{N}^2 - \bar{\sigma}^2) = \frac{g}{H} \bar{M}$$

Then, by substituting these values into the vertical structure equation (3.4), we get:

$$\frac{d^2\bar{\phi}}{d\bar{z}^2} + \bar{\lambda}\bar{M}\bar{\phi} = 0 \qquad (3.7)$$

$$\frac{d\bar{\phi}}{d\bar{z}} + \bar{\lambda}\bar{\phi} = 0 \qquad \text{at } \bar{z} = 0 \text{ (surface)} \qquad (3.8)$$

$$\bar{\phi} = 0 \qquad \text{at } \bar{z} = 1 \text{ (bottom)} \qquad (3.9)$$

which is the equation for a non-homogeneous string with the bottom end rigidly attached and the top end elastically attached. If (3.8) were replaced by the "rigid lid" assumption that:

$$\bar{\phi} = 0 \qquad \text{at } \bar{z} = 0 \qquad (3.10)$$

We would then have a Sturm-Liouville system. However, Courant and Hilbert [1937, p. 410ff] prove two very important theorems which are:

1) If λ_n is the n^{th} eigenvalue of (3.7) under the rigid lid boundary conditions (3.9) and (3.10) and μ_n is the n^{th} eigenvalue of (3.7) under the free surface boundary conditions (3.8) and (3.9), then: $\mu_n \leq \lambda_n$.
2) The assymptotic behaviour of the eigenvalues of equation (3.7) is independent of the boundary conditions and, if n is large enough,

$$\lambda_{n+1} = \frac{(n+1)^2}{n^2} \lambda_n$$

By virtue of Theorem 1 and from the fact that the surface displacements are very small for internal waves, we would expect the system of equations (3.7), (3.8), and (3.9) to be "almost the same" as the Sturm-Liouville system, and we know that for the Sturm-Liouville system it may be proved that:

3) All the eigenvalues are real and positive.
4) The eigenfunctions form a complete, orthogonal set. That is, all possible motions can be produced by linear combinations of the eigenfunctions and, if ϕ_i and ϕ_j are two eigenfunctions, then,

$$\int_0^1 \phi_i \phi_j \, d\bar{z} = 0 \qquad \text{if } i \neq j$$

So, from 1 and 3, we can have confidence that the eigenvalues of the system (3.7), (3.8), and (3.9) will be real and positive and, in addition, we see that by virtue of 2 it will only be necessary to calculate a small number of eigenvalues in order to get approximations to the rest. This last will be very important in Method 2.

3.3 First Approximation to the Eigenvalues

In practical problems, we must have the values of the density distribution at p+1 depths where the p sub-intervals are of equal length $h = 1/p$. If the densities are expressed as σ_t values and if $D\sigma_t$ is the difference between successive σ_t values, then:

$$\bar{M}(k) = H\left[\frac{D\sigma_t}{(1000 + \sigma_t)p} - \frac{\sigma^2}{g}\right] \qquad 1 \le k \le p+1$$

where σ is the frequency of the forcing function and $\bar{M}(1)$ is at the surface and $\bar{M}(p+1)$ is at the bottom.

We can then replace equations (3.7), (3.8), and (3.9) by a corresponding set of finite-difference equations:

$$\frac{1}{h^2}\left(\bar{\phi}(k-1) - 2\bar{\phi}(k) + \bar{\phi}(k+1)\right) + \bar{\lambda}\bar{M}(k)\bar{\phi}(k) = 0$$

$$\frac{1}{h}\left(\bar{\phi}(2) - \bar{\phi}(1)\right) + \bar{\lambda}\bar{\phi}(1) = 0$$

$$\bar{\phi}(p+1) = 0$$

If we then write,

$$a(k) = 1/\bar{M}(k)$$
$$h = 1/p$$
$$\mu = h^2\bar{\lambda}$$

we then have the matrix equation:

$$[A - \mu I]\bar{\phi} = 0$$

where I is the identity matrix and [A] is the matrix:

$$A = \begin{bmatrix} h & -h & & & & & \\ -a_1 & 2a_1 & -a_1 & & & \bigcirc & \\ & -a_2 & 2a_2 & -a_2 & & & \\ & & & & & & \\ & \bigcirc & & & -a_{p-1} & 2a_{p-1} & -a_{p-1} \\ & & & & & -a_p & 2a_p \end{bmatrix}$$

ϕ is a column vector containing the p+1 eigenfunctions and μ is the matrix of eigenvalues of the system.

The matrix A may be diagonalized and the eigenvalues calculated from

$$\bar{\lambda}_k = p^2 \mu_k$$

For this finite-difference approximation the A matrix is tridiagonal and the subroutine to diagonalize the A matrix can make use of this fact. Householder [1964] gives an explanation of the QR transformation and there are a number of other proprietary routines [Grad & Brebner, 1968] which will diagonalize a general matrix; hence the finite-difference approximation to the equations is not unique.

3.4 Further Approximations to the Eigenvalues

If we ignore the bottom boundary condition for the moment, the equations (3.7) and (3.8) have solutions for all eigenvalues $\bar{\lambda}>0$. If we define a function

$$\psi = \frac{d\bar{\phi}}{d\bar{\lambda}}$$

and assume that λ_j^o is an approximation to the j^{th} eigenvalue of equations (3.7) and (3.8), we may write the next approximation as:

$$\bar{\lambda}_j^1 = \lambda_j^o + \delta\lambda_j^o \qquad (3.11)$$

and then the bottom boundary condition becomes:

$$\bar{\phi}(1,\lambda_j^o + \delta\lambda_j^o) = 0$$

Expanding this bottom boundary equation in a Taylor Series to first order, about the initial approximation we have:

$$\bar{\phi}(1,\lambda_j^o + \delta\lambda_j^o) = \bar{\phi}(1,\lambda_j^o) + \psi(1,\lambda_j^o)\delta\lambda_j^o$$

and so, for a small enough value of $\delta\lambda_j^o$, we have, from the bottom boundary condition,

$$\delta\lambda_j^o = -\frac{\bar{\phi}(1,\lambda_j^o)}{\psi(1,\lambda_j^o)} \qquad \text{for } \psi(1,\lambda_j^o)\neq 0 \qquad (3.12)$$

So that the second approximation to the eigenvalue can be obtained from (3.11). Since $\bar{M}$ is independent of $\bar{\lambda}$, then we can differentiate (3.7) and (3.8) with respect to $\bar{\lambda}$ to obtain:

$$\frac{d^2\psi}{dz^2} + \bar{\lambda}\bar{M}\psi + \bar{M}\phi = 0 \qquad (3.13)$$

and $$\frac{d\psi}{d\bar{z}} + \bar{\lambda}\psi + \bar{\phi} = 0 \qquad \text{at } \bar{z} = 0 \qquad (3.14)$$

If we fix the relative vertical displacement of the surface equal to a small constant, η, then the surface boundary condition becomes:

$$\bar{\phi}(o,\lambda) = \eta \neq o$$

and all other values of the vertical displacement will be scaled to this value. So that (3.14) becomes:

$$\frac{d\psi}{d\bar{z}} = -\eta \qquad \text{at } \bar{z} = 0 \qquad (3.15)$$

Hence, we can integrate (3.7) using the initial conditions of (3.8), and using the values of $\phi(\bar{z},\lambda_j^o)$, we can integrate (3.13) using (3.15) as the initial

condition. Using the values of $\bar{\phi}(1,\lambda_j^o)$ and $\psi(1,\lambda_j^o)$, we can then use (3.11) and (3.12) to calculate the next approximation to the eigenvalue which is:

$$\lambda_j^1 = \lambda_j^o - \frac{\bar{\phi}(1,\lambda_j^o)}{\psi(1,\lambda_j^o)}$$

This procedure may be iterated any number of times until the approximation coverges to the required number of digits. Once a sufficiently accurate value is obtained for $\bar{\lambda}$, then equation (3.7) can be integrated using the initial condition (3.8).

It might be thought that an iteration of the solution is of no use since the first approximations are in finite-difference form in any case and so the first approximation should be quite accurate. However, there are two reasons for the iteration. In the first place, the finite-difference form is only accurate if the density distribution is linear between the specified points. This is very seldom the case unless a large number of points are used in the first approximation. It is much more economical of computer time to use only a limited number of points for the first approximation and then to use all of the available points for the integration. Secondly, the integration allows a relaxation of the requirement that the density be specified at equally-spaced depths. If the finite-difference method assumes that the points used are equally-spaced, even though this is not true, a reasonable first approximation may be obtained. The integration routine can then be used to refine this estimate by taking into consideration the actual spacing between the depths.

3.5 Normalization of the Solution

Since we have determined $\bar{\phi}(\bar{z})$ for a number of eigenvalues to within a multiplicative constant, it is natural to want to normalize these values in some systematic way. A very natural way is given by the calculus of variations [Lindsay and Margenau, 1963, pp 442]. For a minimum total energy, equation (3.7) should be substituted into the Euler-Lagrange equations to obtain the normalization equation:

$$\int_0^1 \bar{M}\bar{\phi}^2 \, d\bar{z} = 1$$

Hence, for the j^{th} eigenvalue, we set,

$$\phi_j(\bar{z}) = \gamma_j \, \bar{\phi}_j(\bar{z}) \quad \text{and} \quad \frac{d\phi_j}{d\bar{z}} = \gamma_j \frac{d\bar{\phi}_j(\bar{z})}{d\bar{z}}$$

where: $$\gamma_j = 1/\sqrt{\int_0^1 \bar{M}\bar{\phi}_j^2 d\bar{z}}$$

There are, of course, numerous other ways of normalizing the solution. In method 2, $d\phi/dz$ is set to 1.0 at the surface. Another common normalization in the literature is to set the maximum value of $\phi = 1.0$.

3.6 Computer Program for Method 1

The theory developed in section 3 has been implemented in a single computer program, EIGEN, which is programmed in FORTRAN IV. This program has, as input parameters, the water depth, the number of eigenvalues and eigenfunctions to be calculated, and the period of the forcing function. In a separate input routine the density structure, expressed as sigma-t, are read in along with the sampling depths, with the first value being the surface density and the last value being the bottom density. Provisions are made for plotting the Brunt-Vaisala frequency and density structure on one plot and various user-controlled combinations of the eigenvalues and derivatives of the eigenvalues on other plots. Integration of the differential equations is done by fourth-order Runge-Kutta technique. A complete listing, internally documented, is given as Appendix 1.

It has been found empirically that if the number of points in the vertical is fairly small and does not resolve the density structure adequately, the program will, in some cases, not converge to the proper mode. That is to say, for example, if the initial approximation to an

eigenvalue is not sufficiently close, then the integration routine will converge to a different mode. This sometimes results in the program skipping the computation of certain modes. Provision has been made in the program to make two additional passes to detect missing modes. If modes have been missed, then the program will attempt to recompute them, using as a first approximation, values obtained from neighbouring eigenvalues. If the program still skips the computation of certain modes, the only solution is to supply more points in the vertical density structure. As a rule of thumb, it has been found desirable to have at least three density values for each vertical wavelength, although this may occasionally be relaxed somewhat.

CHAPTER 4

4.0 Method 2

This second method is due to P.F. Hamblin. There are many carryovers from Method 1, since the equations being solved are virtually identical. The development of the vertical structure equation is quite different and no non-dimensionalization is done. The discussion of the eigenvalue problem and the theorems presented for Method 1 are directly applicable. The major differences in this second method lie in the separation of variables and in the computation of the particular eigenvalues. In this separation of variables, no frequency dependence is assumed for the vertical structure. This will lead to difficulties if the forcing frequency is near the Brunt-Vaisala frequency but for long waves is unimportant. The second major difference is that this second method assumes an empirical relationship between the eigenvalues and uses a result from two-layer theory to obtain the first approximation to the eigenvalues. The further approximation to the eigenvalues is calculated in a different way than in Method 1. Numerical experiments have shown the two methods to give virtually identical answers for sufficiently long period waves.

4.1 The Vertical Structure Equation

We assume the linearized, inviscid Boussinesq equations derived in Chapter 2, with the further assumption that we may neglect vertical acceleration so that the equations are:

$$\frac{\partial u}{\partial t} - fv + \frac{1}{\bar{\rho}} \frac{\partial P}{\partial x} = 0 \tag{4.1}$$

$$\frac{\partial v}{\partial t} + fu + \frac{1}{\bar{\rho}} \frac{\partial P}{\partial y} = 0 \tag{4.2}$$

$$\frac{\partial P}{\partial z} + g\rho = 0 \tag{4.3}$$

$$\frac{\partial \rho}{\partial t} + w \frac{\partial \bar{\rho}}{\partial z} = 0 \tag{4.4}$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \qquad (4.5)$$

with the boundary conditions:

$$w = 0 \qquad \text{at } z = H \qquad (4.6)$$

and

$$\frac{\partial P}{\partial t} + w \frac{\partial P}{\partial z} = 0 \qquad \text{at } z = 0 \qquad (4.7)$$

We differentiate (4.1) and (4.2) with respect to z; (4.3) with respect to x and then y, and substitute into (4.3) to obtain:

$$u_{tz} - fv_z + \frac{1}{\rho} g\rho_x = 0$$

$$v_{tz} + fu_z + \frac{1}{\rho} g\rho_y = 0$$

Then, differentiating these two with respect to t and substituting from (4.4), we obtain:

$$u_{ttz} - fv_{tz} - \frac{g}{\rho} \rho_z w_x = 0$$

and

$$v_{ttz} + fu_{tz} - \frac{g}{\rho} \rho_z w_y = 0$$

Solving these two equations simultaneously, we obtain, after defining $N^2 = \frac{g}{\rho} \rho_z$:

$$\left(\frac{\partial}{\partial t^2} + f^2 \right) u_{zt} - N^2(w_{xt} + fw_y) = 0 \qquad (4.8)$$

$$\left(\frac{\partial}{\partial t^2} + f^2 \right) v_{zt} - N^2(w_{yt} - fw_x) = 0 \qquad (4.9)$$

Substituting these equations into equation (4.5), we have:

$$N^2(w_{xx} + w_{yy}) + \left[\frac{\partial^2}{\partial t^2} + f^2 \right] w_{zz} = 0$$

