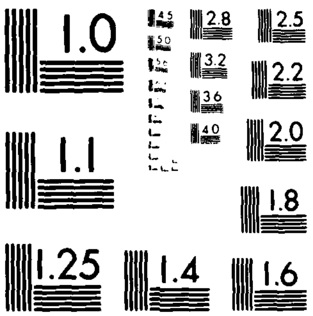
AD-A094 240 INSTITUTE FOR DEFENSE ANALYSES ARLINGTON VA SCIENCE A--ETC F/G 8/3
A NUMERICAL TECHNIQUE FOR THE CALCULATION OF DISPERSION RELATIO--ETC(U)
OCT 80 I W KAY, P J DRAPER, W WASYLKIWSKYJ MDA903-79-C-0202
UNCLASSIFIED IDA-P-1491 IDA/HQ-80-22393 NL

1 of 1
AD A
094240

END
DATE
FILMED
2-81
DTIC



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU

IDA PAPER P-1491

A NUMERICAL TECHNIQUE FOR THE CALCULATION OF DISPERSION RELATIONS AND MODE FUNCTIONS FOR UPPER OCEAN INTERNAL WAVES

Irvin W. Kay
Patricia J. Draper
Wasyl Wasylkiwskyj

October 1980

Prepared for
Defense Advanced Research Projects Agency

IDA

INSTITUTE FOR DEFENSE ANALYSES
SCIENCE AND TECHNOLOGY DIVISION

The work reported in this document was conducted under contract MDA 903 79 C 0202 for the Department of Defense. The publication of this IDA Paper does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that agency.

Approved for public release; distribution unlimited.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A094	3. RECIPIENT'S CATALOG NUMBER 240
4. TITLE (and Subtitle) A Numerical Technique for the Calculation of Dispersion Relations and Mode Functions for Upper Ocean Internal Waves		5. TYPE OF REPORT & PERIOD COVERED FINAL 6. PERFORMING ORG. REPORT NUMBER IDA Paper P-1491
7. AUTHOR(s) Irvin W. Kay, Patricia J. Draper, Wasyl Wasylkiwskyj		8. CONTRACT OR GRANT NUMBER(s) MDA 903 79 C 0202
9. PERFORMING ORGANIZATION NAME AND ADDRESS Institute for Defense Analyses 400 Army-Navy Drive Arlington, Virginia 22202		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS DARPA Assignment A-49
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209		12. REPORT DATE October 1980 13. NUMBER OF PAGES 76
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION DOWNGRADING SCHEDULE ——

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

None

18. SUPPLEMENTARY NOTES

N/A

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

hydrodynamics, internal waves, Väisälä frequency profile, numerical methods, computer program, algorithm, dispersion relation, eigenfunction, infinitely deep ocean

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper is concerned with calculating the internal wave eigenfunctions and dispersion relations for an infinitely deep ocean with an arbitrary Väisälä frequency profile. The method involves numerically integrating the differential equation from a depth where the profile is essentially an exponential function, and therefore where the eigenfunctions are known explicitly, to the surface where each eigenfunction must vanish. The equation that is implied by the surface boundary condition determines

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. (continued)

the dispersion relation, which is obtained numerically by solving the equation by means of the method of *regula falsi*. A computer program that performs the necessary calculations is described and listed.

IDA PAPER P-1491

A NUMERICAL TECHNIQUE FOR THE CALCULATION OF DISPERSION RELATIONS AND MODE FUNCTIONS FOR UPPER OCEAN INTERNAL WAVES

Irvin W. Kay
Patricia J. Draper
Wasyl Wasylkiwskyj

October 1980

IDA

INSTITUTE FOR DEFENSE ANALYSES
SCIENCE AND TECHNOLOGY DIVISON
400 Army-Navy Drive, Arlington, Virginia 22202
Contract MDA 903 79 C 0202
DARPA Assignment A-49

CONTENTS

I. INTRODUCTION

Generally, the numerical approach to finding internal wave eigenfunctions involves numerical integration of the differential equation over the entire ocean depth. For a deep ocean, particularly when the horizontal wavelengths of interest are much shorter than the ocean depth, numerical integration can be confined to the first few hundred meters of the ocean. This simplification arises from the fact that for a wide range of possible profiles the variation in depth dependence is confined to the upper few hundred meters below the surface, after which the profile may be approximated by a decaying exponential. Moreover, for horizontal wavelengths much shorter than the ocean depth the exponentially decaying profile model may be extended to infinity, i.e., the ocean may be assumed to have infinite depth. Since the solutions of the differential equation for a continuously decreasing exponential profile are known in closed form (they are Bessel functions of the first kind), numerical integration of the differential equation needs to be carried out only over the upper few hundred meters of depth, within which an essentially arbitrary Välsälä frequency profile may be prescribed, as derived from experimental data.

The numerical procedure to be presented here will provide any mode function and dispersion relation over any wavelength range for an arbitrary Väisälä frequency profile, as long as it decays exponentially at great depths. The practical limitation, of course, is the amount of computer time one is willing to expend for results covering an appropriate dynamic range. For ordinary ranges of physical interest, the required computing time appears to be of the order of minutes for a dispersion curve and seconds for a mode function.

Dr. P. A. Selwyn, who was kind enough to review the first draft of this paper, has called attention to the fact that there already exists a computer program (Refs. 1 and 2) that performs calculations similar to those undertaken here, but for the case in which the ocean depth is finite. One would expect that it might have been possible to modify the earlier program so as to adapt it to the infinite depth case as well.

Although, in a sense, the question of that possibility was moot by the time this information had been disclosed, nevertheless, the option still exists for a potential user who, if convinced that the earlier program has sufficient advantages over the present one, might be willing to risk whatever numerical complications may be involved in such a task. Therefore, it seems proper to discuss briefly some of the major similarities and differences between the two programs. Unfortunately, a direct comparison of their computational accuracies and speeds is not feasible at present.*

Both programs rely on the same general method, which is to start with the appropriate boundary condition at the deep end of the Väisälä frequency profile and numerically integrate the differential equation over depth for selected values of the wave number and phase velocity. These parameters are varied according to some iteration procedure until the value of the solution of the differential equation at the ocean surface is sufficiently close to zero.

The earlier program solves for the phase velocity in terms of the wave number. Here the converse is true.

*If the adaptation of the earlier program were based on the same idea used here to account for the boundary condition at infinity, i.e., matching with the Bessel function solution for an exponential profile, the running time of the program would be increased considerably.

The earlier program uses a fourth-order Runge-Kutta method for solving the differential equation, so that the error is of fifth order in the incremental step size. Here the Numerov-Manning-Millman method is used, and the error is of sixth order in the incremental step size. This gain in efficiency occurs because the present method takes advantage of the fact that the differential equation does not contain a first derivative term.

The earlier program uses a Newton-Rapheson iteration scheme to find the eigenvalue, whereas here the *regula falsi* scheme is used. Generally, the Newton-Rapheson iteration converges faster but may fail to converge if the trial value is not sufficiently close to the true value. However, once started properly the *regula falsi* iteration is guaranteed to converge.*

Other differences between the two programs primarily have to do with how they handle the problem of mode jumping, which occurs in the calculation when dispersion curves for successive modes approach each other too closely. The details are rather involved and it is difficult to assess the relative merits of the two approaches without considerable further study. However, it may be worth noting that the nature of the *regula falsi* method permits the use of certain eigenvalue properties, derived from the classical Sturm-Liouville theory, that aid in resolving the mode-jumping difficulties encountered by the program presented here.

The mathematical statement of the problem to be solved and some properties of the eigenfunctions and dispersion relations needed in the subsequent development are summarized in Chapter II. In Chapter III the general framework for the numerical

*There are cases in which the convergence is extremely slow, however. When this happens it has been found expedient to switch to internal halving, which is less efficient, in general, but which converges at a predictable rate.

solution of the dispersion curves is presented. The detailed description of the algorithm for the numerical solution of the differential equation and the associated eigenvalues is given in Chapter IV. The computer implementation of this algorithm INTMODE is described in Chapters V and VI. A more detailed description is given in the Appendix.

Some examples of numerical results obtained with INTMODE are presented in Chapter VII.

II. DESCRIPTION OF THE PROBLEM AND SOME GENERAL PROPERTIES OF INTEREST

Interest here is confined to internal waves with horizontal wavelengths of at most a few kilometers so that the effects of the inertial frequency do not enter. Accordingly, the mode functions $\psi_m(y)$ are determined by the differential equation

$$\frac{d^2\psi_m}{dy^2} + K^2\left[\frac{N^2(y)}{\Omega_m{}^2} -1\right]\psi_m(y) = 0, \tag{1}$$

along with the boundary conditions

$$\psi_m(0) = 0, \lim_{y\to+\infty} \psi_m(y) = 0, \tag{2}$$

where the depth y is measured from the surface y = 0. In (1) K is a given wave number, Ω_m is the mode frequency, and N(y) is the Väisälä frequency profile associated with a vertical thermocline in the ocean.

For a given value of K the differential equation (1) subject to the boundary conditions (2) determines uniquely the infinite set of mode functions $\psi_m(y)$. For each mode a dispersion relation is thereby determined; that is, each Ω_m is a well defined function of K,

$$\Omega_m = \Omega_m(K), \; m = 1,2, \ldots, \tag{3}$$

over the interval

$$0 < K < \infty.$$

In general, the $\Omega_m(K)$ are uniformly bounded, monotonic increasing functions that vanish at $K = 0$. As K becomes arbitrarily large each Ω_m asymptotically approaches a single positive constant Ω_0 equal to the maximum value of $N(y)$, the Väisälä frequency. As K approaches zero the slope of each of the dispersion relation curves defined by (3) approaches a different finite positive value $1/\mu_m$; i.e., $\lim\limits_{K\to o} \frac{d\Omega m}{dK} = \lim\limits_{K\to o} \frac{\Omega m}{K} = 1/\mu_m$.