We now look for a solution by setting the vertical velocity

$$w(x,y,z,t) = \phi(z)F(x,y,t) \qquad (4.10)$$

so we have:

$$\frac{F_{xx} + F_{yy}}{\left(\frac{\partial^2}{\partial t^2} + f^2\right) F} = -\frac{\phi_{zz}}{N^2\phi} = \frac{1}{c^2}$$

where c^2 is the constant of separation, which has the dimension of velocity. So, the vertical velocity equation is:

$$\frac{d^2\phi}{dz^2} + \frac{N^2}{c^2}\phi = 0 \tag{4.11}$$

In this separation of variables, the boundary condition at the bottom is simply,

$$\phi(z) = 0 \qquad \text{at } z = H \tag{4.12}$$

from (4.6), but the surface boundary condition cannot be simply written down since the separation of variables makes no assumption about the time dependence of the wave form. From (4.3), which makes the hydrostatic approximation,

$$\frac{\partial P}{\partial z} = -g\bar{\rho}$$

So (4.7) becomes:

$$\frac{\partial P}{\partial t} - wg\bar{\rho} = 0$$

Hence:

$$\frac{\partial^2 P}{\partial t \partial z} - g\bar{\rho}\frac{\partial w}{\partial z} = 0$$

but, differentiating (4.3) with respect to t, gives:

$$\frac{\partial P}{\partial t \partial z} + g\frac{\partial \rho}{\partial t} = 0$$

so,

$$\bar{\rho} \frac{\partial w}{\partial z} + \frac{\partial \rho}{\partial t} = 0$$

and from (4.4):

$$\bar{\rho} \frac{\partial w}{\partial z} - w \frac{\partial \bar{\rho}}{\partial z} = 0$$

Substituting (4.10) and dividing by F, we have:

$$\frac{\partial \phi}{\partial z} - \frac{1}{\bar{\rho}} \frac{\partial \bar{\rho}}{\partial z} \phi = 0$$

so,

$$\frac{\partial \phi}{\partial z} - \frac{N^2}{g} \phi = 0$$

But, from (4.11), we get:

$$\frac{\partial^2 \phi}{\partial z^2} + \frac{g}{c^2} \frac{\partial \phi}{\partial z} = 0$$

and, integrating this expression,

$$\frac{\partial \phi}{\partial z} + \frac{g}{c^2} \phi + K = 0$$

However, since ϕ is proportional to the vertical velocity and there can be no net flow through the surface, the constant of integration, K, must be zero. The surface boundary condition is therefore:

$$\frac{\partial \phi}{\partial z} + \frac{g}{c^2} \phi = 0 \qquad \text{at } z = 0 \qquad (4.13)$$

Equations (4.11), (4.12), and (4.13) define an eigenvalue problem for the vertical velocity structure of an arbitrarily stratified water column. The form of the equations is the same as with the previous method, and so the comments and theorems stated there apply to this

problem also. The function ϕ is proportional to the vertical velocity and also to the vertical displacement of an isopycnal. The function $\frac{d\phi}{dz}$ is proportional to the horizontal velocity.

It is possible to calculate a wave-induced Richardson number, since if $\vec{u}$ is the horizontal velocity, then:

$$R_i = \frac{N^2}{\left(\frac{d\vec{u}}{dz}\right)^2} = \frac{N^2}{\left(\frac{d}{dz}\left(\frac{d\phi}{dz}\right)\right)^2}$$

and from the vertical structure equation:

$$R_i = \frac{N^2}{\left(\frac{N^2}{c^2}\phi\right)^2} = \frac{c^4}{N^2\phi^2}$$

4.2 First Approximation to the Eigenvalues

Method 2 uses a somewhat similar approach to the calculation of eigenvalues and eigenfunctions as Method 1 in that first approximations are obtained, and then these approximations are refined through the integration of the vertical structure equations. The details, however, differ considerably. Rather than using a finite-difference method for obtaining the first approximation, the first approximations are obtained from the restricted theory of shallow-water waves.

The phase speed of the barotropic mode, or zeroth eigenvalue, is given by:

$$c_o^o = \sqrt{gH}$$

The phase speed of the baroclinic mode for a two-layer system, whose layer depths are h_1 and h_2 and whose densities are ρ_1 and ρ_2, is given by:

$$c = \sqrt{g\left(\frac{\rho_2-\rho_1}{\rho_2}\right)\frac{h_1h_2}{h_1+h_2}}$$

So, if we assume a two-layered system where layers are equal to half the water depth and if we assume the surface density, ρ_s, is characteristic

of the upper layer and bottom density, ρ_b, is characteristic of the bottom layer, then the phase speed for the 1st baroclinic mode is given by:

$$c_1^o = \sqrt{g \frac{\rho b - \rho s}{\rho b} \frac{H}{4}}$$

We can then make use of the formula given in Method 1 for the assymptotic behaviour of the eigenvalues to calculate eigenvalues for the further modes as:

$$\begin{aligned} c_2^o &= c_1^o/3 \\ c_3^o &= c_1^o/5 \\ c_4^o &= c_1^o/7 \end{aligned} \qquad (4.14)$$

and, in fact, we have as n becomes large,

$$c_{n+1}^o = \left(\frac{n}{n+1} \right)^2 c_n^o$$

4.3 Further Approximations to the Eigenvalues by the Shooting Method

The further approximations to the eigenvalues are obtained by integrating the vertical structure equations and iterating until convergence is achieved. We first divide the total depth into halves. From the surface, we integrate the vertical velocity equation:

$$\frac{d^2\phi}{dz^2} + \frac{N^2}{(c^o)^2} \phi = 0$$

using initial values at z = 0 of $\phi = -g/c^2$, and $\frac{d\phi}{dz} = 1$, which were obtained from the surface boundary condition to the midpoint at H/2.

We then integrate the vertical veolcity equation from the bottom to the midpoint using initial values of $\phi = 0$ and $\frac{d\phi}{dz} = \frac{N(H)}{c^o}$. This initial value for the relative horizontal velocity is obtained from the fact that near the bottom the Brunt-Vaisala frequency, N, is usually very nearly constant, and so the solution should be approximately:

$$\phi(z) = A \sin \left(\frac{N(H)}{c^o} z \right)$$

So, if A = 1, then $\frac{d\phi}{dz} = \frac{N(H)}{c^o} \cos \left(\frac{N(H)}{c^o} z \right)$, but, if $\phi = 0$, then at the bottom, $\cos \left(\frac{N(H)}{c^o} z \right) = 1$; hence,

$$\left. \frac{d\phi}{dz} \right|_H = \frac{N(H)}{c^o}$$

If, in particular circumstances, N(H) = 0, then the first non-zero value above the bottom is used.

If we have calculated $\phi(c^o)$ and $\frac{d\phi(c^o)}{dz}$ at the midpoint, then clearly,

$$w(c^o) = \frac{d\phi(c^o)}{dz} \phi(c^o) \Big|_{\text{surface}} - \frac{d\phi(c^o)}{dz} \phi(c^o) \Big|_{\text{bottom}} = 0$$

where the term on the left is the result of integrating from the surface to the midpoint and the term on the right is the result of integrating from the bottom to the midpoint. By perturbing the value of c^o somewhat, we get a second approximation to the eigenvalue as $c^1 = c^o + \delta$. The third approximation can be calculated from the well-known Secant formula, which is:

$$c^2 = \frac{c^o \, w(c^1) - c^1 \, w(c^1)}{w(c^1) - w(c^o)}$$

Further approximations may be generated, and the convergence will be slightly better than linear. Once a good approximation for the eigenvalues is calculated, the vertical structure equation may be integrated from the surface to the bottom.

A normalization for this method is provided by the initial conditions on the integration since we have set $\frac{d\phi}{dz} = 1$ at the surface.

4.4 Computer Program for Method 2

The theory developed in section 4 is embodied in three computer programs: PWAVE, QWAVE, and ESURF, which are written in FORTRAN IV.

Input to PWAVE consists of depth-temperature pairs from which $N^2(z)$ is calculated by a subroutine written by F.M. Boyce. Since the program has been applied only to fresh water, the gradient $N^2(z)$ is specialized to temperature only, and the depth contribution to the density field is removed in this calculation. PWAVE computes refined approximations to the eigenvalues up to the fifth mode, including the surface mode, starting with the crude approximations given by (4.14).

The second program QWAVE takes as input the approximations for the eigenvalues generated by PWAVE and computes further approximations by means of the Secant formula. The integration method used is a Predictor-Corrector method, which employs a variable step size depending on the error level specified. (i.e. IBM/360 Scientific Subroutine Package, 1966)

The third program ESURF uses the refined approximations to the eigenvalues and calculates the function ϕ and $d\phi/dz$ for each mode along with the wave-induced Richardson Number for a normalized unit surface velocity. The values N^2, ϕ, $d\phi/dz$, R_i are plotted on the line printer.

Listings of the programs and documentation for them are given in Appendix 2.

REFERENCES

Batchelor, G.K. An Introduction to Fluid Dynamics. Cambridge University Press, 1967.

Bell, T.H. Numerical Calculation of Dispersion Relations for Internal Gravity Waves. Naval Research Laboratory, Washington, D.C., NRL Report 7294, 46 pp., 1972.

Courant, R. and D. Hilbert. Methods of Mathematical Physics. Vol. 1, Interscience, 1937.

Fjeldstad, J.E. Interne Wellen. Geofysiske Publikasjoner, Vol. X, No. 6, 1933.

Grad, J. and M.A. Brebner. Eigenvalues and Eigenvectors of a Real General Matrix. Communications of the A.C.M., Algorithm 343, Vol. II, No. 12, December, 1968, pp. 820-826.

Householder, A. The Theory of Matrices in Numerical Analysis. Blaisdell, New York, 1964.

Keller, H.B. Numerical Methods for Two-Point Boundary Value Problems. Blaisdell Pub. Co., Waltham, Mass. U.S.A. 1968.

Lindsay, R.B. and H. Margenau. Foundations of Physics. Dover, 1963.

Long, R.R. On the Boussinesq Approximation and its Role in the Theory of Internal Waves. Tellus, XVII, 1965, pp. 46-52.

Phillips, O.M. The Dynamics of the Upper Ocean. Cambridge University Press, 1969.

Roberts, J. Internal Gravity Waves in the Ocean. Marcel Dekker, New York, 1975.