The dispersion relations (3) can be expressed parametrically in terms of a quantity μ, which approaches μ_m as K approaches zero; i.e.,

$$\begin{aligned} K_m &= K_m(\mu) \\ \Omega_m &= \frac{K_m(\mu)}{\mu} , \end{aligned} \tag{4}$$

where the set of functions $K_m(\mu)$ are determined by the differential equation

$$\frac{d^2\psi_m}{dy^2} + \mu^2 N^2(y)\psi_m = K_m^2\psi_m, \tag{5}$$

subject to the boundary conditions (2). From this point of view, for each value of μ that is chosen above the minimum value needed for the existence of the particular modes being considered, the boundary value problem determines a set of positive real eigenvalues* $K_m{}^2(\mu)$, the square root of which provides the $K_m(\mu)$.

According to the general theory of eigenfunctions for the linear second order differential equation (Ref. 4), if N(y) is piecewise continuous, bounded and approaches zero as y becomes infinite, for fixed μ there are a finite set of real positive eigenvalues $K_m{}^2$.* As μ increases, the number of these eigenvalues also increases, but for μ smaller than some critical value there

*Actually, the convention is to regard the quantities $- K_m{}^2$ as the eigenvalues, which results in their being characterized as negative real.

may be no eigenvalues at all. It is also known from the general theory that each K_m is an increasing function of μ (cf. Ref. 5, p. 357).

It is assumed that the Väisälä frequency profile $N(y)$ decays exponentially at great depths. This behavior may be characterized more conveniently by actually requiring that after some depth y_0, $N(y)$ becomes an exponential function; i.e.,

$$N(y) = \hat{N}_0 e^{-\frac{y}{b}}, \; y \geq y_0. \tag{6}$$

As observed in Ref. 1 the solution to the boundary value problem associated with (1) and (2) for an exponential profile of the form (6) is proportional to a function $\phi(y)$ given by

$$\phi(y) = J_{Kb} \, (\mu b \hat{N}_0 e^{-\frac{y}{b}}). \tag{7}$$

Thus, the boundary conditions (2) can be restated as

$$\psi(0) = 0, \; \psi(y) = A \, J_{Kb}(\mu b \hat{N}_0 e^{-\frac{y}{b}}), \; y \geq y_0, \tag{8}$$

where A is a constant that can be chosen arbitrarily or to satisfy some normalization condition.

According to Ref. 3, a W.K.B. solution of (1) can be used to determine the approximate m^{th} mode dispersion relation; i.e., within the W.K.B. approximation the dispersion relation is determined by the equation

$$\int_{N>\Omega} p dy = \pi(m - \tfrac{1}{4}), \tag{9}$$

where

$$p(y) = \frac{K}{\Omega} [N^2(y) - \Omega^2]^{1/2} \tag{10}$$

$$= \sqrt{\mu^2 N^2(y) - K^2} \, .$$

III. THE NUMERICAL PROCEDURE TO DETERMINE DISPERSION RELATIONS

In this chapter the principles underlying the numerical procedure to be used for calculating the dispersion relations will be outlined. The details of the procedure, itself, will be presented further on.

Since K is an increasing function of μ, it follows from (9) and (10), by considering the limit as K approaches zero, that the minimum permissible value μ_m for the m^{th} mode is given approximately by

$$\mu_m \sim \frac{\left(m - \frac{1}{4}\right)\pi}{\int_0^\infty N(y)dy} . \tag{11}$$

Since (11) is only an approximate formula, in order to guarantee that a solution K_m exists for the eigenvalue problem, values of μ chosen to calculate the dispersion relation for a given mode should be somewhat larger; e.g., it would be prudent to confine the choice of μ to values such that

$$\mu \geq \frac{\frac{3\pi}{2} + (m-1)\pi}{\int_0^\infty N(y)dy} . \tag{12}$$

The smallest value of μ given by (12) ought to be large enough to guarantee the existence of a real K_m but, ideally, small enough to exclude the existence of any higher mode eigenvalue K_n, $n > m$. It will be found that this is generally true but that there are some noteworthy exceptions.

For any choice of positive K and μ it is possible to find a solution of the differential equation (5), subject to the second condition in (8) which determines the necessary initial values to be imposed for that purpose at the point y_0. The differential equation can be integrated numerically from y_0 down to zero, where a value for $\psi(0)$ will thus be acquired. If the maximum magnitude of the solution over the interval $0 \leq y \leq y_0$ is $|\psi_{max}|$ then a function $\Psi(\mu,K)$, determined by this process, may be defined by

$$\Psi(\mu,K) = \frac{\psi(0)}{|\psi_{max}|} . \tag{13}$$

As defined, $\Psi(\mu,K)$ is a function of μ and K alone; i.e., it is independent of the normalization constant A.*

Because there are cases in which (12) does not lead to a satisfactory initial value for μ (some modes are skipped) a slower but safer procedure than relying on the W.K.B. approximation has been adopted here. The differential equation (5) is solved numerically, subject to the second condition in (8), with K set equal to zero and μ set equal to a sequence of values $\frac{n}{\int_0^\infty N(y)dy}$. For each value of μ in the sequence the sign of $\psi(o)$ is observed. When a change in the sign occurs the corresponding value of μ is used as a trial value, and it is assumed that zero bounds K from below. If it is found that modes are still skipped, the μ increment is decreased and the procedure repeated.

If K^2 happens to be an eigenvalue, then $\Psi(\mu,K)$ vanishes. Thus, the problem of calculating the dispersion relations is

*Because it automatically relates error to a specified dynamic range, the normalization (13) is needed for stability of the numerical process used to solve (14).

equivalent to finding the real (positive) roots $K(\mu)$ of the transcendental equation

$$\Psi(\mu,K) = 0 \tag{14}$$

as the parameter μ varies. For sufficiently small μ there are no roots, while for μ in the interval

$$\mu_1 \leq \mu < \mu_2$$

there is just one root, and for μ in the interval

$$\mu_2 \leq \mu < \mu_3$$

there are exactly two roots, etc.

To calculate the roots of (14), interval halving or, for more rapid convergence, the *regula falsi* (Ref. 6) method can be used. The numerical procedure given here does, in fact, rely upon the *regula falsi* method to solve (14), although the technique of interval halving is used in certain circumstances, to be described, in order to reduce computing time.

It is necessary to begin with two trial values for $K(\mu)$, $\tilde{K}_1$ and $\tilde{K}_2 > \tilde{K}_1$, such that $\Psi(\mu,\tilde{K}_1)$ and $\Psi(\mu,\tilde{K}_2)$ differ in sign, to guarantee that the root $K(\mu)$ lies between $\tilde{K}_1$ and $\tilde{K}_2$; i.e.,

$$\tilde{K}_1 \leq K(\mu) \leq \tilde{K}_2 \ .$$

Since $K(\mu)$ must be positive, initially, the trial value $\tilde{K}_1$ can be zero. For the initial upper bound $\tilde{K}_2$, a quantity defined by

$$\tilde{K}_2 = N_{max},* \tag{15}$$

where N_{max} is the largest value attained by $N(y)$, will suffice.

*The fact that $\tilde{K}_2$ as defined by (15) is an upper bound can be seen by multiplying (5) by ψ_m and integrating from 0 to ∞. Integration of the derivative term by parts shows that that term is negative.

For the higher modes it will again be necessary to begin with $\tilde{K}_1 = 0$. However, $\tilde{K}_2$ may be set equal to the previously calculated value of K for the mode one step down at the same value of μ. That is, since it is known that

$$K_{m-1}(\mu) > K_m(\mu),$$

in calculating $K_m(\mu)$ a value for $\tilde{K}_2$ given by

$$\tilde{K}_2 \leq K_{m-1}(\mu) \qquad (16)$$

can be used.

As indicated, the trial value $\tilde{K}_2$ should be slightly less than $K_{m-1}(\mu)$ to avoid accidentally falling back onto the m-1st mode dispersion curve because of normal errors to be expected in the calculation. A test should be included here to guarantee that the choice of $\tilde{K}_2$ is not too much less than $K_{m-1}(\mu)$: the function $\Psi(\mu,K)$ must change sign in going from $\tilde{K}_1$ to $\tilde{K}_2$.*

In order to obtain a trial value satisfying (16), it is necessary to have an estimate of $K_{m-1}(\mu)$ that is known to be too small. Since it is generally not the case that K_{m-1} would have been calculated previously for exactly the value of μ now encountered in the mode m calculations, the estimate of $K_{m-1}(\mu)$ must be determined by interpolation, e.g., between values $K_{m-1}(\mu_n)$ and $K_{m-1}(\mu_n + \Delta\mu)$, where

$$\mu_n < \mu < \mu_n + \Delta\mu.$$

However, if the K versus μ curves are concave upward such an interpolation will produce an estimate that is too large; hence, the desired sign change in $\Psi(\mu,K)$ would not occur. On the other

*While the *regula falsi* method may still work even if this requirement is not met, it is not actually guaranteed to converge unless the sign change rule is imposed.

hand, a cruder, one-sided, interpolation that does guarantee the sign change can be used. That is, instead of interpolating between $K_{m-1}(\mu_n)$ and $K_{m-1}(\mu_n + \Delta\mu)$ the trial value estimate becomes

$$\tilde{K}_2 = \frac{K_{m-1}(\mu_n)}{\mu_n}\,\mu. \tag{17}$$

From the fact that $\Omega_{m-1} = K_{m-1}(\mu)/\mu$ is a monotonic increasing function* it can be readily inferred that $\tilde{K}_2$ defined by (17) will, in fact, be smaller than $K_{m-1}(\mu)$.