A P P E N D I X 1

- Program EIGEN -

Programmed

by

L.R. Muir

Ocean and Aquatic Sciences
Canada Centre for Inland Waters
P.O. Box 5050
Burlington, Ontario, Canada

```
      PROGRAM EIGEN
C
C     A PROGRAM TO COMPUTE A PRESCRIBED NUMBER OF EIGENVALUES AND THE
C     RELATED EIGENFUNCTIONS AND DERIVATIVES OF THE EIGENFUNCTIONS FOR
C     THE NORMALIZED VERTICAL STRUCTURE EQUATION FOR INTERNAL WAVES.
C     ******************************************************************
C
C
C     PROGRAMMED BY:    L.R. MUIR
C                       OCEAN AND AQUATIC SCIENCES
C                       CANADA CENTRE FOR INLAND WATERS
C                       P.O. BOX 5050
C                       BURLINGTON, ONTARIO, CANADA
C     ******************************************************************
```

```
C
C     SUBROUTINES REQUIRED...
C     1) INPUT - TO READ IN SIGMA-T VALUES AND DEPTHS. MUST BE
C              USER SUPPLIED.
C     2. QREIG - TO CALCULATE FIRST APPROXIMATION TO THE EIGENVALUES.
C     3. SORTA - TO SORT THE EIGENVALUES INTO ASCENDING ORDER
C     4. QR    - CALLED BY QREIG
C     5. INTEG - TO COMPUTE HIGHER APPROXIMATIONS TO THE EIGENVALUES
C              USING THE APPROXIMATIONS GIVEN BY QREIG AND TO COMPUTE
C              THE EIGENFUNCTIONS AND THEIR DERIVATIVES USING A FOURTH-
C              ORDER RUNGE-KUTTA TECHNIQUE.
C     6. NMLZ - TO NORMALIZE THE EIGENFUNCTION AND ITS DERIVATINE SO THA
C        THE TOTAL ENERGY IS 1.0.
C     7. PDEN - TO PLOT THE DENSITY AND THE BRUNT-VAISALA FREQUENCY AS
C              A FUNCTION OF DEPTH.
C     8. PEIF - TO PLOT VARIOUS COMBINATIONS OF THE EIGENFUNCTIONS OR
C        THEIR DERIVATIVES AS A FUNCTION OF DEPTH. THEY ARE NORMALIZED
C        THAT THE MAXIMUN RELATIVE VALUE IS 1.0.
C     9. CORNS - TO CORRECT BRUNT-VAISALA FREQUENCY FOR SOUND SPEED
C              EFFECTS. THIS SUBROUTINE IS NOT NORMALLY USED EXCEPT
C              WHEN THE DENSITY GRADIENT IS VERY SMALL. IF THE
C              ROUTINE IS TO BE USED,THE TEMPERATURE AND SALINITY
C              STRUCTURE MUST BE SUPPLIED AND THE C REMOVED FROM
C              COLUMN 1 OF THE CALL STATEMENT.
C     10. SNDVEL - A FUNCTION SUBROUTINE CALLED BY CORNS TO CALCULATE
C              SOUND VELOCITY FROM TEMP. AND SLINITY.
C     11. CHPR - TO COMPUTE MODE NUMBER ACTUALLY COMPUTED(CF. MODE
C        NUMBER THAT WAS SUPPOSED TO BE COMPUTED), AND TO PRINT
C        OUT EIGENFUNCTIONS AND ITS DERIVATIVE.
C
C      MAXIMUM DIMENSIONS :
C      1. 51 POINTS IN THE VERTICAL.
C     2. 11 EIGENVALUES AND EIGENFUNCTIONS.
C     3. 26 POINTS USED FOR THE FIRST APPPROXIMATIONS TO THE EIGENVALUES.
C     NAPP = NUMBER OF POINTS USED IN THE FIRST APPROXIMATION TO THE
C        EIGENVALUES. PROGRAM WILL CALCULATE NAPP IF AN IMPROPER VALUE
C        SPECIFIED.
C
C     ********************************************************************
C
C     INPUT DESCRIPTION...
C     H =  DEPTH OF WATER COLUMN
C     NPTS = NO. OF POINTS AT WHICH THE DENSITIES ARE INPUT.THE FIRST
C          MUST BE AT THE SURFACE AND THE LAST MUST BE AT THE BOTTOM. TH
C      NEED NOT BE AT EQUALLY SPACED INTERVALS...ALTHOUGH THE FIRST
C        APPROXIMATIONS ARE CALCULATED AS IF THE POINTS WERE GIVEN AT E
C        SPACINGS.
C     Z(I) = ARRAY OF DEPTH VALUES AT WHICH THE SIGMA-T ARE GIVEN. ALL
C        OUTPUT IS REFERED TO THESE DEPTHS WHICH ARE NORMALIZED IN THE
C        PROGRAM SO THAT THE SURFACE IS AT 0.0 AND THE BOTTOM IS AT 1.0.
C     LAT = LATITUDE OF THE OBSERVATIONS,FOR CALCULATION OF CORIOLIS.
C     NEIG = NO. OF EIGENVALUES (MODES) TO BE CALCULATED.
C        DOES NOT INCLUDE THE SURFACE MODE (MODE 0)
C     PER  = PERIOD OF WAVE BEING CALCULATED - IN HOURS/CYCLE.
C     ITIT = ANY 80 CHARACTER TITLE
C     ST(I)= SIGMA-T AT DEPTH I,  I=1 MUST BE AT THE SURFACE  AND I=NPTS
C              MUST BE AT THE BOTTOM (DEPTH=H).
C
C     ********************************************************************
C
C     DATA INPUT:
C     1. ITIT(I)(I=1,20) FMT(20A4)
C     2. H,NPTS,NEIG,PER,NAPP,LAT    FMT(F5.0,I5,I5,F10.8,I5,F5.0)
C     3. INPUT(NPTS,ST,Z) - USER SUPPLIED ROUTINE.
C     4. CORNS - IF NEEDED - MUST SUPPLY T,S PAIRS.
C     5. PLOT CONTROL- IF DESIRED MAY BE ANY COMBINATION.
C     6. KEY    FMT(A4)  - IF KEY=9999  NEW JOB STARTS AT CARD 1.
C                                 8888    END OF JOB.
C
```

```
C     ******************************************************************
CC
      DIMENSION ST(52),DST(52),AM(26,26)
      DIMENSION PHI(51,11),DPHI(51,11),ICARD(19)
      DIMENSION  Z(51),F(51),FP(51)
      DIMENSION  ITIT(20)
      DIMENSION BF(51)
      DIMENSION EV(11),EC(11)
      DIMENSION ITITLE(20)
      EQUIVALENCE(ITIT,ITITLE)
      REAL NS(52),MUR(26),MUI(26)
      REAL H
      REAL LAT
      REAL LAMBDA
      DATA PI,G/3.14159,9.81/
C     ******************************************************************
CCC
CCC   INITIALIZE AND READ DATA
C
      CALL PLOT(0.0,1.0,-3)
    1 CONTINUE
      CALL ZEROV(PHI,561)
      CALL ZEROV(DPHI,561)
      CALL ZEROV(EV,11)
      CALL ZEROV(EC,11)
C
      READ(60,100)(ITIT(I),I=1,20)
  100 FORMAT(20A4)
      PRINT 101, (ITIT(I),I=1,20)
  101 FORMAT(1H1,20A4)
C
      READ(60,110) H,NPTS,NEIG,PER,NAPP,LAT
  110 FORMAT(F5.0,I5,I5,F10.8,I5,F5.0)
      PRINT 111,H,PER,LAT
  111 FORMAT(1H0,*WATER DEPTH = *,F5.0,/*  WAVE PERIOD = *,F10.5,/
     1  ,*  LATITUDE OF OBSERVATION = *,F5.0)
      SG=((PI/(PER*1800.))**2)/G
      NEIG = NEIG + 1
      NMIN1=NPTS-1
C
      CALL INPUT(NPTS,ST,Z,H)
C
CCC   COMPUTE DENSITY GRADIENT
C
      DO 10 I=1,NMIN1
      J=I+1
      DZ = (Z(J) -Z(I))*H
      DST(I) = (ST(J)-ST(I))/DZ
      NS(I) = H*(DST(I)/(1000.+ST(I)) - SG)
      IF(NS(I).GE.0.) GO TO 10
      PRINT 150,I
  150 FORMAT(1H0,*DENSITY STRUCTURE IS UNSTABLE AT DEPTH NO. *,I5)
      STOP
   10 CONTINUE
      NS(NPTS) = H*(DST(NMIN1)/(1000.+ST(NPTS))-SG)
C
CCCCC
CCCCC CORRECT N**2 TO TAKE SOUND VELOCITY INTO ACCOUNT.
CCCCC IF THE DENSITY STRUCTURE IS VERY SLIGHT THIS MAY BECOME IMPORTANT.
CCCCC TO INCLUDE THE EFFECT, REMOVE THE C FROM THE CALL TO CORNS.
CCCCC TEMPERATURE AND SALINITY ARE READ IN CORNS.AND MUST BE IN THE DATA
CCCCC
CCCCC CALL CORNS(NS,NPTS,H)
CCCCC
CCCCC PRINT OUT INITIAL DATA FOR CHECKING...
C
      PRINT 200
```

```
  200 FORMAT(1H0,* I   DEPTH   SIGMA-T   D(ST)/DZ    BV FREQ     BV PER      N
     1S*,/,1H0,30X,*(SEC/CYC)   (CYC/SEC)*,/)
      DO 20 I=1,NPTS
      BV = SQRT((G/H)*NS(I) + SG)
      BF(I) = BV/(2.0*PI)
      BVP = 1./BF(I)
      DEP = Z(I)*H
      PRINT 201,I,DEP,ST(I),DST(I),BF(I),BVP,NS(I)
  201 FORMAT(1H ,I3,1X,F5.1,3X,F5.2,4X,F6.4,2X,F8.7,2X,F9.3,3X,F9.6)
   20 CONTINUE
C
CCCCC
CCCCC
C
      DECIMATE FOR FIRST APPROXIMATION OF EIGENVALUES,AND FILL AM MATRIX
      USE NAPP POINTS FOR THIS APPROXIMATION. NAPP MUST BE GREATER THAN
      NEIG AND MUST BE A PRIME FACTOR OF NPTS+1.
      FOR THIS APPROXIMATION, THE DEPTHS ARE ASSUMED EQUALLY SPACED.

      IF(NAPP.LT.NEIG) NAPP = NPTS
      NPP1 = NPTS + 1
      IREM = MOD(NPP1,NAPP)
      IF(IREM.NE.0) NAPP = NPTS
      FN = FLOAT(NAPP-1)
      AM(1,1)=1.0/(NS(1)+FN)
      AM(1,2)=-AM(1,1)
      AM(NAPP,NAPP-1)=-1./NS(NPTS)
      AM(NAPP,NAPP)   = -AM(NAPP,NAPP-1)*2.
      IDEC =  NMIN1/(NAPP-1)
      J=1
      II=1+IDEC
      IL =  IDEC*(NAPP-2) + 1
      DO 30 I=II,IL,IDEC
      J= J+1
      AI= 1./NS(I)
      AM(J,J-1)= -AI
      AM(J,J)=  2.*AI
      AM(J,J+1)= -AI
   30 CONTINUE
C
CCCCC
C
      DIAGONALIZE AM MATRIX AND SORT EIGEVALUES INTO ASCENDING ORDER.

      CALL QREIG(AM,26,NAPP,MUR,MUI)
      CALL SORTA(MUR,NAPP)
      FNZ= FN*FN
      DO 40 I=1,NEIG
      EV(I) = FNZ*MUR(I)
   40 CONTINUE
C
CCC
C
      PRINT FIRST APPROXIMATION TO EIGENVALUES.

      PRINT 101,(ITIT(I),I=1,20)
      PRINT 250,NPTS,NAPP,FNZ
  250 FORMAT(1H0,*FROM THE *,I5,* POINTS READ IN,*,I5,* WERE USED TO
     1 COMPUTE THE EIGENVALUES*,/,1H ,* MULTIPLICATIVE FACTOR IS *,F8.3)
      PRINT 251
  251 FORMAT(1H0,*I*,9X,*MUR(I)*,9X,* EV(I)*,/)
      PRINT 202,(I,MUR(I),EV(I),I=1,NEIG)
  202 FORMAT(1H ,I2,4X,E10.5,5X,E10.5)
C
CCCCCC
      GET FURTHER APPROXIMATIONS TO THE EIGENVALUES BY INTEGRATING
       FROM THE TOP TO THE BOTTOM AND ADJUSTING UNTIL CONVERGENCE.

      COMPUTE - ONE PASS THROUGH FOR EACH MODE TO BE COMPUTED.
C

      DO 60 K=1,NEIG
      MODE = K-1
      C = 0.01/FLOAT(K)
      LAMBDA = EV(K)
```

```
      CALL INTEG(NPTS,NS,LAMBDA,C,F,FP,IT,Z)
      CALL NMLZ(NS,F,FP,NPTS,QQ,Z)
      CALL CHPR(MODE,F,FP,EC,LAMBDA,IT,QQ,PHI,DPHI,ITIT,Z,NMIN1)
   60 CONTINUE
C
CCCCCCC
      RECOMPUTE FOR MODES MISSED. VARIOUS METHODS ARE USED FOR GETTING
      APPROXIMATIONS TO THE EIGENVALUES.THIS SECTION IS PASSED THROUGH T
      ONLY - IF THERE ARE STILL MISSING MODES,YOU ARE OUT OF LUCK AND
      NEED MORE DATA POINTS IN THE VERTICAL.
C
      DO 64 II=1,2
      DO 64 I=1,NEIG
      IF(EC(I).GT.0.5) GO TO 64
      IF(I.GT.1) GO TO 57
      EC(1) = 1.0
      GO TO 63
   57 IF(EC(I-1).LT.0.5.OR.EC(I+1).LT.0.5) GO TO 58
      EC(I) = (EC(I-1) + EC(I+1))/2.0
      GO TO 63
   58 IF(I.GT.2) GO TO 59
      EC(2) = 8.0*(1000.-ST(NPTS))/(ST(NPTS)-ST(1))
      GO TO 63
   59 IF(EC(I-1).LT.0.5) GO TO 61
      EC(I) = FLOAT(I*I)*EC(I-1)/FLOAT((I-1)*(I-1))
      GO TO 63
   61 IF(EC(I+1).LT.0.5) GO TO 62
      EC(I) = FLOAT(I*I)*EC(I+1)/FLOAT((I+1)*(I+1))
      GO TO 63
   62 EC(I) = 0.0
      GO TO 64
   63 MODE = I-1
      C = 0.01/FLOAT(I)
      LAMBDA = EC(I)
      CALL INTEG(NPTS,NS,LAMBDA,C,F,FP,IT,Z)
      CALL NMLZ(NS,F,FP,NPTS,QQ,Z)
      CALL CHPR(MODE,F,FP,EC,LAMBDA,IT,QQ,PHI,DPHI,ITIT,Z,NMIN1)
   64 CONTINUE
C
C     PRINT OUT TABLES OF NORMALIZED VALUES
      PRINT 500,(ITITLE(I),I=1,20)
  500 FORMAT(1H1,20A4,/,1H0,* RELATIVE VERTICAL DISPLACEMENT*/)
      PRINT 550,G,H,PER
  550 FORMAT(1H0,*GRAVITATIONAL CONSTANT = *,F5.2,5X,*DEPTH = *,F5.0,
     1   5X,* WAVE PERIOD = *,F10.5,/)
      PRINT 502,(I,I=1,NEIG)
  502 FORMAT(1H0,5X,*DEPTH*,5X,1H0,5X,10(4X,I2,5X),/)
      DO 65 I=1,NPTS
      PRINT 501,I,Z(I),(PHI(I,J),J=1,NEIG)
  501 FORMAT(1H ,I2,3X,F5.3,11(3X,F8.3))
   65 CONTINUE
      PRINT 503,(ITITLE(I),I=1,20)
  503 FORMAT(1H1,20A4,/,1H0, * RELATIVE HORIZONTAL VELOCITY *)
      PRINT 550,G,H,PER
      PRINT 502,(I,I=1,NEIG)
      DO 66 I=1,NPTS
      PRINT 501,I,Z(I),(DPHI(I,J),J=1,NEIG)
   66 CONTINUE
C
CCCCCC
      PRINT TABLES WITH WAVELENGTHS AND PHASE SPEEDS.
      WAVELENGTH IS CORRECTED FOR CORIOLIS IF LAT HAS BEEN ENTERED.
      NOTE THE ASSUMED DISPERSION RELATIONSHIP.........
C
      PRINT 101,(ITIT(I),I=1,20)