Once K has been calculated for the initial choice of μ for a given mode, the value of μ is increased by adding a small increment $\Delta\mu$. A corresponding increment for the lower bound trial value $\tilde{K}_1$ can be obtained from an interpolation analogous to (17).

That this can be done so that $\tilde{K}_1$ continues to be a lower bound can be seen as follows. By definition,

$$K = \mu\Omega.$$

Therefore,

$$\frac{dK}{d\mu} = \Omega + \mu\,\frac{d\Omega}{d\mu}\,. \tag{18}$$

Although Ω increases monotonically with μ, it is uniformly bounded by the maximum Väisälä frequency; hence, the second term on the right of (18) approaches zero as μ becomes arbitrarily large. This is evident in view of the fact that, since the derivative of log μ is $\frac{1}{\mu}$, $\frac{d\Omega}{d\mu}$ approaches zero faster than $\frac{1}{\mu}$. Then, for large enough μ, according to (18),

$$\Delta\, K \sim \Omega\, \Delta\, \mu. \tag{19}$$

*If it were not, a case of anomolous dispersion would be implied since, as already observed, K is an increasing function of μ.

Moreover, because Ω is a monotonic function of μ, the estimate of ΔK given by (19) is always too small.

Thus, the new $\tilde{K}_1$ can be chosen in accordance with (19); i.e.,

$$\tilde{K}_1 = K_\mu + \Omega_\mu \Delta\mu = K_\mu + \Delta\mu \frac{K_\mu}{\mu}, \qquad (20)$$

where K_μ is the previously calculated value of K corresponding to the value of μ before the increment $\Delta\mu$ is added. For modes higher than the first ($m = 1$) the use of (17) to obtain the upper bound $\tilde{K}_2$ continues each time μ is incremented, while the lower bound $\tilde{K}_1$ is obtained from (20). For the first mode, however, (15) is the only estimate immediately available for the upper bound $\tilde{K}_2$ as μ is incremented, although (20) can still be used to estimate the lower bound $\tilde{K}_1$.

IV. DETAILS OF THE NUMERICAL PROCEDURE

A. SOLUTION OF THE DIFFERENTIAL EQUATION

The Numerov-Manning-Millman method (Ref. 6, pp. 204-205) is particularly convenient for solving (5) numerically for given values of μ and K. The method requires two starting values; for a step size h, $\psi(y_o)$ and $\psi(y_o+h)$ must be furnished initially. Then the differential equation can be integrated by means of a single recursion relation that involves only ψ and its second derivative, which is obtained from ψ and the relationship supplied by the differential equation, itself.

The starting values of ψ are obtained by recognizing that at y_o and y_o+h the profile N(y) is an exponential function of the form (6). Thus, in accordance with (7), at these points $\psi(y)$ can be set equal to $J_{Kb}(\mu b\hat{N}_o e^{-\frac{y}{b}})$.

When N(y) is prescribed numerically over an interval $(0,y_o)$ the resolution of N(y) implies a limit on how small the step size h may be taken. Conversely, a natural limitation on how large h may be is the requirement that it be small compared to the minimum wavelength λ to be considered. Since the wavelength is given by

$$\lambda = \frac{2\pi}{K} ,$$

this means that the size of h is governed by the largest value to be considered for the wave number K.

B. SOLUTION OF THE EIGENVALUE EQUATION

The eigenvalues K that determine the dispersion relation for each mode are found by solving (14) over an appropriate range of values for μ. For this purpose the *regula falsi* method (Ref. 6, pp. 4-5) seems most effective.

In some cases, convergence of the *regula falsi* method is too slow. Therefore, if twenty iterations occur without satisfying the prescribed error criterion the computer program switches to interval halving with an error criterion applied to K rather than ψ.

C. SELECTING INCREMENTS OF μ

In accordance with (12), the increment $\delta\mu$ used to obtain the starting value of μ in going from the dispersion relation for one mode to that for the next is normally given by adding increments

$$\delta\mu = \frac{1}{\int_0^\infty N(y)dy} \tag{21}$$

until $\psi(\mu,o)$ changes sign. At the start of the mode, $\tilde{K}_1$ is then set equal to zero.

If the increment $\Delta\mu$ along a single mode is too large, a jump to the next mode may occur. This can be guarded against by calculating $\psi(\mu + \Delta\mu, K_\mu)$ which in that case would have a different sign than $\psi(\mu,\tilde{K}_1)$, where $\tilde{K}_1$ is a lower bound used in calculating K_μ.

Evidently, as a practical matter $\Delta\mu$ must not be too large. It is also true, however, that $\Delta\mu$ must not be too small. While, theoretically, trial values are chosen so as to guarantee the necessary sign change in $\Psi(\mu,K)$ for the *regula falsi* method, in practice it turns out that when $\Delta\mu$ is sufficiently small the sign change may, nevertheless, fail to occur. This is due to the residual calculation error in the K that corresponds to the value of μ before it is incremented. This error is sufficient in some cases to overcome the theoretical inequality relied upon in the derivation of the rule for selecting $\tilde{K}_2$.

A compromise rule for selecting the μ increment is to let $\Delta\mu$ be about $\frac{1}{10}\,\delta\mu$. That is, a reasonable choice that seems adequate in practice is given by

$$\Delta\mu = \frac{1}{2\int_0^\infty N(y)dy} . \qquad (22)$$

D. ESTIMATING THE ERROR IN K

The test used to determine when to stop the *regula falsi* iterations in calculating K is the condition

$$|\Psi(\mu,K)|<\varepsilon. \qquad (23)$$

The value chosen for this purpose in current applications is 10^{-7}, which is intended to provide at least a 60 dB dynamic range for the corresponding mode functions.

Therefore, the error in K is not given directly; however, it can be estimated by linear extrapolation. If Ψ_n is the value of $\Psi(\mu,K)$ that just meets the test (23) and Ψ_{n-1} is the value of $\Psi(\mu,K)$ in the iteration just before that one, then the quantity

$$\frac{\Delta K}{\Delta\Psi} = \frac{K_n - K_{n-1}}{\Psi_n - \Psi_{n-1}} \qquad (24)$$

where K_n and K_{n-1} are the corresponding estimates of K in the two iterations, is approximately the rate of change of K with respect to a change in $\Psi(\mu,K)$. Then the error ε_K in K corresponding to ε will be given approximately by

$$\varepsilon_K = \frac{\Delta K}{\Delta \Psi}\,\varepsilon. \tag{25}$$

The error estimate ε_K can be used to prevent the anomaly mentioned earlier, that too small a choice of $\Delta\mu$ can result in a failure to obtain a sign change in $\Psi(\mu,K)$ using the trial value $\tilde{K}_1$ obtained by means of (20). The idea is to make sure that the error in the calculated value of K is always negative, i.e., that the calculated value of K is too small. This can be done by subtracting ε_K after the iterations for K are completed.

V. COMPUTER REALIZATION OF THE ALGORITHM

The computer program DISPER was designed to calculate the K = K(μ) relationship using the numerical techniques described earlier in this paper. This program will write the (K,μ) pairs as calculated along each mode to disk or tape and will plot a graph of the (K,Ω) curves, referred to as dispersion curves.

A. INPUTS

The inputs to the program are of two types: (1) parameters read in under a NAMELIST option, and (2) data points read in from punched data cards.

1. NAMELIST/PARAM/XO, B, STOPK, STOPMU, ND, EPS

XO
Real
The X-coordinate of the last data point of the numerically defined function N(X). For the STD data XO = 220. m.

B
Real
Decay constant
For the STD data B = 1300.

STOPK
Real
The maximum K value for which the user wants dispersion curves.

STOPMU
Integer
The number of dispersion curves to be calculated.

ND
Integer
The number of data points +1 to be read into the array N.

EPS	Real The error criteria imposed on the numerical solutions to the differential equations. For the STD data EPS = 1.E-7.

2. Data Cards

ITITLE	Integer Ten character title of the N(x) values.
N	Real array Dimensioned 500, read in on punched cards under the format (8F10.5). N contains the equispaced data points that numerically define the function N(X). The points are spaced a distance of XO/(ND-2) apart.

B. OUTPUTS

1. Printout

a. The parameters defined by the NAMELIST option are listed at the end of the program for verification purposes.

b. The value of $\int N(y)dy$ is printed next, followed by the values of N.

c. At the end of the calculations for each mode the number of (K,μ) pairs, the maximum estimated error for K, and the complete list of (K,μ) pairs for that mode are printed.

d. Occasionally the error in K cannot be estimated. When that occurs a message indicating this fact and the current values of K and μ are printed.

2. Disk or Tape

The values of the NAMELIST/PARAM/B, XO, STOPK, EPS, ND, STOPMU are written on TAPE2 under the format (4E22.7, 2I5//). Next the values of N are written to TAPE2 under the format (8E10.5).

At the end of each mode the number of (K,μ) pairs calculated is written to TAPE2, format (//I5). The (K,μ) pairs are then written to TAPE2, format (2E22.7).

TAPE2 may be defined as a permanent file by using a catalog control card, or it may be defined as a magnetic tape by using a label control card.

3. PLOT

A 10" by 10" graph consisting of the STOPMU different curves of the (K,Ω) pairs is plotted at the end of the program.

C. EXTERNAL REFERENCES

DISPER references several external subroutines that must be provided by the user through control cards that attach the appropriate permanent files.

The necessary routines are listed below under the name of the permanent file on which they reside.

1. INTMODE

INITIAL — Reads the data values of N(X), calculates the integral

$$\int_0^\infty N(x)\,dx$$

GUESS — Calculates "best" estimate of K given μ.

DIFF — Numerically solves the differential equation using the Numerov-Manning-Millman method. Called by GUESS.

XMUØ — Function to find MUØ for each mode.

OUTPUTK — Writes the (K,μ) pairs to TAPE2.

PLOTER — Sets up the calls to the CalComp plotting routines.

PSCALE Scales the axes to the calculated data. Called by PLOTER.

ERRPRO Processes detected errors through a call to ABRTJOB. Is called by all of the routines on this permanent file.

2. IDALIB

PLOTS
PLOT
LINE
DAXIS
SYMBOL
NUMBER CalComp plotting routines called by PLOTER.

ABRTJOB Error processor that generates TRACEBACK, prints error messages, and terminates the job. Called by ERRPRO.

3. BESSEL

JBESS
JAIRY
GAMLN Routines to calculate the BESSEL functions. Acquired from the Argonne National Laboratory.

D. ERROR MESSAGES

We have attempted to anticipate some of the errors a user might encounter when using DISPER under very general conditions. If one of these errors is detected by the program the error processor ERRPRO is called. ERRPRO does three things, (1) prints a brief message describing the error, (2) indicates in which routine the error occurred, and (3) terminates the job without a dump.

A table of error messages generated by DISPER and possible corrective actions that might resolve the problem is presented below.

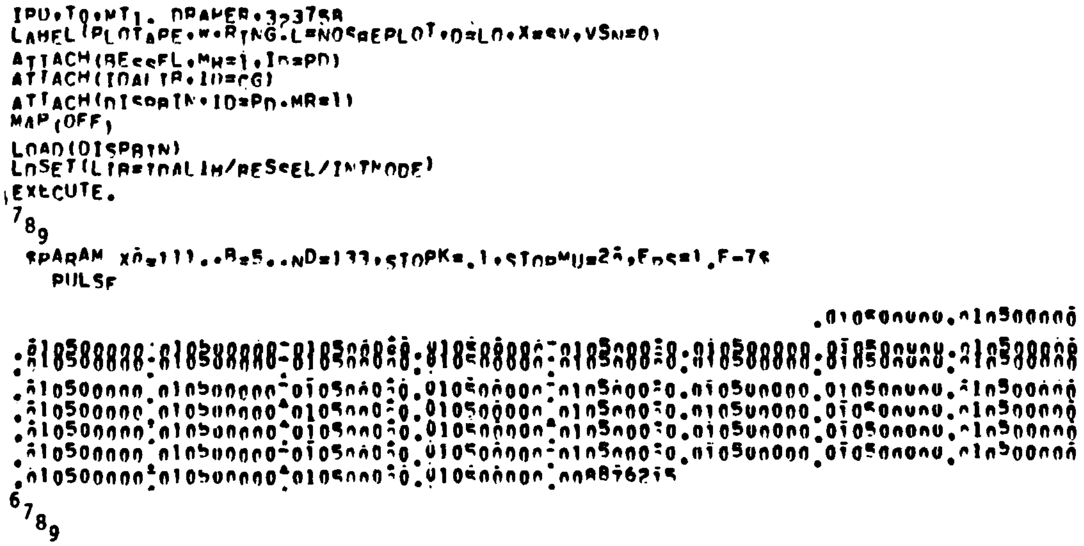
TABLE 1. ERROR MESSAGES AND POSSIBLE CORRECTIVE ACTIONS

Message	Significance	Action	Issued By
END-OF-FILE,UNITS	ND-1 is greater than the number of data points provided for N.	Decrease ND or provide more data.	INITIAL
INDEFINITE OPERAND 0/0	Solution to differential equation is inconsistent.	Look for coding errors in the routine DIFF.	DIFF
DIVISION BY ZERO	Algorithm for solving the differential equation has broken down.	Data points may be too far apart. Introduce more data, perhaps through interpolation.	DIFF
FIRST MODE CANNOT BE CONSTRUCTED	Cannot find MU0 for this mode.	Reexamine data.	XMU0
TOO MANY POINTS	More than 500 (K,μ) pairs are need to construct this mode.	Decrease STOPK, or revise program by redimensioning SAVEK, SAVEMU, TEMPK, and TEMPMU.	DISPER
KI and K0 ON SAME SIDE OF CURVE	Estimates for K are not upper and lower bounds.		GUESS
	(1) EPS too severe. (2) Extrapolating initial estimate of K1 beyond K values of the preceding mode.	(1) Reduce EPS. (2) Reduce STOPMU or increase STOPK.	

E. DETAILS OF JOB EXECUTION

The following is a sample card deck for executing DISPER.

1.



```
IPU,T0,MT1. DRAPER,323758
LABEL(PLOTAPE,W,RING,L=NOSREPLOT,D=LO,X=SV,VSN=0)
ATTACH(RESSFL,MR=1,ID=PD)
ATTACH(IDALIB,ID=CG)
ATTACH(DISPBIN,ID=PD,MR=1)
MAP(OFF)
LOAD(DISPBIN)
LDSET(LIB=IDALIB/RESSEL/INTMODE)
EXECUTE.
7
 8
  9
 $PARAM X0=111.,B=5.,ND=133,STOPK=.1,STOPMU=20,EPS=1.E-7$
   PULSE
                                                                        .01050000.01050000
.01050000.01050000.01050000.0.01050000.01050000.0.01050000.01050000.01050000
.01050000.01050000.01050000.0.01050000.01050000.0.01050000.01050000.01050000
.01050000.01050000.01050000.0.01050000.01050000.0.01050000.01050000.01050000
.01050000.01050000.01050000.0.01050000.01050000.0.01050000.01050000.01050000
.01050000.01050000.01050000.0.01050000.01050000.0.01050000.01050000.01050000
.01050000.01050000.01050000.0.01050000.00087625
6
 7
  8
   9
```

This will produce a listing of the (K,μ) pairs for each mode and a plot of the dispersion curves. It will not create a tape or permanent file of the (K,μ) pairs.

If the user wishes a permanent record of the (K,μ) pairs additional control cards must be included.

For a permanent file the deck would contain two additional cards. The sample deck below is an example.

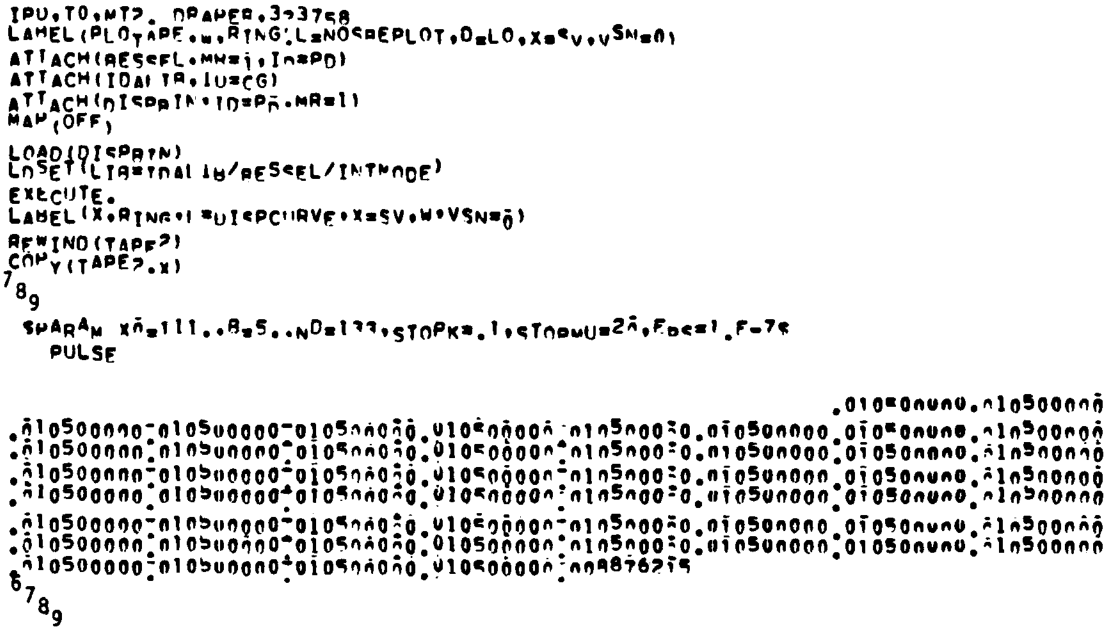
2.

```
IPU,T0,MT1. DRAPER,3,3758
REQUEST(TAPE2,*PF)
LABEL(PLOTAPE,W,RING,L=NOSREPLOT,D=LO,X=SV,VSN=0)
ATTACH(RESSFL,MR=1,ID=PD)
ATTACH(IDALTB,ID=CG)
ATTACH(DISPRIN,ID=PD,MR=1)
MAP(OFF)
LOAD(DISPRIN)
LDSET(LIB=IMALIB/RESSEL/INTMODE)
EXECUTE.
CATALOG(TAPE2,DISPCURVE,ID=PD)
7
 8
  9
 $PARAM X0=111.,B=5.,ND=133,STOPK=.1,STOPMU=20,EPS=1.E-7$
   PULSE

                                                                     .01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.00876215
6
 7
  8
   9
```

If the (K,μ) pairs are to be written to a magnetic tape the job stream might look like the example shown below:

3.



```
IPU,T0,MT2. DRAPER,3?37?8
LAHEL(PLOTAPE,W,RING,L=NOSREPLOT,D=LO,X=SV,VSN=0)
ATTACH(RESSEL,MH=1,ID=PD)
ATTACH(IDAL IB,ID=CG)
ATTACH(DISPRIN,ID=PD,MR=1)
MAP(OFF)
LOAD(DISPRIN)
LDSET(LIB=IDAL IB/RESSEL/INTMODE)
EXECUTE.
LABEL(X,RING,L=DISPCURVE,X=SV,W,VSN=0)
REWIND(TAPE2)
COPY(TAPE2,X)
789
 $PARAM X0=111.,B=5.,ND=133,STOPK=.1,STOPMU=20,FDC=1.E-7$
   PULSE

                                                                      .0105000000.0105000000
.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000
.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000
.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000
.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000
.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000
.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000.0105000000
.0105000000.0105000000.0105000000.0105000000.0098762?5
6789
```

Note the change in the job card as well as the additional control cards.