      PRINT 550,G,H,PER
      PRINT 555
  555 FORMAT(1H0,* MODE    APPROX*,8X,*CALC*,6X,*PHAXE*,10X,*WAVE*,8X,
```

```
     1*WAVE*,/,2H ,* NO  EIGENVALUE*,4X,*EIGENVALUE   SPEED*,9X,*LENGTH*
     2,5X,*NUMBER*,/,1H ,31X,*(M/SEC)*,7X,*(METRES)  (RAD/METRE)*,/)
      DO 70 I=1,NEIG
      IF(EC(I).LT.0.5) GO TO 70
      MDE = I-1
      CEL = SQRT(G*H/EC(I))
      FCOR = (1.46E-4)*SIN(LAT/57.2958)/(2.0*PI)
      FMOD = SQRT(((1.0/(3600.*PER))**2) - (FCOR*FCOR))
      WVL = CEL/FMOD
      WVN = 2.0*PI/WVL
      PRINT 505,MDE,EV(I),EC(I),CEL,WVL,WVN
  505 FORMAT(1H0,I3,3X,F10.2,5X,F10.2,1X,F10.6,4X,F10.2,2X,F10.8)
   70 CONTINUE
C
CCC   PUNCH OUT REL VERT DISP  FOR PROGRM IGWAMP
C
      DO 67 II=2,NEIG
      PUNCH 650,(PHI(KK,II),KK=2,NMIN1)
  650 FORMAT(10F8.4)
   67 CONTINUE

C
CCC   PLOT CONTROL
C
  105 READ(60,700) KEY,ICARD
  700 FORMAT(A4,19A4)
C
      IF(KEY.EQ.4H9999) GO TO 01
      IF(KEY.NE.4HPDEN) GO TO 115
      CALL PDEN(NPTS,ST,BF,Z,ICARD,ITITLE,H)
      GO TO 105
  115 IF(KEY.NE.4HPEIF) GO TO 120
      CALL PEIF(NPTS,Z,H,EV,PHI,DPHI,ICARD,ITITLE)
      GO TO 105
C
  120 IF (KEY.EQ.4H8888) GO TO 90
      PRINT 750,KEY
  750 FORMAT(1H0,* UNRECOGNIZABLE KEY WORD IN PLOT CONTROL IS *,A4,* --
     1 PROGRAM TERMINATED---*)
      STOP

C     TERMINATE PROGRAM
C
   90 PRINT 600
  600 FORMAT(1H0,*ALL COMPUTATIONS COMPLETED*)
      CALL PLOT(0.0,0.0,999)
      STOP
      END
```

```
      SUBROUTINE CHPR(MODE,F,FP,EC,LAMBDA,IT,QQ,PHI,DPHI,ITIT,Z,NMIN1)
      DIMENSION F(51),FP(51),EC(11),PHI(51,11),DPHI(51,11),ITIT(20)
      DIMENSION Z(51)
      REAL LAMBDA
C
CCC    TO CHECK ON MODE NUMBER ACTUALLY COMPUTED,TO PRINT OUT VALUES
CCC     AND RESORT ARRAYS.
C
      MCOMP = 0
      DO 55 I=2,NMIN1
      IF(F(I).LT.F(I-1).AND.F(I).LT.F(I+1)) MCOMP = MCOMP+1
      IF(F(I).GT.F(I-1).AND.F(I).GT.F(I+1)) MCOMP = MCOMP + 1
   55 CONTINUE
      IF(MODE.EQ.MCOMP) GO TO 56
      PRINT 270,MODE,MCOMP
  270 FORMAT(1H0,* INSTEAD OF MODE *,I5,*  THE *,I5,*TH MODE WAS CALC.*)
      IF(MCOMP.GT.11) MCOMP=11
   56 MODE = MCOMP
C
CCC    PRINT EIGENFUNCTIONS AND DERIVATIVES
CCCC      ONE PAGE FOR EACH EIGENVALUE.
CCC    PLACE IN LARGE PLOTTING ARRAY.
C
      PRINT 101, (ITIT(I), I=1,20)
  101 FORMAT(1H1,20A4)
      PRINT 300,MODE,LAMBDA,IT,QQ
  300 FORMAT(/,1H0,5X,* MODE NUMBER*,I5,/,1H ,*RECOMPUTED EIGENVALUE = *
     1,E20.8,*  OBTAINED AFTER *,I5,* ITERATIONS*,/,1H ,* NORMALIZATION
     2FACTOR =*,E14.2,//,* NOR.DEP.*,10X,*F*,19X,*FP*,)
      MODE = MODE+1
      NPTS = NMIN1 + 1
      EC(MODE) = LAMBDA
      DO 60 I=1,NPTS
      PHI(I,MODE) = F(I)
      DPHI(I,MODE) = FP(I)

      PRINT 400, Z(I),F(I),FP(I)
  400 FORMAT(1H ,F5.4,2(5X,E15.3))
   60 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE QREIG(A,L,M,ROOTR,ROOTI)

C     PROGRAM TO CALL QR TRANSFORMATION. MAXIMUM ITER IS 50.
CCC
CCC   THIS SUBROUTINE USES HOUSEHOLDERS QR TRANSFORMATION METHOD
CCC   TO DIAGONALIZE THE MATRIX A AND HENCE CALCULATE THE REAL
CCC   AND IMAGINARY PARTS OF THE EIGENVALUES. IT MAY BE REPLACED
CCC   BY THE ROUTINES PUBLISHED BY GRAD AND BREBNER,ALGORITHM
CCC   NO. 343,CMM. OF ACM IF MORE DOCUMENTATION IS REQUIRED.
C
      DIMENSION A(26,26),ROOTR(L),ROOTI(L)
      N = M
      JJ=1
177   XNN=0.0
      XN2=0.0
      AA=0.0
      B=0.0
      C=0.0
      DD=0.0
      R=0.0
      SIG=0.0
      ITER = 0
 17   IF(N-2) 83,14,12
83    DO 200 IJK=1,L
      ROOTI(IJK)=0.0
      ROOTR(IJK)=A(IJK,IJK)
200   CONTINUE
  1   RETURN
 14   JJ=-1
   12 X = (A(N-1,N-1) - A(N,N))**2
      S = 4.0*A(N,N-1)*A(N-1,N)
      ITER = ITER + 1
      IF(X .EQ. 0.0 .OR. ABS(S/X) .GT. 1.0E-8) GO TO 15
   16 IF(ABS(A(N-1,N-1))-ABS(A(N,N)))  32,32,31
   31 E = A(N-1,N-1)
      G = A(N,N)
      GO TO 33
   32 G = A(N-1,N-1)
      E = A(N,N)
   33 F = 0.
      GO TO 24
   15 S = X + S
      X = A(N-1,N-1) + A(N,N)
      IF(S)     18,19,19
 19   SQ=SQRT(S)
      F=0.0
      IF (X) 21,21,22
 21   E=(X-SQ)/2.0
      G=(X+SQ)/2.0
      GO TO 24
 22   G=(X-SQ)/2.0
      E=(X+SQ)/2.0
      GO TO 24
   18 F = SQRT(-S)/2.0
      E=X/2.0
      G=X/2.0
 24   H=-F
      IF(JJ) 85,70,70
   70 D = 1.0E-10*(ABS(G) + F)
      IF(ABS(A(N-1,N-2)) .GT. D)  GO TO 26
 85   ROOTR(N) = E
      ROOTI(N) = F
      ROOTR(N-1) = G
      ROOTI(N-1) = H
      N=N-2
      IF(JJ)  1,177,177
   26 IF(ABS(A(N,N-1)) .GT. 1.0E-10*ABS(A(N,N)))  GO TO 50
 87   ROOTR(N) = A(N,N)
      ROOTI(N) = 0.0
      N=N-1
```

```
      GO TO 177
50    IF(ABS(ABS(XNN/A(N,N-1))-1.0)-1.0E-6)   63,63,62
62    IF(ABS(ABS(XN2/A(N-1,N-2))-1.0)-1.0E-6)   63,63,700
63    VQ=ABS(A(N,N-1))-ABS(A(N-1,N-2))
      IF  (ITER-15) 53,164,64
164   IF(VQ) 165,165,166
165   R  =  A(N-1,N-2)**2
      SIG  =  2.0*A(N-1,N-2)
      GO  TO 60
166   R  =  A(N,N-1)**2
      SIG  =  2.0*A(N,N-1)
      GO  TO 60
64    IF(VQ)  87,87,85
700   IF(ITER .GT.  50) GO TO 63
      IF(ITER .GT.  5  ) GO TO 53
701   Z1=  ((E-AA)**2+(F-B)**2)/(E*E+F*F)
      Z2=  ((G-C)**2+(H-DD)**2)/(G*G+H*H)
      IF(Z1-0.25)   51,51,52
51    IF(Z2-0.25)    53,53,54
53    R=E*G-F*H
      SIG=E+G
      GO  TO 60
54    R=E*E
      SIG=E+E
      GO  TO 60
52    IF(Z2-0.25) 55,55,601
55    R=G*G
      SIG=2.*G
      GO  TO 60
601   R=0.0
      SIG=0.0
60    XNN=A(N,N-1)
      XN2=A(N-1,N-2)
      CALL  QRT(A,L,M,N,R,SIG,D)
      AA=E
      B=F
      C=G
      DD=H
      GO  TO 12
       END
```

```
      SUBROUTINE QRT(A,L,M,N,R,SIG,D)
      DIMENSION A(26,26)
C
CC    CALL BY QREIG
C
      N1 = N - 1
      IA=N-2
      IP=N-2
      IF(N-3) 101,10,60
 60   DO 12 J = 3,N1
      J1 = N - J
      IF(ABS(A(J1+1,J1))-D)     10,10,11
   11 DEN = A(J1+1,J1+1)*(A(J1+1,J1+1)-SIG)+A(J1+1,J1+2)*A(J1+2,J1+1)+R
      IF(DEN) 61,12,61
 61   IF(ABS(A(J1+1,J1)*A(J1+2,J1+1)*(ABS(A(J1+1,J1+1)+A(J1+2,J1+2)
     1-SIG)+ABS(A(J1+3,J1+2)))/DEN)-D)  10,10,12
 12   IP=J1
 10   DO  14  J=1,IP
      J1=IP-J+1
      IF(ABS(A(J1+1,J1))-D)     13,13,14
 14   IQ=J1
 13   DO  100  I=IP,N1
      IF(I-IP)     15,15,16
 15   G1=A(IP,IP)*(A(IP,IP)-SIG)+A(IP,IP+1)*A(IP+1,IP)+R
      G2=A(IP+1,IP)*(A(IP,IP)+A(IP+1,IP+1)-SIG)
      G3=A(IP+1,IP)*A(IP+2,IP+1)
      A(IP+2,IP)=0.0
      GO TO 19
 16   G1=A(I,I-1)
      G2=A(I+1,I-1)
      IF(I-IA)     17,17,18
 17   G3 = A(I+2,I-1)
      GO TO 19
 18   G3 = 0.0
 19   XK = SIGN(SQRT(G1*G1 + G2*G2 + G3*G3), G1)
 22   IF(XK)    23,24,23
 23   AL = G1/XK+1.0
      PSI1 = G2/(G1+XK)
      PSI2 = G3/(G1+XK)
      GO TO 25
 24   AL=2.0
      PSI1=0.0
      PSI2=0.0
 25   IF(I-IQ)     26,27,26
 26   IF(I-IP)     29,28,29
 28   A(I,I-1)=-A(I,I-1)
      GO TO 27
 29   A(I,I-1)=-XK
 27   DO 30  J=I,N
      IF(I-IA)   31,31,32
 31   C = PSI2*A(I+2,J)
      GO TO 33
 32   C=0.0
 33   E=AL*(A(I,J)+PSI1*A(I+1,J)+C)
      A(I,J)=A(I,J)-E
      A(I+1,J)=A(I+1,J)-PSI1*E
      IF(I-IA)     34,34,30
 34   A(I+2,J)=A(I+2,J)-PSI2*E
 30   CONTINUE
      IF(I-IA)     35,35,36
 35   JJ=I+2
      GO TO  37
 36   JJ=N
 37   DO  40  J=IQ,JJ
      IF(I-IA)   38,38,39
 38   C = PSI2*A(J,I+2)
      GO TO  41
 39   C=0.0
```

```
41    E=AL*(A(J,I)+PSI1*A(J,I+1)+C)
      A(J,I)=A(J,I)-E
      A(J,I+1)=A(J,I+1)-PSI1*E
      IF(I-IA)  42,42,40
42    A(J,I+2)=A(J,I+2)-PSI2*E
40    CONTINUE
      IF(I-N+3)    43,43,100
43    E = AL*PSI2*A(I+3,I+2)
      A(I+3,I)=-E
      A(I+3,I+1)=-PSI1*E
      A(I+3,I+2)=A(I+3,I+2)-PSI2*E
100   CONTINUE
101   RETURN
       END
```

```
      SUBROUTINE SORTA(A,N)
C     SUBPROGRAM FOR SORTING AN ARRAY OF REAL NUMBERS INTO
C     ASCENDING ORDER.
C
C     UWMS-1171
C
C     UNIVERSITY OF WASHINGTON
C     DEPARTMENT OF OCEANOGRAPHY
C     SEATTLE, WASHINGTON, 98195
C
C     PROGRAMMED BY ..  PAAVO KOVALA                AUGUST 1972
C
C     A = ARRAY OF THE NUMBERS
C     N = NUMBER OF ELEMENTS IN A
C     THE SORTED ARRAY OCCUPIES THE SAME LOCATIONS AS THE ORIGINAL ONE.
C
      REAL A(N)
      DO 2 I=2,N
      K = I - 1
      DO 1 J=I,N
    1 IF (A(K) .GT. A(J)) K = J
      IF (K .LT. I) GOTO 2
      Q = A(K)
      A(K) = A(I-1)
      A(I-1) = Q
    2 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INTEG(NPTS,NS,LAMBDA,C,X,XP,NI,Z)
C
C     THIS SUBROUTINE COMPUTES A FURTHER APPROXIMATION TO THE EIGENVALUE
C     AND THEN USES IT TO COMPUTE THE EIGENFUNCTION AND ITS DERIVATIVES.
C     THE FIRST INTEGRATION SOLVES THE EQUATION
C         FZZ + LAMBDA*AM*F = 0
C     WITH THE END CONDITIONS   FZ + LAMBDA*C = 0 FOR Z=0
C                               F + C = 0   FOR Z=0
C     THE SECOND INTEGRATION SOLVES THE EQUATION
C         GZZ + LAMBDA*AM*G + AM*F = 0
C     WITH THE END CONDITIONS  GZ=0 AND G=0  AT Z=0
C     THE REVISED EIGENVALUE IS COMPUTED AND THE WHOLE PROCESS IS ITERAT
C     UNTIL CONVERGENCE.
C
C     DESCRIPTION OF VARIABLES....
C
C     NP = NUMBER OF INTEGRATION POINTS
C     AM = ARRAY OF NP VALUES OF THE FUNCTION M
C     LAMBDA = EIGENVALUE - APPROSIMATION IN INPUT, FINAL VALUE IN OUTPU
C     C = CHOSEN INITIAL VALUE OF F AT Z=0
C     F = ARRAY OF NP COMPUTED VALUES OF THE EIGENFUNCTION
C     FP = ARRAY OF NP DERIVATIVES OF THE EIGENFUNCTION
C     NI = NUMBER OF ITERATIONS USED FOR CONVERGENCE
C     INSERT = NO. OF INTERPOLATED VALUES INSERTED BETWEEN EACH OF THE
C     MEASURED VALUES IN ORDER TO REDUCE THE SIZE OF THE COMPUTAIONAL ST
C     FACT = A DIAGNOSTIC FOR THE ACCURACY OF THE INTEGRATION. IF FACT
C              IS GREATER THAN A FEW HUNDREDTHS, THE STEP SIZE IS TOO
C              LARGE.
C
      DIMENSION X(51),XP(51)
      DIMENSION F(2002),FP(2002),AM(2002)
      DIMENSION Z(51)
      REAL NS(51)
      REAL K1,K2,K3,K4,LAMBDA
      DATA MNI,ACCUR/15,1.E-5/
C
C     COMPUTE INITIAL STEP SIZE TO BE ABOUT 0.001. IF THIS IS NOT SMALL
C     ENOUGH,THEN THE STEP SIZE IS REDUCED UNTIL THE REQUIRED ACCURACY
C     IS OBTAINED. A MAXIMUM OF 2000 INTERPOLATED VALUES IS ALLOWED
C     BETWEEN EACH OF THE MEASURED VALUES.
C
      RMIN = Z(2) -Z(1)
      DO 2 I=3,NPTS
      DIF = Z(I) - Z(I-1)
      RMIN = AMIN1(RMIN,DIF)
    2 CONTINUE
      INSERT = IFIX(1000*RMIN) - 1
C
    1 CONTINUE
C
      NP = INSERT + 1
      NPP1 = NP + 1
      STEP = (Z(2) - Z(1))/FLOAT(NP)
      PRINT 1000,INSERT,STEP
 1000 FORMAT(1H0,*INSERT = *,I5,5X,*STEP SIZE FOR FIRST INTERVAL = *
     1 F10.8)
C
      DO 30 NI = 1,MNI
      X(1) = C
      XP(1) = -LAMBDA*C
      G1 = 0.
      GP1 = -C
C
      DO 25 II = 2,NPTS
      STEP = (Z(II) - Z(II-1))/FLOAT(NP)
      DELN = NS(II) - NS(II-1)
      DO 5 KK=1,NPP1
```

```
      AM(KK) = (FLOAT(KK-1)/FLOAT(NP))*DELN + NS(II-1)
    5 CONTINUE
C
C  INTEGRATE THE FIRST DIFFERENTIAL EQUATION
      F(1) = X(II-1)
      FP(1) = XP(II-1)
      DO 10 I=2,NPP1
      R1 = FP(I-1)
      K1 = -LAMBDA*AM(I-1)*F(I-1)
      A = (AM(I-1) + AM(I))/2.
      K2 = -LAMBDA*A*(F(I-1) + 0.5*STEP*K1)
      K3 = -LAMBDA*A*(F(I-1) + 0.5*STEP*K2)
      K4 = -LAMBDA*AM(I)*(F(I-1) + STEP*K3)
      R2 = R1 + 0.16666667*(K1+2.0*K2+2.0*K3+K4)*STEP
      FP(I) = R2
      F(I) = F(I-1) + 0.5*STEP*(R1+R2)
   10 CONTINUE
      X(II) = F(NPP1)
      XP(II) = FP(NPP1)
C
CC      INTEGRATE THE AUXILLIARY DIFFERENTIAL EQUATION
C
      DO 20 I=2,NPP1
      R1 = GP1
      K1 = -LAMBDA*AM(I-1)*G1 - AM(I-1)*F(I-1)
      A1 = 0.5*(AM(I-1) + AM(I))
      A2 =0.5*(F(I-1) + F(I))
      K2 = -LAMBDA*A1*(G1 + 0.5*STEP*K1) -A1*A2
      K3 = -LAMBDA*A1*(G1 + 0.5*STEP*K2) -A1*A2
      K4 = -LAMBDA*AM(I)*(G1 + STEP*K3) -AM(I)*F(I)
      R2 = R1 + 0.16666667*STEP*(K1+2.0*K2+2.0*K3+K4)
      GP1 = R2
      G1 = G1 + 0.5*STEP*(R1+R2)
   20 CONTINUE
      FACT1= (K3-K2)/(K2-K1)
      IF(ABS(FACT1).LT.0.05) GO TO 21
      IF(INSERT.EQ.2000) GO TO 50
      INSERT = 1 + 2*INSERT
      IF (INSERT.GT.2000) INSERT=2000
      GO TO 1
   21 CONTINUE
   25 CONTINUE
C
CC      RECOMPUTE EIGENVALUE AND TEST FOR CONVERGENCE
C
      Q = X(NPTS)/G1
      QQ = Q/LAMBDA
      IF(ABS(QQ).LE.ACCUR) GO TO 40
      LAMBDA = LAMBDA- Q
C
      IF(NI.GE.MNI) PRINT 100,NI,LAMBDA,Q
  100 FORMAT(1H0,*CONVERGENCE NOT ATTAINED AFTER*,I3,*  ITERATIONS*,/,
     1   *   NEW EIGENVALUE = *,E12.5,*  WAS CORRECTED BY *,E12.5)
   30 CONTINUE
   40 CONTINUE
      PRINT 1003,NI,LAMBDA,Q
 1003 FORMAT(1H0,* CONVERGENCE ATTAINED AFTER*,I3,* ITERATIONS*,/,
     1  * FINAL EIGENVALUE = *,E12.5,* WAS CORRECTED BY*, E12.5)
      GO TO 60
   50 PRINT 1004
 1004 FORMAT(1H0,*STEP SIZE TOO SMALL -ATTEMPT ABANDONED*)
      LAMBDA = 0.0
   60 CONTINUE
C
      RETURN
      END
```

```
      SUBROUTINE NMLZ(AM,F,FP,NP,Q,Z)
C     SUBPROGRAM FOR NORMALIZING AN EIGENFUNCTION AND ITS DERIVATIVE
C     OF THE DIFFERENTIAL EQUATION ON INTERNAL WAVES
C     FPP + LAMBDA * (N ** 2 - SIGMA ** 2) * F = 0
C     WITH  FP + LAMBDA * F = 0  FOR Z = 0  AND  F = 0 FOR Z = 1
C     SO THAT THE INTEGRAL FROM 0 TO 1 OF  M * F ** 2  WILL BE 1.
C     A SIMPLE TRAPAZOIDAL RULE IS USED FOR THE INTEGRATION.
C
C     AM = ARRAY OF NP VALUES OF FUNCTION M
C     F = ARRAY OF NP VALUES OF EIGENFUNCTION F
C     FP = ARRAY OF NP VALUES OF DERIVATIVE OF THE EIGENFUNCTION
C     NP = NUMBER OF EVENLY SPACED POINTS IN THE CLOSED INTERVAL (0,1)
C     Q = NORMALIZATION FACTOR
C
      DIMENSION Z(NP)
      REAL AM(NP),F(NP),FP(NP)
      S1 = 0.0
      DO 1 I=2,NP
      S1 = S1 + 0.5*(AM(I-1)*F(I-1)*F(I-1) + AM(I)*F(I)*F(I))*(Z(I) -
     1  Z(I-1))
    1 CONTINUE
      Q = SQRT(1./S1)
      DO 2  I=1,NP
      F(I) = Q * F(I)
    2 FP(I) = Q * FP(I)
      RETURN
      END
```

```
      SUBROUTINE CORNS(NS,NPTS,H)
C
CC    TO CORRECT N**2 FOR SOUND VELOCITY.
C   SOUND VELOCITY IS CALCULATED FROM WILSONS FORNULA.
C
      REAL NS(51),TEMP(51),SAL(51)
C
CC    READ IN TEMPERATURE AND SALINITY ARRAYS.
      READ 100,(TEMP(I),I=1,NPTS)
  100 FORMAT(5F10.5)
      READ 100,(SAL(K),K=1,NPTS)
C
      DO 10 I=1,NPTS
      DEP = H*(FLOAT(I-1))/(NPTS-1)
      PRESS = DEP*0.980665
      SV = SNOVEL(PRESS,TEMP(I),SAL(I))
      NS(I) = NS(I) - H/(SV*SV)
   10 CONTINUE
      RETURN
      END
```

```
      FUNCTION  SNOVEL(PREST,TEMP,SAL)
C ***     FUNCTION SNOVEL CALCULATE SOUND VELOCITY USING WILSONS EQ.
C ***     PRESSURE IS CHANGED FROM DB UNITS TO ABSOLUTE PRESSURE IN
C ***        KG/CM**2
      PRESS=1.00323+0.1019716*PREST
      P2=PRESS*PRESS
      P3=P2*PRESS
      S35=SAL-35.
      T2=TEMP*TEMP
      T3=T2*TEMP
      VP=1.60272E-1*PRESS+1.0268E-5*P2+3.5216E-9*P3-3.3603E-12*P3*PRESS
      VS=S35*(1.39799+1.69202E-3*S35)
      VT=TEMP*(4.5721-4.4532E-2*TEMP-2.6045E-4*T2+7.9851E-6*T3)
      VSTP=S35*(-1.1244E-2*TEMP+7.7711E-7*T2+7.7016E-5*PRESS-1.2943E-7
     *    *P2+3.1580E-8*PRESS*TEMP+1.5790E-9*PRESS*T2)
     *    +PRESS*(-1.8607E-4*TEMP+7.4812E-6*T2+4.5283E-8*T3)
     *    +P2*(-2.5294E-7*TEMP+1.8563E-9*T2)+P3*(-1.9646E-10*TEMP)
      SNOVEL=1449.14+VP+VS+VT+VSTP
      RETURN
      END
```

```
      SUBROUTINE PDEN(NPTS,ST,BF,Z,ICARD,ITITLE,H)
C
CCC   TO PRODUCE A PLOT OF DENSITY AND BRUNT-VAISALA VERSUS DEPTH.
CCCCC
CCCCC NPTS - NO. OF POINTS IN THE VERTICAL.
CCCCC ST - ARRAY OF SIGMA-T VALUES.
CCCCC BF - ARRAY OF BRUNT-VAISALA FREQUENCIES.
CCCCC Z - ARRAY OF NORMALIXED DEPTHS.
CCCCC ICAARD - ARRAY OF CONTROL PARAMETERS FROM PLOT CARD.
CCCCC     -THIS ARRAY STARS IN COLUMN 5 OF THE PLOT CARD.
CCCCC    ZAX = LENGTH OF DEPTH AXIS IN INCHES - DEFAULT VALUE IS 5.0 IN
CCCCC   XAX = LENGTH OF HORIZONTAL AXIX IN INCHES - DEFOULT = 4.0.
CCCCC   SMIN = NINIMUM VALU OF SIGMA-T - DEFAULT =10.0.
CCCCC   SMAX = MAXINIM VALUE OF SIGMA-T - DEFAULT VALUE IS 30.0.
CCC     FMIN = MINIMUM B.V. FREQ. - DEFAULT VALUE IS 0.002 C/SEC.
        FMAX = MAXIMUM VALUE OF B.V.FREQ. - DEFOULT IS 0.02 C/SEC.
      DIMENSION ST(53),BF(53),ICARD(19),ITITLE(20),Z(53),IBCD(10)
      REAL H
C
CCC   DECODE ICARD ARRAY AND SORT OUT PLOT PARAMETERS.
C
      DECODE(76,100,ICARD) ZAX,XAX,SMIN,SMAX,FMIN,FMAX
  100 FORMAT(4(F5.3,5X),F10.9,5X,F10.9)
      IF(ZAX.EQ.0.0) ZAX=5.0
      IF(XAX.EQ.0.) XAX = 4.0
      IF(SMIN.EQ.0.) SMIN=10.0
      IF(SMAX.EQ.0.) SMAX = 30.0
      IF(FMAX.EQ.0.) FMAX=0.02
C
CCC    SCALE ARRAYS IN INCHES WITH ORIGIN TO BE AT TOP LEFT CORNER OF PL
C
      DO 10 I=1,NPTS
      Z(I) = -1.*ZAX*Z(I)
      ST(I) = XAX*(ST(I)-SMIN)/(SMAX-SMIN)
      BF(I) = XAX*(BF(I)-FMIN)/(FMAX-FMIN)
   10 CONTINUE
C
CCC   MOVE PLOT ORIGIN AND DRAW AXES.
C
      YPAGE = ZAX + 2.0
      CALL PLOT(0.0,YPAGE,-3)
      DELTV = H/ZAX
      CALL AXIS(0.0,0.0,5HDEPTH,-5,ZAX,-90.,0.0,DELTV)
      CALL AXIS(XAX,0.0,1H ,1,ZAX,-90.,0.0,DELTV)
      DELTV = (FMAX-FMIN)/XAX
      CALL AXIS(0.0,0.0,17HB-V FREQ. (C/SEC),+17,XAX,0.0,FMIN,DELTV)
      YPAGE = -1.0*ZAX
      DELTV =(SMAX-SMIN)/XAX
      CALL AXIS(0.0,YPAGE,7HSIGMA-T,-7,XAX,0.0,SMIN,DELTV)
C
CCCC  DRAW DENSITY PLOT
C
      IPEN = -1
      DO 15 I=1,NPTS
      INT = 17
      KK=(I/5)*5
      IF(KK.EQ.I) INT = 2
      CALL SYMBOL(ST(I),Z(I),0.07,INT,0.0,IPEN)
      IPEN = -2
   15 CONTINUE
C
CCC    DRAW B-V PLOT
C
      IPEN = -1
      DO 16 I=1,NPTS
      INT = 17
      KK = (I/5)*5
```

```
      IF(KK.EQ.I) INT=1
      CALL SYMBOL(8F(I),Z(I),0.07,INT,0.0,IPEN)
      IPEN = -2
   16 CONTINUE
C
CC    PUT PLOT ORIGIN BACK TO STARTING LOCATION AND PLOT TITLES.
C
      YPAGE = -1.*(ZAX+2)
      CALL PLOT(0.0,YPAGE,-3)
      CALL SYMBOL(0.0,1.0,0.07,24HTRIANGLE DENOTES SIGMA-T,0.0,24)
      ENCODE(40,200,IBCD) (ITITLE(I),I=1,10)
  200 FORMAT(10A4)
      CALL SYMBOL(0.0,0.5,0.14,IBCD,0.0,40)
      ENCODE(40,200,IBCD) ( ITITLE(I),I=11,20)
      CALL SYMBOL(0.0,0.0,0.14,IBCD,0.0,40)
C
C      MOVE PLOT ORIGIN FOR NEXT PLOT AND RESTORE ARRAY Z.
      XPAGE = XAX+3.0
      CALL PLOT(XPAGE,0.0,-3)
      DO 20 I=1,NPTS
   20 Z(I) = -Z(I)/ZAX
      RETURN
      END
```

```
      SUBROUTINE PEIF(NPTS,Z,H,EV,PHI,DPHI,ICARD,ITITLE)
C
C       TO PRODUCE PLOTS OF VARIOUS CONBINATIONS OF EIGENFUNCTIONS
C         AND THEIR DERIVATIVES VERSUS DEPTH.
C      NPTS = NUMBER OF POINTS IN THE VERTICAL.
C      Z = ARRAY OF NORMALIZED DEPTHE.
C      H = ACTUAL WATER DEPTH.
C     PHI = TWO DIMENSIONAL ARRAY CONTAINING EIGENFUNCTIONS.
C      DPHI = ARRRAY CONTAINING DERIVATIVES OF EIGENFUNCTIOSN.
C     ICARD = ARRAY OF CONTROL PARAMETERS FOR PLOT.
C         ZAX = LENGTH OF DEPTH AXIX IN INCHES. - DEFAULT = 5.0
CC        XAX = LENGTH OF HORIZONTAL AXIS IN INCHES - DEFAULT = 4.0.
C         NPL = NO. OF EIGENFUNCTIONS OR DERIVATIVES ON THIS PLOT.  MAX=1
C         IC,ICODE = SET OF NPL INTERGER PAIRS
C              IC = 1,EIGENFUNCTION IS OLOTTED.
C              IC = 2,DERIVATIVE OF EIGENFUNCTION IS PLOTTED.
C              IMODE = MODE NUMBER TO T BE PLOTTED.
C
C
      DIMENSION Z(53),EV(10),PHI(51,11),DPHI(51,11),ICARD(19)
      DIMENSION ITITLE(20),IBCD(10),FCT(53),IC(10),IMODE(10)
      REAL H
C
C     DECODE ICARD ARRAY AND SORT OUT PLOT PARAMETERS.
C
      DECODE(76,100,ICARD) ZAX,XAX,NPL,(IC(I),IMODE(I),I=1,10)
  100 FORMAT(2F5.3,I5,5X,10(I2,I2,1X))
      IF(ZAX.EQ.0.) ZAX = 5.0
      IF(XAX.EQ.0.0) XAX = 4.0
C
C     SCALE ZARRAY AND PLOT AXES
C
      DO 10 I=1,NPTS
      Z(I) = -1.*ZAX*Z(I)
   10 CONTINUE
      YPAGE = ZAX + 2.0
      CALL PLOT(0.0,YPAGE,-3)
      DELTV = H/ZAX
      CALL AXIS(0.0,0.0,5HDEPTH,-5,ZAX,-90.0,0.0,DELTV)
      CALL AXIS(XAX,0.0,1H ,1,ZAX,-90.0,0.0,DELTV)
      DELTV = 2.0/XAX
      CALL AXIS(0.0,0.0,1H ,1,XAX,0.0,-1.0,DELTV)
      YPAGE = -ZAX
      CALL AXIS(0.0,YPAGE,1H ,-1,XAX,0.0,-1.0,DELTV)
C
C     PLOT TITLE
CC
      ENCODE(40,200,IBCD) (ITITLE(I),I=1,10)
  200 FORMAT(10A4)
      YPAGE = -ZAX-1.5
      CALL SYMBOL(0.0,YPAGE,0.14,IBCD,0.0,40)
      ENCODE(40,200,IBCD) (ITITLE(I),I=11,20)
      YPAGE = YPAGE - 0.5
      CALL SYMBOL(0.0,YPAGE,0.14,IBCD,0.0,40)
C
C       PLACE ARRRAY TO BE PLOTTED INTO ARRAY FCT.
C
      DO 60 I =1,NPL
      J = IC(I)
      K = IMODE(I) + 1
      IF(J.EQ.1) GO TO 25
      IF(J.EQ.2) GO TO 35
      PRINT 300,I
  300 FORMAT(1H0,* UNRECOGNIZABLE CODE WHILE TRYING TO PLOT THE*,I5,
     1  *TH MODE*)
      GO TO 60
   25 DO 30 II = 1,NPTS
   30 FCT(II) = PHI(II,K)
```

```
      GO TO 45
   35 DO 40 II=1,NPTS
   40 FCT(II) = DPHI(II,K)
   45 CONTINUE
C
CCC      NORMALIZE FINCTION INTO INCHES.
C
      RMAX = 0.0
      DO 50 II = 1,NPTS
      RMAX = AMAX1(RMAX,ABS(FCT(II)))
   50 CONTINUE
      IF(RMAX.LT.1.E-8) GO TO 60
      DO 55 II =1,NPTS
      FCT(II) = (1.0 + (FCT(II)/RMAX))*(XAX/2.0)
   55 CONTINUE
C
CCC   PLOT ARRAY
C
      IPEN = -1
      DO 59 II = 1,NPTS
      JJ = 17
      CALL SYMBOL(FCT(II),Z(II),0.07,JJ,0.0,IPEN)
      IPEN = -2
   59 CONTINUE
   60 CONTINUE
C
CCC   RESCALE ARRAYS
CCC   REPLACE PLOT ORIGIN AND RETURN
C
      DO 65 I=1,NPTS
      Z(I) = -Z(I)/ZAX
   65 CONTINUE
      YPAGE = -ZAX - 2.0
      XPAGE = XAX + 3.0
      CALL PLOT(XPAGE,YPAGE,-3)
      RETURN
      END
```

```
      SUBROUTINE INPUT(NPTS,ST,Z,H)
C
      DIMENSION ST(51),Z(51)
      REAL H
C
C     READ IN DENSITIES FOR ST LAWRENCE TEST CASE
CC
      READ(60,100)(ST(K),K=1,NPTS)
  100 FORMAT(15F5.3)
C
      READ(60,106)(Z(I),I=1,NPTS)
  106 FORMAT(16F5.1)
C
      DO 50 I=1,NPTS
      Z(I) = Z(I)/H
   50 CONTINUE
C     ***************************************************************
      RETURN
      END
         FINIS

STOP
```