Should both a magnetic tape and a permanent file be desired the job stream would look like the next example.

```
IPU,T0,MT2. DRAPER,323758
REQUEST(TAPE2,*PF)
LABEL(PLOTAPE,W,RING,L=NOREPLOT,D=LO,X=SV,VSN=0)
ATTACH(RESSEL,MR=1,ID=PD)
ATTACH(IDALTR,ID=CG)
ATTACH(DISPRIN,ID=PD,MR=1)
MAP(OFF)
LOAD(DISPRIN)
LDSET(LIB=IDALIB/RESSEL/INTMODE)
EXECUTE.
CATALOG(TAPE2,DISPCURVE,ID=PD)
LABEL(X,RING,L=DISPCURVE,X=SV,W,VSN=0)
REWIND(TAPE2)
COPY(TAPE2,X)
789
 $PARAM X0=111.,R=5.,ND=133,STOPK=.1,STOPMU=20,EPS=1.E-7$
   PULSE


                                                                                   .01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000

.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000

.01050000.01050000.01050000.01050000.01050000.01050000.01050000.01050000
.01050000.01050000.01050000.01050000.00987625
6789
```

VI. PROGRAM MODE

The computer program MODE was designed to calculate and plot the normalized mode functions $\Psi(\mu,K)$. Given (1) the dispersion curves created by DISPER, (2) a set of consecutive mode numbers, and (3) a value for K, this program will use the dispersion curves and linear interpolation to find the correponding μ values. It will then calculate and plot the mode function for each mode number.

A. INPUTS

The inputs to this program are of two types: (1) the outputs of DISPER, and (2) a NAMELIST option.

1. TAPE5

TAPE 5 is defined to be the disk file or magnetic tape produced by DISPER.

B,XØ,STOPK,EPS,ND,STOPMU — The first record on TAPE5 is the defining parameters used by DISPER to create the dispersion curves. They are read in under the format (4E22.7, 2I5//). For definitions see inputs to DISPER.

N — Empirical data, format (8F10.5). See Inputs to DISPER. There will be ND-1 values of N.

NPTS — Integer
The number of (K,μ) pairs for the current mode. Format (//I5).

K1, MU1

Real
The K and MU values of each mode, (2E22.7).

STOPX

Real
The depth to which the mode function is to be calculated. Must be greater than or equal to XØ.

IFIRST

Integer
The first mode to be calculated.

LAST

Integer
The last mode to be calculated. All modes between IFIRST and LAST are calculated.

B. OUTPUTS

1. Printouts

a. The parameters defined by TAPE5 and the NAMELIST option are listed at the end of the program for verification purposes.

b. N is listed.

c. Mode number is printed followed by a list of PSI values for that mode. Format is (4E22.7).

2. Plots

10" by 10" graphs of the (X,PSI) values will be plotted, one plot for each mode.

C. EXTERNAL REFERENCES

MODE references several external subroutines that must be provided by the user through control cards that attach the appropriate permanent files.

The necessary routines are listed below under the name of the permanent file on which they reside.

1. INTMODE

INITIAL
Reads the data values of N(X), calculates the integral

$$\int_0^\infty N(x)\, dx$$

DIFF
Numerically solves the differential equation using the Numerov-Manning-Millman method.

PSCALE
Scales the axes to the calculated data. Called by PLOTMODE.

ERRPRO
Processes detected errors through a call to ABRTJOB. Is called by all of the routines on this permanent file.

INTNP
Calculates the integral

$$\int_0^\infty N^2(x)\Psi^2(x)\, dx$$

FUNCT2
Calculates X * J(ν,X)**2. Called by INTNP.

FUNCT4
Calculates the alternative asymptotic approximation for X * J(ν,X)**2. Called by INTNP.

PLOTMOD
Sets up the calls to the CalComp plotting routines.

2. IDALIB

PLOTS
PLOT
LINE
DAXIS
SYMBOL
NUMBER

CalComp plotting routines called by PLOTMOD.

ABRTJOB — Error processor that generates TRACEBACK, prints error messages, and terminates the job. Called by ERRPRO.

GAUSS — Numerical integrating routine using Gaussian quadrature. Called by INTNP.

3. BESSEL

JBESS
JAIRY
GAMLN — Routines to calculate the BESSEL functions. Acquired from the Argonne National Laboratory.

D. ERROR MESSAGES

A table of error messages generated by MODE and possible corrective actions that might resolve the problem is presented below.

TABLE 2. ERROR MESSAGES GENERATED BY MODE AND POSSIBLE CORRECTIVE ACTIONS

Message	Significance	Action	Issued By
More than 500 pairs were needed for this mode.	DISPER was altered to permit more than 500 pairs to be calculated.	Make similar changes in MODE.	MODE
The maximum K value on TAPE5 is less than the K of interest.	The dispersion curves were not calculated to this value of K.	Rerun DISPER with STOPK greater than K, or reduce value of K.	MODE
There are not enough modes on TAPE5.	STOPMU is less than LAST.	Reduce LAST to less than STOPMU, or rerun DISPER with STOPMU greater than LAST.	MODE
Indefinite operand Ø/Ø	Solution to differential equation is inconsistent.	Look for coding errors in the routine DIFF.	DIFF
Division by Zero	Algorithm for solving the differential equation has broken down.	Data points may be too far apart. Introduce more data, perhaps through interpolation.	DIFF

E. DETAILS OF JOB EXECUTION

The following is a sample card deck for executing MODE, when the dispersion curves are on a permanent file.

```
IPD,MT1, DRAPER,323/58
ATTACH(MODE,ID=PD)
FTN(I=MODE,L=0)
MAP(OFF)
ATTACH(INTMODE,ID=PD)
ATTACH(BESSEL,ID=PD)
ATTACH(IDALIB,ID=CG)
LABEL(PLOTAPE,W,RING,L=NOSHEPLOT,D=LO,X=SV,VSN=0)
LDSET(LIB=INTMODE/BESSEL/IDALIB)
LGO.
789

 $INPUT IFIRST=1,LAST=2,K=.02,STOPX=250.$
6789
```

Next is a sample card deck for executing MODE when the dispersion curves are on a magnetic tape. Note the VSN number should be the one assigned at the time DISPER executed.

```
IPD,MT2, DRAPER,323/58
ATTACH(MODE,ID=PD)
FTN(I=MODE,L=0)
MAP(OFF)
ATTACH(INTMODE,ID=PD)
ATTACH(BESSEL,ID=PD)
ATTACH(IDALIB,ID=CG)
LABEL(TAPE5,R,NORING,L=DISPCURVE,VSN=5555)
LABEL(PLOTAPE,W,RING,L=NOSHEPLOT,D=LO,X=SV,VSN=0)
LDSET(LIB=INTMODE/BESSEL/IDALIB)
LGO.
789

 $INPUT IFIRST=1,LAST=2,K=.02,STOPX=250.$
6789
```

VII. EXAMPLES OF DISPERSION RELATIONS AND EIGENFUNCTIONS OBTAINED WITH INTMODE

The two Väisälä frequency profiles considered in the sample calculations employing INTMODE are shown in Fig. 1. The profile labeled "exponentially stratified ocean" corresponds to a deep ocean without a thermocline and $N(y) = .00528 \exp - y/1300$ radian/sec, where y is in meters. This profile is identical to the one used by Garrett and Munk (Ref. 7). For the exponential Väisälä frequency profile, the mode functions are Bessel functions. Consequently, results of INTMODE for this profile can be compared with results based on analytical formulae, thus providing a check on the accuracy of the numerical technique. The sharp thermocline, labeled "STD data set", is taken from (Ref. 8) and is based on measured towed thermistor chain data in the tropical Pacific Ocean. The data extends to a depth of 220 meters; at greater depths an exponential profile with a decay constant of 1300 meters is assumed.

The dispersion curves for the first 25 modes, corresponding to the STD data set, are plotted in Fig. 2. The angular frequency is in radians/sec. For the STD data set, plots of the first four internal wave modes are shown in Fig. 3 for $K = .01$ radians/meter ($\lambda \simeq 628$m) and in Fig. 4 for $K = .02$ radians/meter ($\lambda \simeq 328$m). The mode functions are all normalized in accordance with

$$\int_0^\infty \psi^2(y)\, N^2(y) dy = 1 .$$

Since N(y) decays exponentially with depth, this normalization constraint leads to a progressive increase of the mode maximum with mode number and depth, a feature corroborated by the plots in Figs. 3 and 4.