A P P E N D I X 2

- Programs PWAVE, QWAVE, ESURF -

Programmed

by

P. F. Hamblin and K. Beal

National Water Research Institute
Canada Centre for Inland Waters
P. O. Box 5050
Burlington, Ontario, Canada

```
      PROGRAM PWAVE
C
      REAL Y(2),DERY(2),PRMT(5),AUX(16,2)
      DIMENSION Z(72),XN2(72)
      DIMENSION A(5),PRMT4(5)
C
      COMMON/A/C,G
      COMMON/C/Z,XN2,P1,P2,BOTN2,TDEPTH,NO
C
      EXTERNAL FCT,OUTP
C
      DATA A/1.E-2,1.E-4,1.E-4,1.E-4,1.E-4/
      DATA PRMT4/.001,.001,.01,.01,.1/
      G=9.81
      NM=1
C
      WRITE (6,3001)
 3001 FORMAT (*Q*)
C
      WRITE (6,7210)
 7210 FORMAT(*1*,5X,*MODEL STRUCTURE OF INTERNAL WAVES FOR LAKE ONTARIO
     1 TEMPERATURE PROFILE*/)
C
      CALL GETNSQ
      DO 1060 K=1,NM
C
      WRITE (6,3030)
 3030 FORMAT (/1X,134(*-*),1X)
C
      DO 1030 I=1,5
      CALL ESTIM(I)
C
      CUPPER=C+(C/2.0)
      CLOWER=C-(C/2.0)
      CONST=CLOWER/20.0
C
      C=CLOWER
C
 1020 CONTINUE
      PRMT(1)=0.
      PRMT(2)=TDEPTH/2.0
      PRMT(3)=TDEPTH/200.0
      PRMT(4)=PRMT4(I)
      PRMT(5)=0.
      Y(1)=1.*A(I)
      Y(2)=(G/(C**2))*A(I)
      DERY(1)=.5
      DERY(2)=.5
      NDIM=2
      CALL HPCG(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)
      IHLF1=IHLF
      P=Y(1)
      Q=Y(2)
C
      PRMT(1)=TDEPTH
      PRMT(2)=TDEPTH/2.0
      PRMT(3)=-(TDEPTH/200.0)
      PRMT(4)=PRMT4(I)
      PRMT(5)=0.
      Y(1)=0.0
      Y(2)=SQRT(BOTN2)/C
      DERY(1)=.5
      DERY(2)=.5
      NDIM=2
      CALL HPCG(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)
      IHLF2=IHLF
      R=Y(1)
      S=Y(2)
```

```
C
      W=P*S-Q*R
C
      WRITE(6,17) C,W,P,Q,R,S
 17   FORMAT (/10X,6(2X,E18.10))
      WRITE (6,3010) IHLF1,IHLF2
3010  FORMAT (10X,*  IHLF1=*,I3,10X,*  IHLF2=*,I3)
C
      C=C+CONST
      IF(C.LE.CUPPER) GO TO 1020
C
      WRITE(6,3030)
C
 1030 CONTINUE
C
 1060 CONTINUE
C
      STOP
      END
```

```
SUBROUTINE FCT(X,Y,DERY)
REAL Y(2),DERY(2)
COMMON/A/C,G
DERY(1)=-1.0*Y(2)
DERY(2)=(HH(X)*Y(1))/C**2
RETURN
END
```

```
      SUBROUTINE OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      REAL Y(2),DERY(2),PRMT(5)
      RETURN
      END
```

```
      SUBROUTINE ESTIM(I)
      DIMENSION Z(72),XN2(72)
      COMMON/A/C,G
      COMMON/C/Z,XN2,P1,P2,BOTN2,TDEPTH,NO
C-    CALCULATES AN ESTIMATE (C) FOR MODE (I).
      GO TO (1,2,3,4,5),I
 1    C=SQRT(G*TDEPTH)
      GO TO 6
 2    CC=SQRT((G*(P2-P1)/P2)*(TDEPTH/4.0))
      C=CC
      GO TO 6
 3    C=CC/3.0
      GO TO 6
 4    C=CC/5.0
      GO TO 6
 5    C=CC/7.0
 6    WRITE( 6,10) C
 10   FORMAT(1X,*ACTUAL C = *,E18.10//)
      RETURN
      END
```

```
      FUNCTION HH(X)
      DIMENSION Z(72),XN2(72)
      COMMON/C/Z,XN2,P1,P2,BOTN2,TDEPTH,NO
      IF(X.NE.Z(1)) GO TO 10
      HH=XN2(1)
      RETURN
10    CONTINUE
      N=NO-1
      DO 20  I=1,N
      IF(X.GT.Z(I).AND.X.LE.Z(I+1)) GO TO 30
20    CONTINUE
      WRITE(6,100) X
100   FORMAT(1X,F7.2)
      I=N
30    HH=XN2(I)
      RETURN
      END
```

```
      SUBROUTINE GETNSQ
      DIMENSION T(72),Z(72),RHO(72),DRT(72),XN2(72)
      REAL ID(5)
      COMMON/C/Z,XN2,P1,P2,BOTN2,TDEPTH,NO
C
C     READ THE TEMPERATURE PROFILE
      READ(5,5)NO
 5    FORMAT(I3)
      IF(NO.GT.72)STOP 123
      Z(1) =0.0
      READ(5,10) (ID(I),I=1,5),T(1),(T(I),Z(I),I=2,NO)
   10 FORMAT(A3,A6,A3,A5,A5,F3.1,8(F3.1,F3.0)/12(F3.1,F3.0)/12(F3.1,F3.0
     1 ))
C
      DO 20 I=1,NO
      CALL DENS(T(I),Z(I),RHO(I),DRT(I))
 20   CONTINUE
C
C     STORE BOTTOM DEPTH
      TDEPTH=Z(NO)
C
C     STORE SURFACE AND BOTTOM DENSITY
      P1=RHO(1)
      P2=RHO(NO)
C
      G=+9.81
      N=NO-1
      DO 30 I=1,N
      XN2(I)=(G/(RHO(I)+RHO(I+1)))*(DRT(I)+DRT(I+1))*((T(I+1)-T(I))/(Z(I
     A+1)-Z(I)))
 30   CONTINUE
C
C     STORE XN2 AT BOTTOM
C     IF 0 LOOK FOR NEXT NON ZERO VALUE
      K=N
   40 BOTN2=ABS(XN2(K))
      IF(BOTN2.NE.0.0) GO TO 50
      K=K-1
      GO TO 40
C
   50 WRITE(6,90)(ID(I),I=1,5)
   90 FORMAT(2X,*STA. NO.*,1X,A3,2X,*DAY*,1X,A8,2X,*TIME*,1X,A4,2X,*LAT.
     1*,1X,A6,2X,*LONG.*,1X,A6/)
      WRITE(6,100)
 100  FORMAT(2X,*TEMP*,2X,*DEPTH*,6X,*RHO*,10X,*DRT*,12X,*XN2*/)
      DO 110 I=1,N
      WRITE(6,120) T(I),Z(I),RHO(I),DRT(I),XN2(I)
 110  CONTINUE
 120  FORMAT(1X,F6.2,1X,F6.2,1X,F10.6,3X,F10.6/40X,E18.10)
      WRITE(6,130) T(NO),Z(NO),RHO(NO),DRT(NO)
 130  FORMAT(1X,F6.2,1X,F6.2,1X,F10.6,3X,F10.6)
C
      RETURN
      END
```

```
      SUBROUTINE DENS(T,Z,RHO,DRT)
      DIMENSION P(4),RL(4),VO(9),TP(9)
      DATA(P(K),K=1,4)/58.05267,-1.1253317,.66123869E-02,-.14661625E-04/
      DATA(RL(K),K=1,4)/21.55053,-.4695911,.003096363,-.7431182E-05/
      DATA(VO(K),K=1,9)/-.7435626E-03,.3704258E-04,-.6315724E-06,.982957
     16E-08,-.1197269E-09,.1005461E-11,-.5437898E-14,.169946E-16,-.22950
     263E-19/
      DATA POO,PO1,RLO,VOO/5919.512,.0981,1788.316,.6980547/
C     EQUATION OF STATE FOR WATER AS GIVEN BY FISHER, WILLIAMS AND DIAL
CC    IN FIFTH REPORT OF JOINT PANEL ON OCEANOGRAPHIC TABLES AND STAND-
CC    ARDS, KIEL  DECEMBER 1969 (UNESCO). EQUATION IS OF THE TUMLIRZ
C     TYPE FOR SPECIFIC VOLUME V
      TP(1)=T
      DO 100 K=2,9
      TP(K)=T*TP(K-1)
  100 CONTINUE
      SV=VOO
      SP=POO
      SRL=RLO
      DO 101 K=1,9
      TT=TP(K)
      IF(K.GT.4)GOTO 102
      SP=SP+P(K)*TT
      SRL=SRL+RL(K)*TT
  102 SV=SV+VO(K)*TT
  101 CONTINUE
      DSV=VO(1)
      DSRL=RL(1)
      DSP=P(1)
      DO 103 K=1,8
      TT=TP(K)*FLOAT(K+1)
      IF(K.GT.3)GO TO 104
      DSRL=DSRL+RL(K+1)*TT
      DSP=DSP+P(K+1)*TT
  104 DSV=DSV+VO(K+1)*TT
  103 CONTINUE
      PP=SP+PO1*Z
      V=SV+SRL/PP
      VT=DSV+(PP*DSRL-SRL*DSP)/(PP*PP)
      RHO=1./V
      DRT=-VT*RHO*RHO
      RETURN
      END
         FINIS
```

```
      PROGRAM QWAVE
C
      DIMENSION ESTIM(100),W(100)
      DIMENSION Y(2),DERY(2),PRMT(5),AUX(16,2)
      DIMENSION FSTIM(5),A(5),PRMT4(5)
      DIMENSION Z(72),XN2(72)
C
      COMMON/A/C
      COMMON/C/Z,XN2,BOTN2,TDEPTH,NO
C
      EXTERNAL FCT,OUTP
C
      DATA A/1.E-2,1.E-4,1.E-4,1.E-4,1.E-4/
      DATA PRMT4/.001,.001,.001,.001,.01/
      READ(5,1070) FSTIM(1)
      DO 1050 L=1,1
 1070 FORMAT(E14.10)
      G=9.81
      EST3= 1.0005
      RLIM=0.00000000001
C
C
      WRITE (6,7210)
 7210 FORMAT(*1*,5X,*MODEL STRUCTURE OF INTERNAL WAVES FOR LAKE ONTARIO
     1 TEMPERATURE PROFILE*/)
C
      CALL GETNSQ
C
C
      WRITE(6,1080) FSTIM(1)
 1080 FORMAT(*1*/19X,E17.10//)
      DO 1020 N=1,1
C
C
      WRITE (6,3030)
 3030 FORMAT (/1X,134(*-*),1X)
C
      ESTIM(1)=FSTIM(N)
      IF(ESTIM(1).EQ.0.0) GO TO 1020
      DO 1030 I=1,100
C
      C=ESTIM(I)
      PRMT(1)=0.0
      PRMT(2)=TDEPTH/2.0
      PRMT(3)=TDEPTH/200.0
      PRMT(4)=PRMT4(N)
      PRMT(5)=0.
      Y(1)=1.*A(N)
      Y(2)=(G/(C**2))*A(N)
      DERY(1)=.5
      DERY(2)=.5
      NDIM=2
      CALL HPCG(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)
      IHLF1=IHLF
      P=Y(1)
      Q=Y(2)
C
      C=ESTIM(I)
      PRMT(1)=TDEPTH
      PRMT(2)=TDEPTH/2.0
      PRMT(3)=-(TDEPTH/200.0)
      PRMT(4)=PRMT4(N)
      PRMT(5)=0.
      Y(1)=0.0
      Y(2)=SQRT(BOTN2)/C
      DERY(1)=.5
      DERY(2)=.5
      NDIM=2
```

```
      CALL HPCG(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)
      IHLF2=IHLF
      R=Y(1)
      S=Y(2)
C
      W(I)=P*S-Q*R
C
      WRITE(6,17)ESTIM(I),W(I),P,Q,R,S
 17   FORMAT (/10X,6(2X,E18.10))
      WRITE (6,3010) IHLF1,IHLF2
3010  FORMAT (10X,*  IHLF1=*,I3,10X,*  IHLF2=*,I3)
C
      DIFF=W(I)-W(I-1)
      IF(ABS(DIFF).LT.RLIM) PS = ESTIM(I)
      IF(ABS(DIFF).LT.RLIM) GO TO 1020
C
1015  IF (I.EQ.1) ESTIM(2)=ESTIM(1)*EST3
      IF (I.GT.1) ESTIM(I+1)=ESTIM(I)-((W(I)*(ESTIM(I)-ESTIM(I-1)))/
     $  (W(I)-W(I-1)))
1030  CONTINUE
C
1020  CONTINUE
      H=PS**2/G
      WRITE(61,12) PS,H
  12  FORMAT(1H-,2X,*PHASE SPEED = *,1X,E18.10,2X,*DEPTH = *,1X,E18.10)
1050  CONTINUE
      STOP
      END
```

```
      FUNCTION HH(X)
      DIMENSION Z(72),XN2(72)
      COMMON/C/Z,XN2,BOTN2,TDEPTH,NO
      IF(X.NE.Z(1)) GO TO 10
      HH=XN2(1)
      RETURN
10    CONTINUE
      N=NO-1
      DO 20 I=1,N
      IF(X.GT.Z(I).AND.X.LE.Z(I+1)) GO TO 30
20    CONTINUE
      STOP 111
30    HH=XN2(I)
      RETURN
      END
```

```
      SUBROUTINE GETNSQ
      DIMENSION T(72),Z(72),RHO(72),DRT(72),XN2(72)
      DIMENSION ID(5),D(72),RH(72),DT(72),TA(72)
      REAL ID
      COMMON/C/Z,XN2,BOTN2,TDEPTH,NO
C
C     READ THE TEMPERATURE PROFILE
      READ(5,5) NO
    5 FORMAT(I3)
      IF(NO.GT.72) STOP 123
      Z(1)=0.0
      READ(5,10) (ID(I),I=1,5),T(1),(T(I),Z(I),I=2,NO)
   10 FORMAT(A3,A6,A3,A5,A5,F3.1,8(F3.1,F3.0))
C
      DO 20 I=1,NO
      CALL DENS(T(I),Z(I),RHO(I),DRT(I))
   20 CONTINUE
C
C     STORE BOTTOM DEPTH
      TDEPTH=Z(NO)
C
      G=+9.81
      N=NO-1
      DO 30 I=1,N
      XN2(I)=(G/(RHO(I)+RHO(I+1)))*(DRT(I)+DRT(I+1))*((T(I+1)-T(I))/(Z(I
     A+1)-Z(I)))
   30 CONTINUE
C
C     STORE XN2 AT BOTTOM
C     IF 0 LOOK FOR NEXT NON ZERO VALUE
      K=N
   40 BOTN2=ABS(XN2(K))
      IF(BOTN2.NE.0.0) GO TO 50
      K=K-1
      GO TO 40
C
   50 WRITE(6,90)(ID(I),I=1,5)