INTMODE is capable of yielding mode functions of any order (subject to course to the resolution of the data with respect to depth. Figure 5 shows a plot of the 25th mode.

The dispersion relations for the exponential profile are shown in Fig. 6; the first four mode functions are shown in Fig. 7 and Fig. 8, for K = .01 and K = .02 radians/meter, respectively.

Examples of dispersion relations for other profiles are shown in Figs. 9 and 10. The corresponding profiles are, respectively, those referred to in Fig. 12 as STD data and NRL data. An additional example of dispersion relations, corresponding to a pulse shaped profile, is shown in Fig. 11. This profile is of the form

$$\begin{aligned} N(y) &= 0; && 0 < y < 65.45\text{m}, \\ N(y) &= .0105 \text{ rps} && 65.45\text{m} < y < 111\text{m} \\ N(y) &= 0; && y > 111\text{m}. \end{aligned}$$

In the region y < 111m N(y) was approximated by the rapidly decaying exponential exp - y/5.

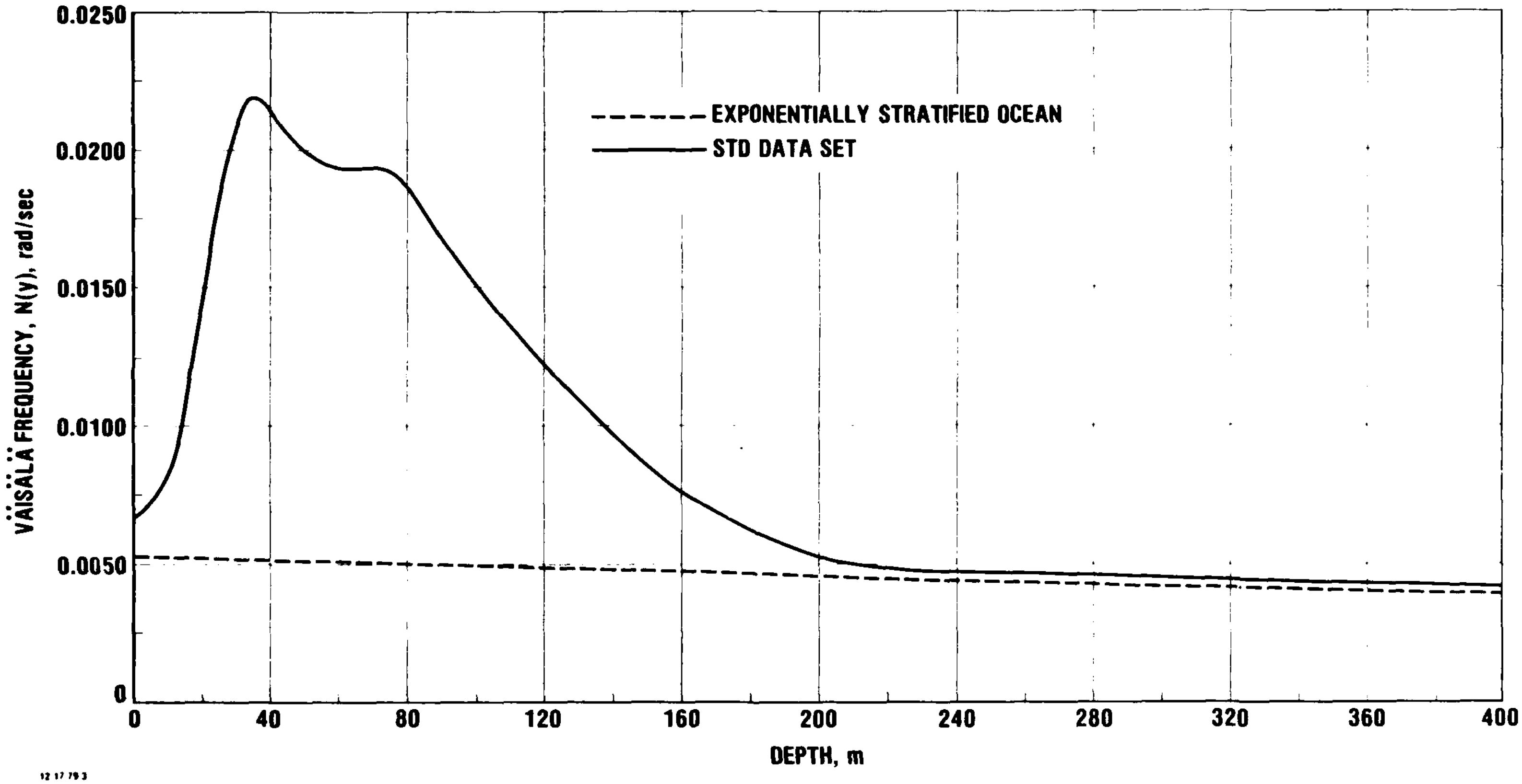


FIGURE 1. Väisälä frequency profiles used in the calculation of internal wave mode functions and dispersion relations.

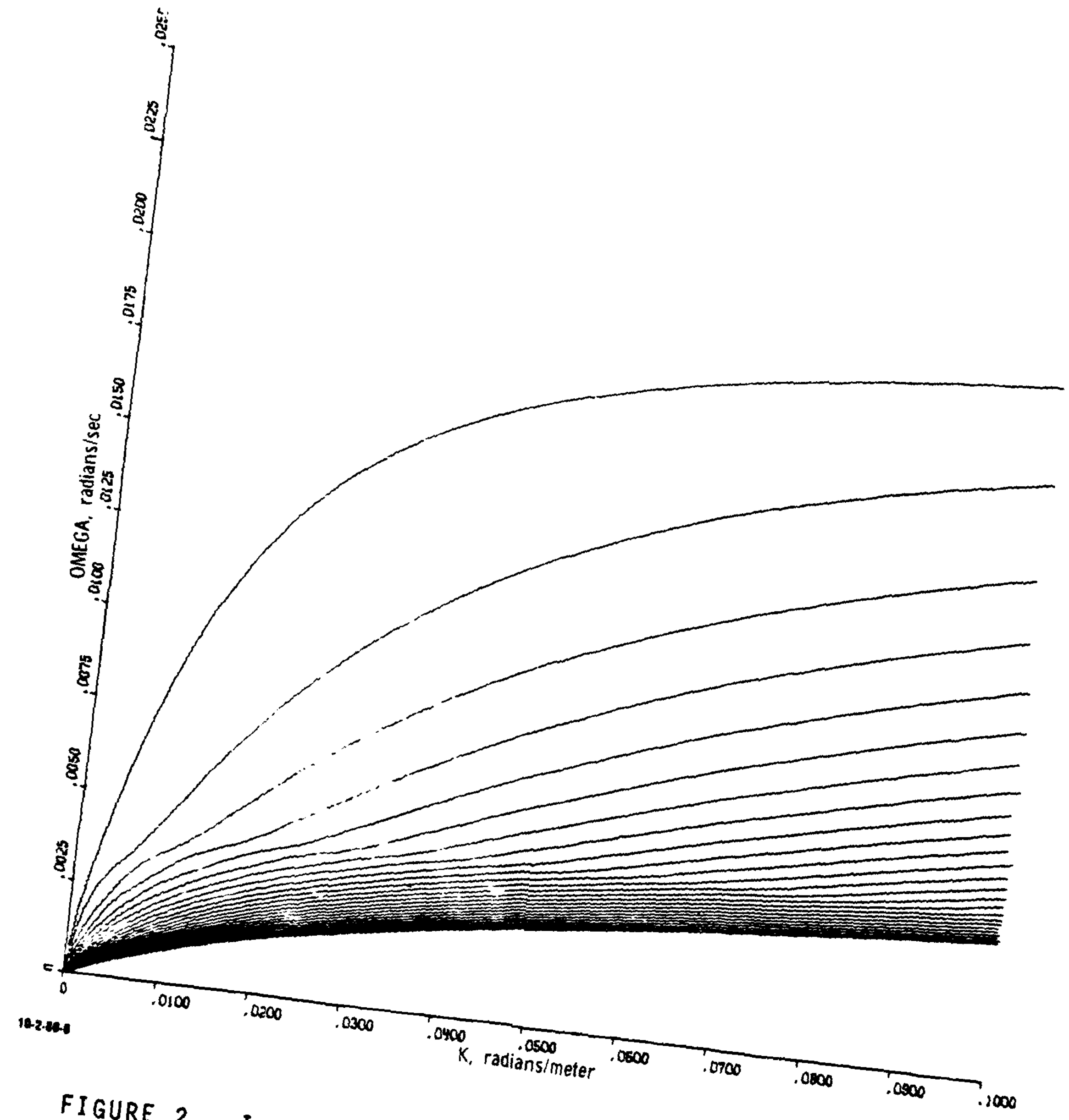
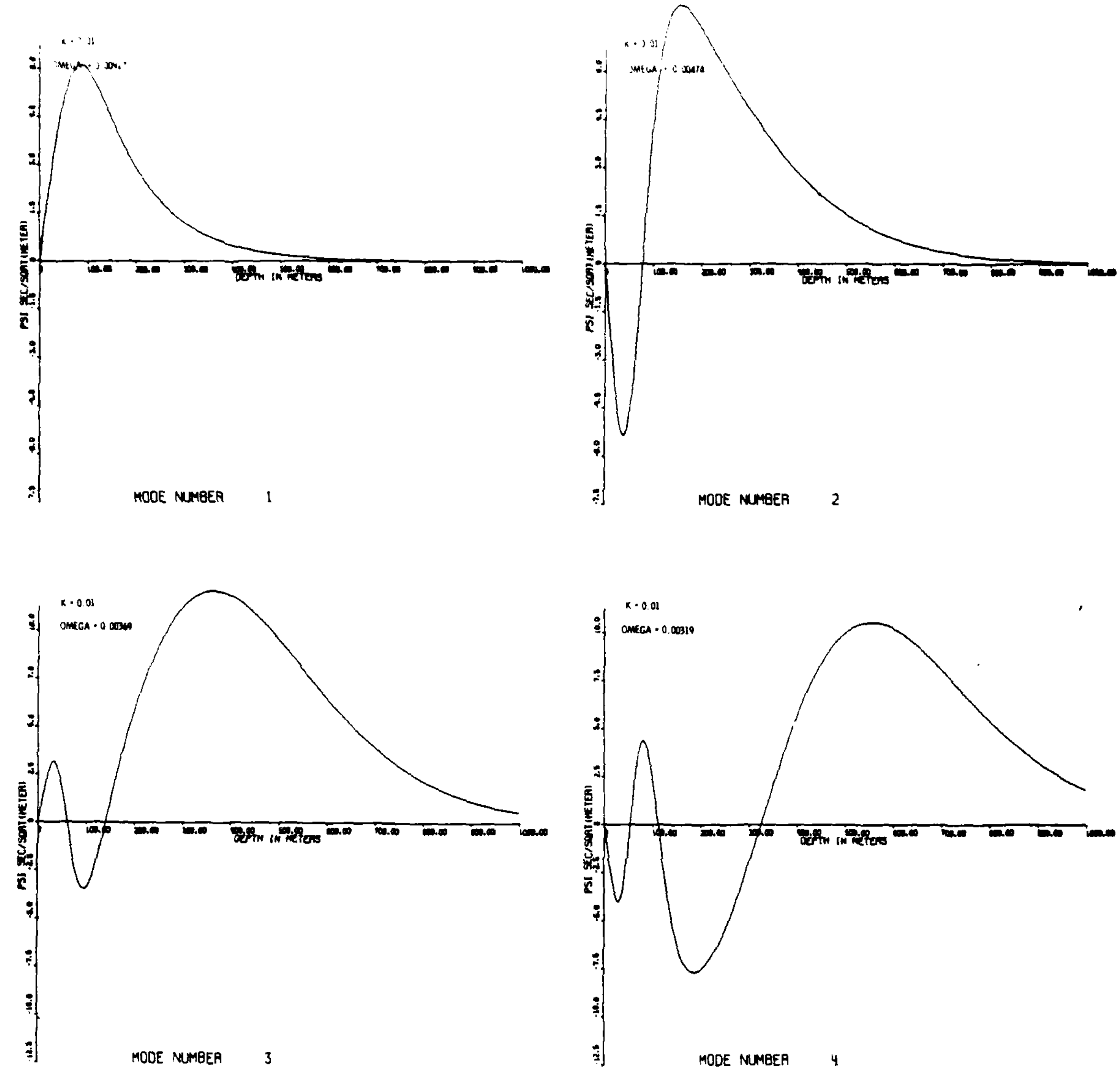
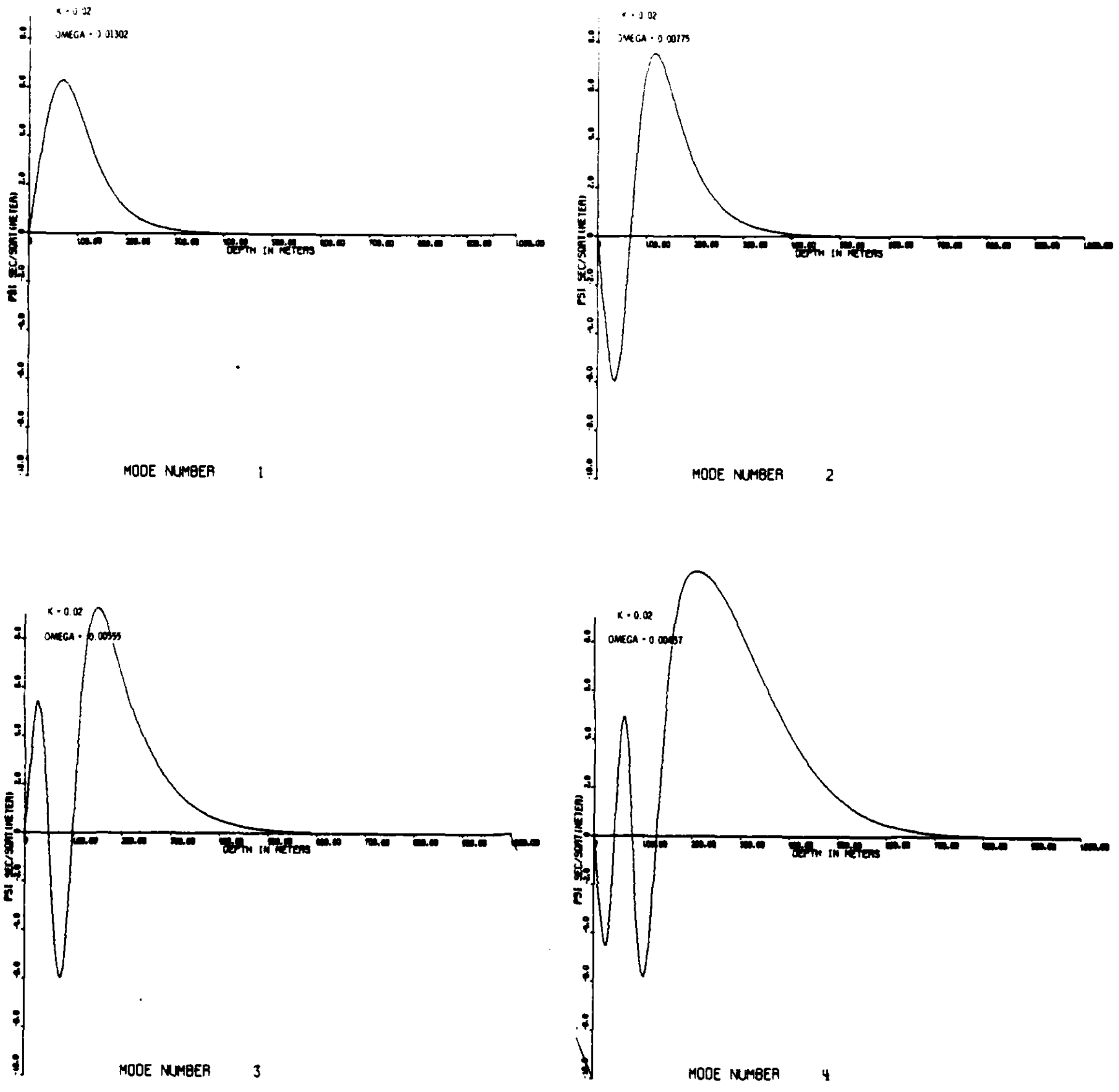


FIGURE 2. Internal wave mode dispersion curves for the Väisälä frequency profile corresponding to the "STD data set" in Fig. 1.



10-2-80-9

FIGURE 3. The first four internal wave modes for the STD data set (Fig. 1) (λ ≈ 628 meters).



18-2-88-18

FIGURE 4. The first four internal wave modes for the STD data set (Fig. 1) ($\lambda \simeq 314$ meters).