   90 FORMAT(2X,*STA. NO.*,1X,A3,2X,*DAY*,1X,A8,2X,*TIME*,1X,A4,2X,*LAT.
     1*,1X,A6,2X,*LONG.*,1X,A6/)
      WRITE(6,100)
  100 FORMAT(2X,*TEMP*,2X,*DEPTH*,6X,*RHO*,10X,*DRT*,12X,*XN2*/)
      DO 110 I=1,N
      WRITE(6,120) T(I),Z(I),RHO(I),DRT(I),XN2(I)
      TA(I)=(T(I)+T(I+1))/2
      D(I)=(Z(I)+Z(I+1))/2
      RH(I)=(RHO(I)+RHO(I+1))/2
      DT(I)=(DRT(I)+DRT(I+1))/2
  110 CONTINUE
  120 FORMAT(1X,F6.2,1X,F6.2,1X,F10.6,3X,F10.6/40X,E18.10)
      WRITE(6,130) T(NO),Z(NO),RHO(NO),DRT(NO)
  130 FORMAT(1X,F6.2,1X,F6.2,1X,F10.6,3X,F10.6)
C
      RETURN
      END
```

```
      PROGRAM ESURF
C
      REAL                    Y(2),PRMT(5),DERY(2)   ,RI(301)
      REAL YY(2,301),AUX(16,2)
      REAL XX(301)
      DIMENSION A(5),PRMT4(5)   ,AN(320)
      DIMENSION Z(72),XN2(72)
      COMMON/A/C
      COMMON/B/YY,XX,XINC,INDEX,XNO
      COMMON/C/Z,XN2,BOTN2,TDEPTH,NO
      EXTERNAL FCT,OUTP
C
      DATA A/1.E-2,1.E-4,1.E-4,1.E-4,1.E-4/
      DATA PRMT4/.001,.001,.001,.001,.01/
      DO 1111 L=1,1
      G=9.81
C
      WRITE (6,3001)
 3001 FORMAT (*Q*)
C
      CALL GETNSQ
      DO 5000 N=1,1
 1000 READ (5,1070) RM,SIGMA
 1070 FORMAT (    2E20.10)
      IF ((RM.LT.1.0E-99).AND.(SIGMA.LT.1.0E-99)) GO TO 9999
C
      C=SIGMA
      PRMT(1)=0.0
      PRMT(2)=TDEPTH
      PRMT(3)=TDEPTH/100.0
      PRMT(4)=PRMT4(N)
      PRMT(5)=0.0
      Y(1)=1.0*A(N)
      YY(1,1)=Y(1)
      XX(1)=0.0
      Y(2)=(G/(C**2))*A(N)
      YY(2,1)=Y(2)
      DERY(1)=0.5
      DERY(2)=0.5
      NDIM=2
      INDEX=0
      XINC=PRMT(3)
      XNO=0
      CALL HPCG(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)
      IHLF1=IHLF
C
C
      WRITE (6,2010) RM,SIGMA,IHLF1
 2010 FORMAT (*1*,09X,*WAVE NUMBER*,5X,F9.5
     $      /10X,*SIGMA*,5X,E16.10
     $      /10X,*IHLF1*,5X,I2)
C
C
      XMIN=XX(1)
      XMAX=XX(1)
      YMIN1=YY(1,1)
      YMAX1=YY(1,1)
      YMIN2=YY(2,1)
      YMAX2=YY(2,1)
CALCULATE GRADIENT RICHARDSON NO. INDUCED BY MODE (RELATIVE)
      DO 3 I=1,INDEX
      P=HH( XX(I))
      AN(I)=P
      IF(P.EQ..0) P=1.E-06
      RI(I)=C**4/(P*YY(1,I)**2)
      RI(I)=ALOG10(RI(I))
      WRITE(6,4) I, YY(1,I),RI(I),P
    4 FORMAT(I8,3E10.3)
```

```
  3    CONTINU E
        YMIN3=RI(1)
       YMAX3=RI(1)
       YMIN4=AN(1)
       YMAX4=AN(1)
       DO 2000 I=2,INDEX
       IF(RI(I).GT.YMAX3) YMAX3= RI(I)
       IF(AN(I).LT.YMIN4) YMIN4=AN (I)
       IF(AN(I).GT.YMAX4) YMAX4=AN (I)
       IF(RI(I).LT.YMIN3) YMIN3= RI(I)
       IF(XX(I).LT.XMIN)XMIN=XX(I)
       IF(XX(I).GT.XMAX)XMAX=XX(I)
       IF(YY(1,I).LT.YMIN1)YMIN1=YY(1,I)
       IF(YY(1,I).GT.YMAX1)YMAX1=YY(1,I)
       IF(YY(2,I).LT.YMIN2)YMIN2=YY(2,I)
       IF(YY(2,I).GT.YMAX2)YMAX2=YY(2,I)
 2000  CONTINUE
       DO 1040 I=1,INDEX
       CALL PLOTPT(XX(I),YY(1,I),36)
 1040  CONTINUE
       CALL SCALE (0.0,TDEPTH,YMIN1,YMAX1)
       CALL OUTPLT
       DO 1050 I=1,INDEX
       CALL PLOTPT(XX(I),YY(2,I),37)
 1050  CONTINUE
       CALL SCALE (0.0,TDEPTH,YMIN2,YMAX2)
       CALL OUTPLT
       DO 8 I=1,INDEX
  8    CALL PLOTPT(XX(I),RI(I),38)
       CALL SCALE(0.0,TDEPTH,YMIN3,YMAX3)
       CALL OUTPLT
       DO 9 I=1,INDEX
       P=AN(I)
  9    CALL PLOTPT(XX(I),P,35)
       CALL SCALE(0.0,TDEPTH,YMIN4,YMAX4)
       CALL OUTPLT
C
       WRITE (6,3210)
 3210  FORMAT (////////////1X,134(*-*),1X)
C
 5000  CONTINUE
C
 1111  CONTINUE
 9999  STOP
       END
```

```
      FUNCTION HH(X)
      DIMENSION Z(72),XN2(72)
      COMMON/C/Z,XN2,BOTN2,TDEPTH,NO
      IF(X.NE.Z(1)) GO TO 10
      HH=XN2(1)
      RETURN
10    CONTINUE
      DO 20 I=1,71
      IF(X.GT.Z(I).AND.X.LE.Z(I+1)) GO TO 30
20    CONTINUE
      I=71
30    HH=XN2(I)
      RETURN
      END
```

```
      SUBROUTINE GETNSQ
      DIMENSION T(72),Z(72),RHO(72),DRT(72),XN2(72)
      REAL ID(5)
      COMMON/C/Z,XN2,BOTN2,TDEPTH,NO
C
C     READ THE TEMPERATURE PROFILE
      READ(5,5) NO
 5    FORMAT(I3)
      IF(NO.GT.72) STOP 123
      Z(1) =0.0
      READ(5,10) (ID(I),I=1,5),T(1),(T(I),Z(I),I=2,NO)
   10 FORMAT(A3,A6,A3,A5,A5,F3.1,8(F3.1,F3.0)/12(F3.1,F3.0)/12(F3.1,F3.0
     1 ))
C
      DO 20 I=1,NO
      CALL DENS(T(I),Z(I),RHO(I),DRT(I))
 20   CONTINUE
C     STORE BOTTOM DEPTH
      TDEPTH=Z(NO)
C
C     STORE SURFACE AND BOTTOM DENSITY
      P1=RHO(1)
      P2=RHO(NO)
C
      G=+9.81
      N=NO-1
      DO 30 I=1,N
      XN2(I)=(G/(RHO(I)+RHO(I+1)))*(DRT(I)+DRT(I+1))*((T(I+1)-T(I))/(Z(I
     A+1)-Z(I)))
 30   CONTINUE
C
C     STORE XN2 AT BOTTOM
C     IF 0 LOOK FOR NEXT NON ZERO VALUE
      K=N
 40   BOTN2=XN2(K)
      IF(BOTN2.NE.0.0) GO TO 50
      K=K-1
      GO TO 40
C
   50 WRITE(6,90) (ID(I),I=1,5)
   90 FORMAT(2X,*STA. NO.*,1X,A3,2X,*DAY*,1X,A8,2X,*TIME*,1X,A4,2X,*LAT.
     1*,1X,A6,2X,*LONG.*,1X,A6/)
      WRITE(6,100)
 100  FORMAT(2X,*TEMP*,2X,*DEPTH*,6X,*RHO*,10X,*DRT*,12X,*XN2*/)
      DO 110 I=1,N
      WRITE(6,120) T(I),Z(I),RHO(I),DRT(I),XN2(I)
 110  CONTINUE
 120  FORMAT(1X,F6.2,1X,F6.2,1X,F10.6,3X,F10.6/40X,E18.10)
      WRITE(6,130) T(NO),Z(NO),RHO(NO),DRT(NO)
 130  FORMAT(1X,F6.2,1X,F6.2,1X,F10.6,3X,F10.6//)
C
      RETURN
      END
```

```
      SUBROUTINE OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      REAL Y(2),DERY(2),PRMT(5)
      REAL YY(2,301)
      REAL XX(301)
      COMMON/B/YY,XX,XINC,INDEX,XNO
C
      IF(X.LT.XNO) GO TO 9999
      INDEX=INDEX+1
      XX(INDEX)=X
      YY(1,INDEX)=Y(1)
      YY(2,INDEX)=Y(2)
      XNO=XNO+XINC
      WRITE(6,10)X,Y(1),Y(2)
 10   FORMAT(1X,F6.2,2X,E18.10,3X,E18.10)
 9999 RETURN
      END
         FINIS
```

```
      SUBROUTINE MBS004(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)
         SUBROUTINE HPCG

         PURPOSE
            TO SOLVE A SYSTEM OF FIRST ORDER ORDINARY GENERAL
            DIFFERENTIAL EQUATIONS WITH GIVEN INITIAL VALUES.

         USAGE
            CALL HPCG (PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)
            PARAMETERS FCT AND OUTP REQUIRE AN EXTERNAL STATEMENT.

         DESCRIPTION OF PARAMETERS
            PRMT    - AN INPUT AND OUTPUT VECTOR WITH DIMENSION GREATER
                      OR EQUAL TO 5, WHICH SPECIFIES THE PARAMETERS OF
                      THE INTERVAL AND OF ACCURACY AND WHICH SERVES FOR
                      COMMUNICATION BETWEEN OUTPUT SUBROUTINE (FURNISHED
                      BY THE USER) AND SUBROUTINE HPCG. EXCEPT PRMT(5)
                      THE COMPONENTS ARE NOT DESTROYED BY SUBROUTINE
                      HPCG AND THEY ARE
            PRMT(1)- LOWER BOUND OF THE INTERVAL (INPUT),
            PRMT(2)- UPPER BOUND OF THE INTERVAL (INPUT),
            PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE
                     (INPUT),
            PRMT(4)- UPPER ERROR BOUND (INPUT). IF ABSOLUTE ERROR IS
                     GREATER THAN PRMT(4), INCREMENT GETS HALVED.
                     IF INCREMENT IS LESS THAN PRMT(3) AND ABSOLUTE
                     ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.
                     THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS
                     OUTPUT SUBROUTINE.
            PRMT(5)- NO INPUT PARAMETER. SUBROUTINE HPCG INITIALIZES
                     PRMT(5)=0. IF THE USER WANTS TO TERMINATE
                     SUBROUTINE HPCG AT ANY OUTPUT POINT, HE HAS TO
                     CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE
                     OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE
                     FEASIBLE IF ITS DIMENSION IS DEFINED GREATER
                     THAN 5. HOWEVER SUBROUTINE HPCG DOES NOT REQUIRE
                     AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL
                     FOR HANDING RESULT VALUES TO THE MAIN PROGRAM
                     (CALLING HPCG) WHICH ARE OBTAINED BY SPECIAL
                     MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP.
            Y       - INPUT VECTOR OF INITIAL VALUES.  (DESTROYED)
                      LATERON Y IS THE RESULTING VECTOR OF DEPENDENT
                      VARIABLES COMPUTED AT INTERMEDIATE POINTS X.
            DERY    - INPUT VECTOR OF ERROR WEIGHTS.  (DESTROYED)
                      THE SUM OF ITS COMPONENTS MUST BE EQUAL TO 1.
                      LATERON DERY IS THE VECTOR OF DERIVATIVES, WHICH
                      BELONG TO FUNCTION VALUES Y AT A POINT X.
            NDIM    - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF
                      EQUATIONS IN THE SYSTEM.
            IHLF    - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF
                      BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS
                      GREATER THAN 10, SUBROUTINE HPCG RETURNS WITH
                      ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.
                      ERROR MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE
                      PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-
                      PRMT(1)) RESPECTIVELY.
            FCT     - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT
                      COMPUTES THE RIGHT HAND SIDES DERY OF THE SYSTEM
                      TO GIVEN VALUES OF X AND Y. ITS PARAMETER LIST
                      MUST BE X,Y,DERY. THE SUBROUTINE SHOULD NOT
                      DESTROY X AND Y.
            OUTP    - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.
                      ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT.
                      NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,
                      PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY
                      SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO,
                      SUBROUTINE HPCG IS TERMINATED.
            AUX     - AN AUXILIARY STORAGE ARRAY WITH 16 ROWS AND NDIM
```

```
C                    COLUMNS.
C
C         REMARKS
C            THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF
C            (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE
C                NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE
C                IHLF=11),
C            (2) INITIAL INCREMENT IS EQUAL TO 0 OR HAS WRONG SIGN
C                (ERROR MESSAGES IHLF=12 OR IHLF=13),
C            (3) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,
C            (4) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.
C
C         SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C            THE EXTERNAL SUBROUTINES FCT(X,Y,DERY) AND
C            OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED BY THE USER.
C
C         METHOD
C            EVALUATION IS DONE BY MEANS OF HAMMINGS MODIFIED PREDICTOR-
C            CORRECTOR METHOD. IT IS A FOURTH ORDER METHOD, USING 4
C            PRECEEDING POINTS FOR COMPUTATION OF A NEW VECTOR Y OF THE
C            DEPENDENT VARIABLES.
C            FOURTH ORDER RUNGE-KUTTA METHOD SUGGESTED BY RALSTON IS
C            USED FOR ADJUSTMENT OF THE INITIAL INCREMENT AND FOR
C            COMPUTATION OF STARTING VALUES.
C            SUBROUTINE HPCG AUTOMATICALLY ADJUSTS THE INCREMENT DURING
C            THE WHOLE COMPUTATION BY HALVING OR DOUBLING.
C            TO GET FULL FLEXIBILITY IN OUTPUT, AN OUTPUT SUBROUTINE
C            MUST BE CODED BY THE USER.
C            FOR REFERENCE, SEE
C            (1)   RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL
C                  COMPUTERS, WILEY, NEW YORK/LONDON, 1960, PP.95-109.
C            (2)   RALSTON, RUNGE-KUTTA METHODS WITH MINIMUM ERROR BOUNDS,
C                  MTAC, VOL.16, ISS.80 (1962), PP.431-437.
C
C     ..................................................................
C***     NAME OF HPCG CHANGED TO MBS004 FOR EMRLIB
C
      DIMENSION PRMT(1),Y(1),DERY(1),AUX(16,1)
      N=1
      IHLF=0
      X=PRMT(1)
      H=PRMT(3)
      PRMT(5)=0.
      DO 1 I=1,NDIM
      AUX(16,I)=0.
      AUX(15,I)=DERY(I)
    1 AUX(1,I)=Y(I)
      IF(H*(PRMT(2)-X))3,2,4
C
C     ERROR RETURNS
    2 IHLF=12
      GOTO 4
    3 IHLF=13
C
C     COMPUTATION OF DERY FOR STARTING VALUES
    4 CALL FCT(X,Y,DERY)
C
C     RECORDING OF STARTING VALUES
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))6,5,6
    5 IF(IHLF)7,7,8
    6 RETURN
    7 DO 8 I=1,NDIM
    8 AUX(8,I)=DERY(I)
C
C     COMPUTATION OF AUX(2,I)
```

```
      ISW=1
      GOTO 100
C
    9 X=X+H
      DO 10 I=1,NDIM
   10 AUX(2,I)=Y(I)
C
C     INCREMENT H IS TESTED BY MEANS OF BISECTION
   11 IHLF=IHLF+1
      X=X-H
      DO 12 I=1,NDIM
   12 AUX(4,I)=AUX(2,I)
      H=.5*H
      N=1
      ISW=2
      GOTO 100
C
   13 X=X+H
      CALL FCT(X,Y,DERY)
      N=2
      DO 14 I=1,NDIM
      AUX(2,I)=Y(I)
   14 AUX(9,I)=DERY(I)
      ISW=3
      GOTO 100
C
C     COMPUTATION OF TEST VALUE DELT
   15 DELT=0.
      DO 16 I=1,NDIM
   16 DELT=DELT+AUX(15,I)*ABS(Y(I)-AUX(4,I))
      DELT=.06666667*DELT
      IF(DELT-PRMT(4))19,19,17
   17 IF(IHLF-10)11,18,18
C
C     NO SATISFACTORY ACCURACY AFTER 10 BISECTIONS. ERROR MESSAGE.
   18 IHLF=11
      X=X+H
      GOTO 4
C
C     THERE IS SATISFACTORY ACCURACY AFTER LESS THAN 11 BISECTIONS.
   19 X=X+H
      CALL FCT(X,Y,DERY)
      DO 20 I=1,NDIM
      AUX(3,I)=Y(I)
   20 AUX(10,I)=DERY(I)
      N=3
      ISW=4
      GOTO 100
C
   21 N=1
      X=X+H
      CALL FCT(X,Y,DERY)
      X=PRMT(1)
      DO 22 I=1,NDIM
      AUX(11,I)=DERY(I)
   22 Y(I)=AUX(1,I)+H*(.375*AUX(8,I)+.7916667*AUX(9,I)
     1-.2083333*AUX(10,I)+.04166667*DERY(I))
   23 X=X+H
      N=N+1
      CALL FCT(X,Y,DERY)
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))6,24,6
   24 IF(N-4)25,200,200
   25 DO 26 I=1,NDIM
      AUX(N,I)=Y(I)
   26 AUX(N+7,I)=DERY(I)
      IF(N-3)27,29,200
C
```

```
   27 DO 28 I=1,NDIM
      DELT=AUX(9,I)+AUX(9,I)
      DELT=DELT+DELT
   28 Y(I)=AUX(1,I)+.3333333*H*(AUX(8,I)+DELT+AUX(10,I))
      GOTO 23
C
   29 DO 30 I=1,NDIM
      DELT=AUX(9,I)+AUX(10,I)
      DELT=DELT+DELT+DELT
   30 Y(I)=AUX(1,I)+.375*H*(AUX(8,I)+DELT+AUX(11,I))
      GOTO 23
C
C
C     ******************************************************************
C     THE FOLLOWING PART OF SUBROUTINE HPCG COMPUTES BY MEANS OF
C     RUNGE-KUTTA METHOD STARTING VALUES FOR THE NOT SELF-STARTING
C     PREDICTOR-CORRECTOR METHOD.
  100 DO 101 I=1,NDIM
      Z=H*AUX(N+7,I)
      AUX(5,I)=Z
  101 Y(I)=AUX(N,I)+.4*Z
C     Z IS AN AUXILIARY STORAGE LOCATION
C
      Z=X+.4*H
      CALL FCT(Z,Y,DERY)
      DO 102 I=1,NDIM
      Z=H*DERY(I)
      AUX(6,I)=Z
  102 Y(I)=AUX(N,I)+.2969776*AUX(5,I)+.1587596*Z
C
      Z=X+.4557372*H
      CALL FCT(Z,Y,DERY)
      DO 103 I=1,NDIM
      Z=H*DERY(I)
      AUX(7,I)=Z
  103 Y(I)=AUX(N,I)+.2181004*AUX(5,I)-3.050965*AUX(6,I)+3.832865*Z
C
      Z=X+H
      CALL FCT(Z,Y,DERY)
      DO 104 I=1,NDIM
  104 Y(I)=AUX(N,I)+.1747603*AUX(5,I)-.5514807*AUX(6,I)
     1+1.205536*AUX(7,I)+.1711848*H*DERY(I)
      GOTO(9,13,15,21),ISW
C     ******************************************************************
C
C
C     POSSIBLE BREAK-POINT FOR LINKAGE
C
C
C     STARTING VALUES ARE COMPUTED.
C     NOW START HAMMINGS MODIFIED PREDICTOR-CORRECTOR METHOD.
  200 ISTEP=3
  201 IF(N-8)204,202,204
C
C     N=8 CAUSES THE ROWS OF AUX TO CHANGE THEIR STORAGE LOCATIONS
  202 DO 203 N=2,7
      DO 203 I=1,NDIM
      AUX(N-1,I)=AUX(N,I)
  203 AUX(N+6,I)=AUX(N+7,I)
      N=7
C
C     N LESS THAN 8 CAUSES N+1 TO GET N
  204 N=N+1
C
C     COMPUTATION OF NEXT VECTOR Y
      DO 205 I=1,NDIM
      AUX(N-1,I)=Y(I)
  205 AUX(N+6,I)=DERY(I)
```

```
      X=X+H
  206 ISTEP=ISTEP+1
      DO 207 I=1,NDIM
     0DELT=AUX(N-4,I)+1.333333*H*(AUX(N+6,I)+AUX(N+6,I)-AUX(N+5,I)+
     1AUX(N+4,I)+AUX(N+4,I))
      Y(I)=DELT-.9256198*AUX(16,I)
  207 AUX(16,I)=DELT
C     PREDICTOR IS NOW GENERATED IN ROW 16 OF AUX, MODIFIED PREDICTOR
CC    IS GENERATED IN Y. DELT MEANS AN AUXILIARY STORAGE.
C
      CALL FCT(X,Y,DERY)
C     DERIVATIVE OF MODIFIED PREDICTOR IS GENERATED IN DERY
C
      DO 208 I=1,NDIM
     0DELT=.125*(9.*AUX(N-1,I)-AUX(N-3,I)+3.*H*(DERY(I)+AUX(N+6,I)+
     1AUX(N+6,I)-AUX(N+5,I)))
      AUX(16,I)=AUX(16,I)-DELT
  208 Y(I)=DELT+.07438017*AUX(16,I)
CC
CC    TEST WHETHER H MUST BE HALVED OR DOUBLED
      DELT=0.
      DO 209 I=1,NDIM
  209 DELT=DELT+AUX(15,I)*ABS(AUX(16,I))
      IF(DELT-PRMT(4))210,222,222
C
CC    H MUST NOT BE HALVED. THAT MEANS Y(I) ARE GOOD.
  210 CALL FCT(X,Y,DERY)
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))212,211,212
  211 IF(IHLF-11)213,212,212
  212 RETURN
  213 IF(H*(X-PRMT(2)))214,212,212
  214 IF(ABS(X-PRMT(2))-.1*ABS(H))212,215,215
  215 IF(DELT-.02*PRMT(4))216,216,201
CCC
CCC   H COULD BE DOUBLED IF ALL NECESSARY PRECEEDING VALUES ARE
CC    AVAILABLE
  216 IF(IHLF)201,201,217
  217 IF(N-7)201,218,218
  218 IF(ISTEP-4)201,219,219
  219 IMOD=ISTEP/2
      IF(ISTEP-IMOD-IMOD)201,220,201
  220 H=H+H
      IHLF=IHLF-1
      ISTEP=0
      DO 221 I=1,NDIM
      AUX(N-1,I)=AUX(N-2,I)
      AUX(N-2,I)=AUX(N-4,I)
      AUX(N-3,I)=AUX(N-6,I)
      AUX(N+6,I)=AUX(N+5,I)
      AUX(N+5,I)=AUX(N+3,I)
      AUX(N+4,I)=AUX(N+1,I)
      DELT=AUX(N+6,I)+AUX(N+5,I)
      DELT=DELT+DELT+DELT
 2210 AUX(16,I)=8.962963*(Y(I)-AUX(N-3,I))-3.361111*H*(DERY(I)+DELT
     1+AUX(N+4,I))
      GOTO 201
CCC
CCC   H MUST BE HALVED
  222 IHLF=IHLF+1
      IF(IHLF-10)223,223,210
  223 H=.5*H
      ISTEP=0
      DO 224 I=1,NDIM
     0Y(I)=.00390625*(80.*AUX(N-1,I)+135.*AUX(N-2,I)+40.*AUX(N-3,I)+
     1AUX(N-4,I))-.1171875*(AUX(N+5,I)-6.*AUX(N+5,I)-AUX(N+4,I))*H
```

```
     0AUX(N-4,I)=.00390625*(12.*AUX(N-1,I)+135.*AUX(N-2,I)+
     1108.*AUX(N-3,I)+AUX(N-4,I))-.0234375*(AUX(N+6,I)+18.*AUX(N+5,I)-
     29.*AUX(N+4,I))*H
      AUX(N-3,I)=AUX(N-2,I)
224   AUX(N+4,I)=AUX(N+5,I)
      X=X-H
      DELT=X-(H+H)
      CALL FCT(DELT,Y,DERY)
      DO 225 I=1,NDIM
      AUX(N-2,I)=Y(I)
      AUX(N+5,I)=DERY(I)
225   Y(I)=AUX(N-4,I)
      DELT=DELT-(H+H)
      CALL FCT(DELT,Y,DERY)
      DO 226 I=1,NDIM
      DELT=AUX(N+5,I)+AUX(N+4,I)
      DELT=DELT+DELT+DELT
     0AUX(16,I)=8.962963*(AUX(N-1,I)-Y(I))-3.361111*H*(AUX(N+6,I)+DELT
     1+DERY(I))
226   AUX(N+3,I)=DERY(I)
      GOTO 206
      END

STOP
```