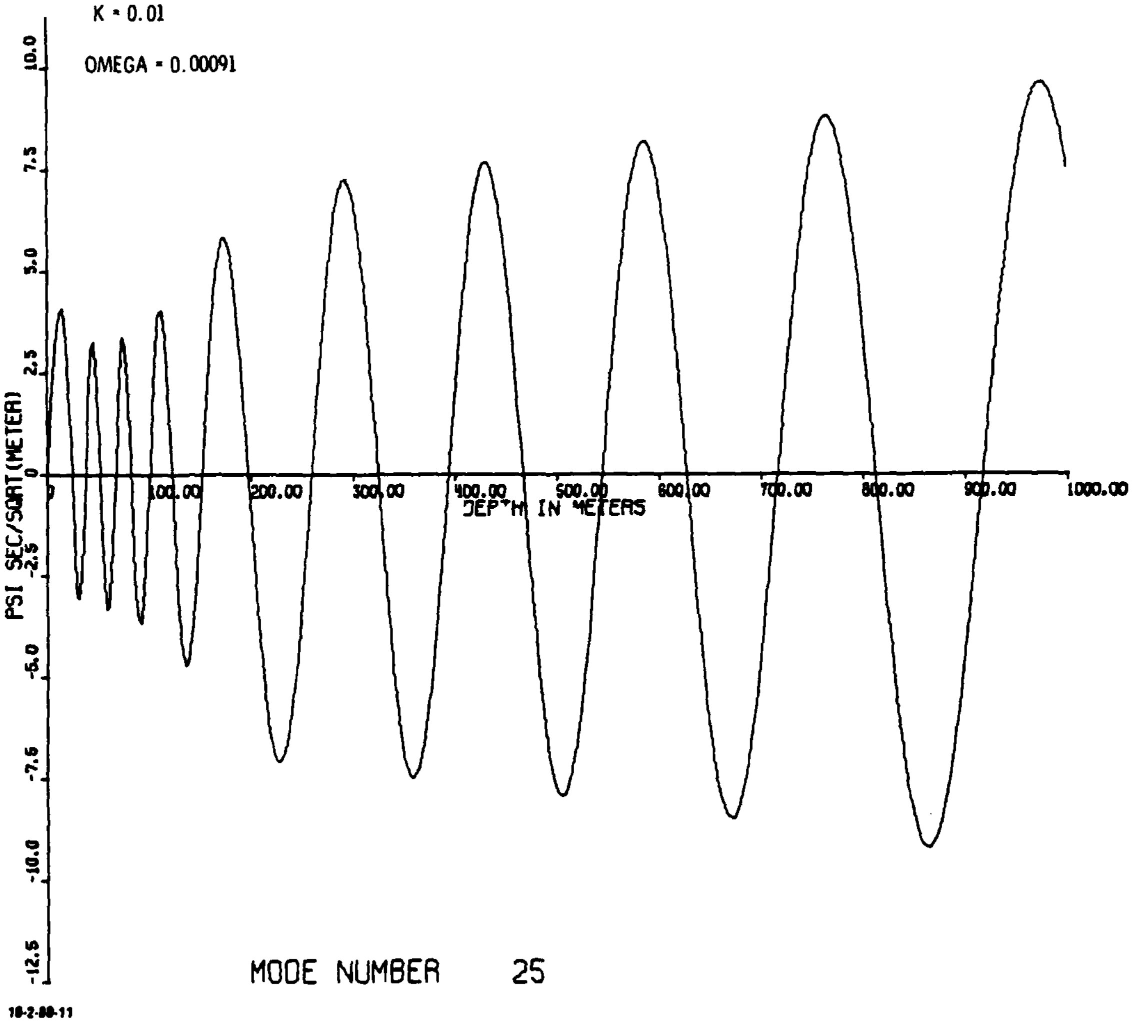


FIGURE 5. The 25th mode for the STD data set (λ ≃ 628 meters).

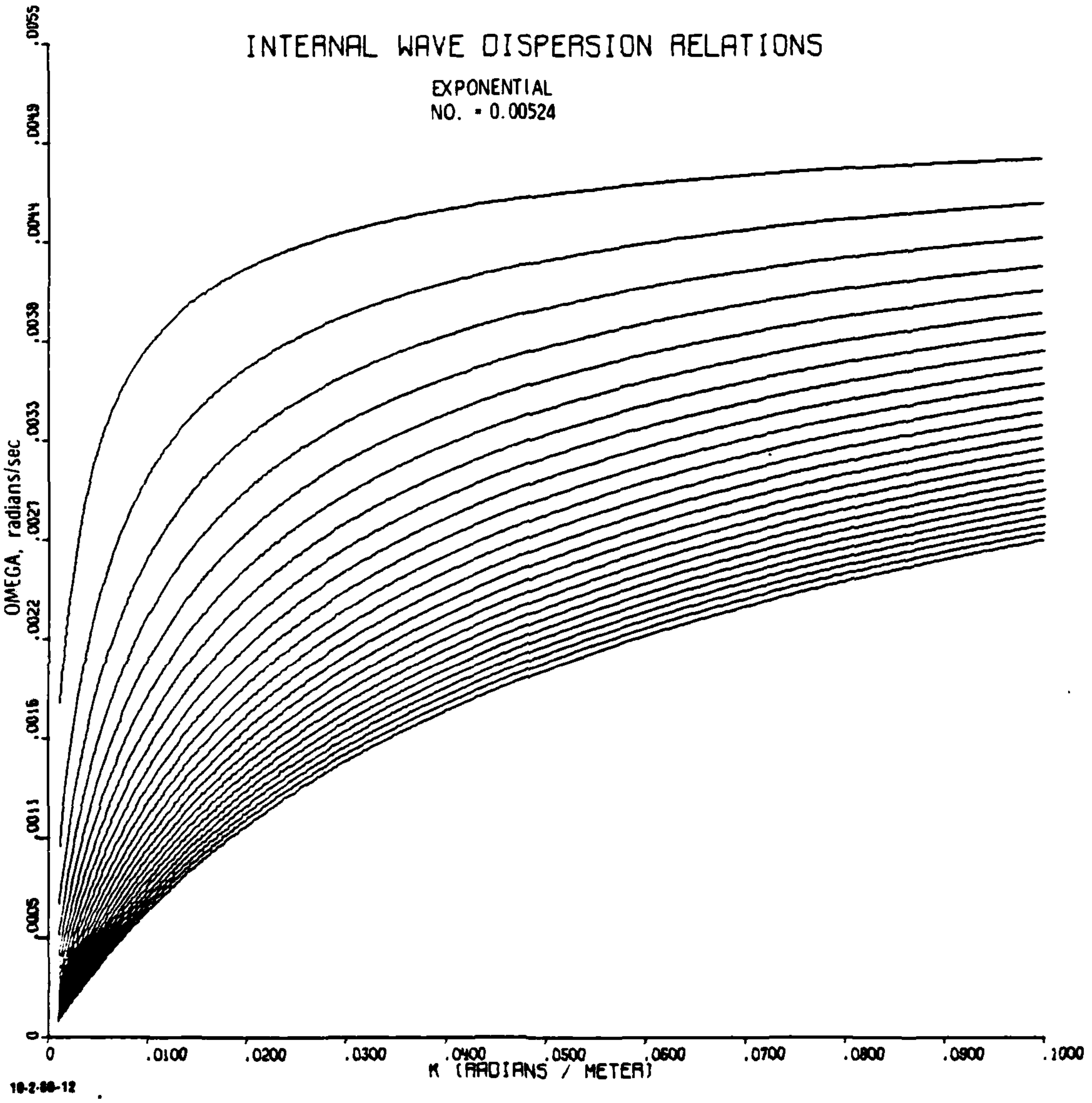
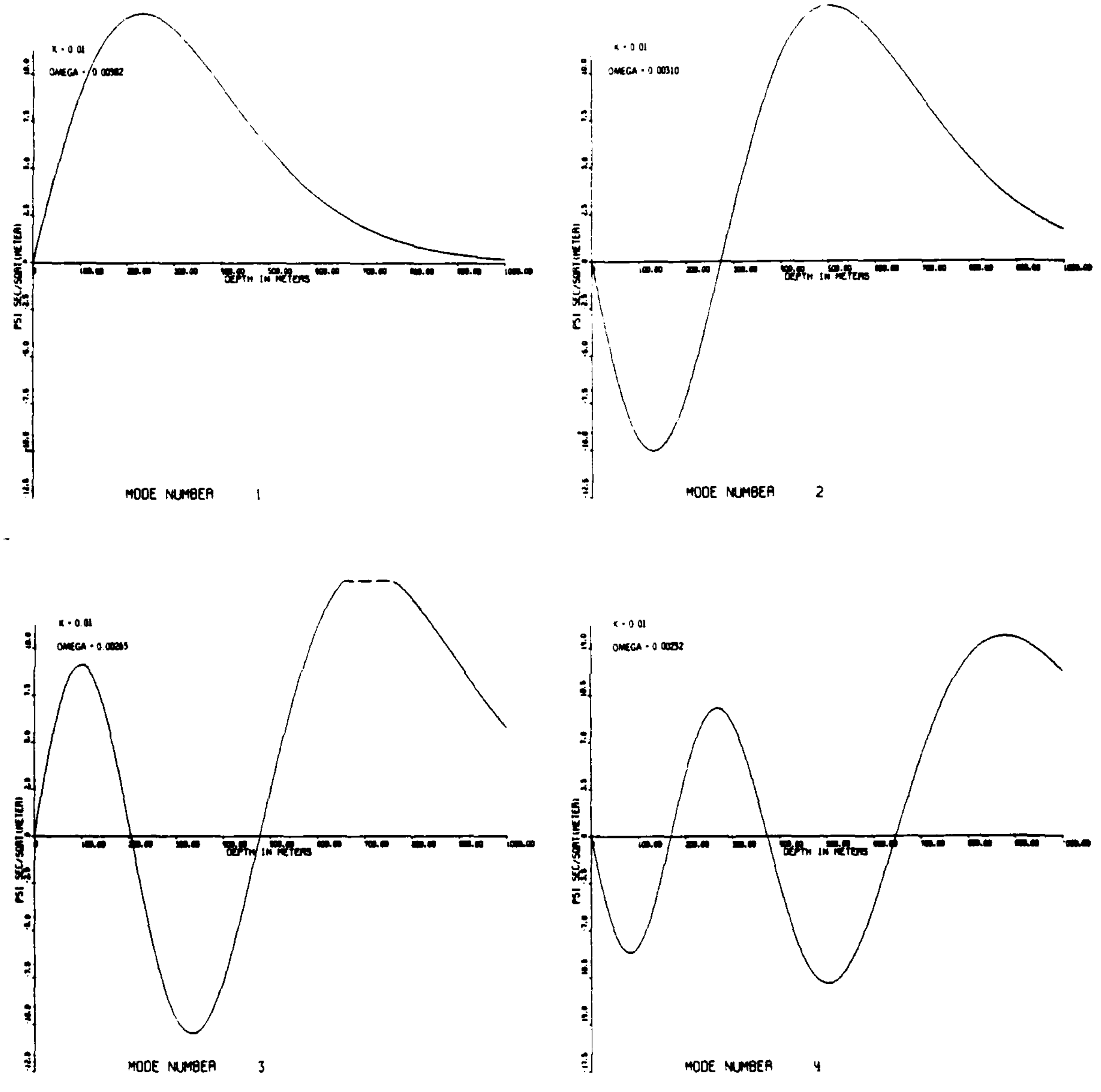
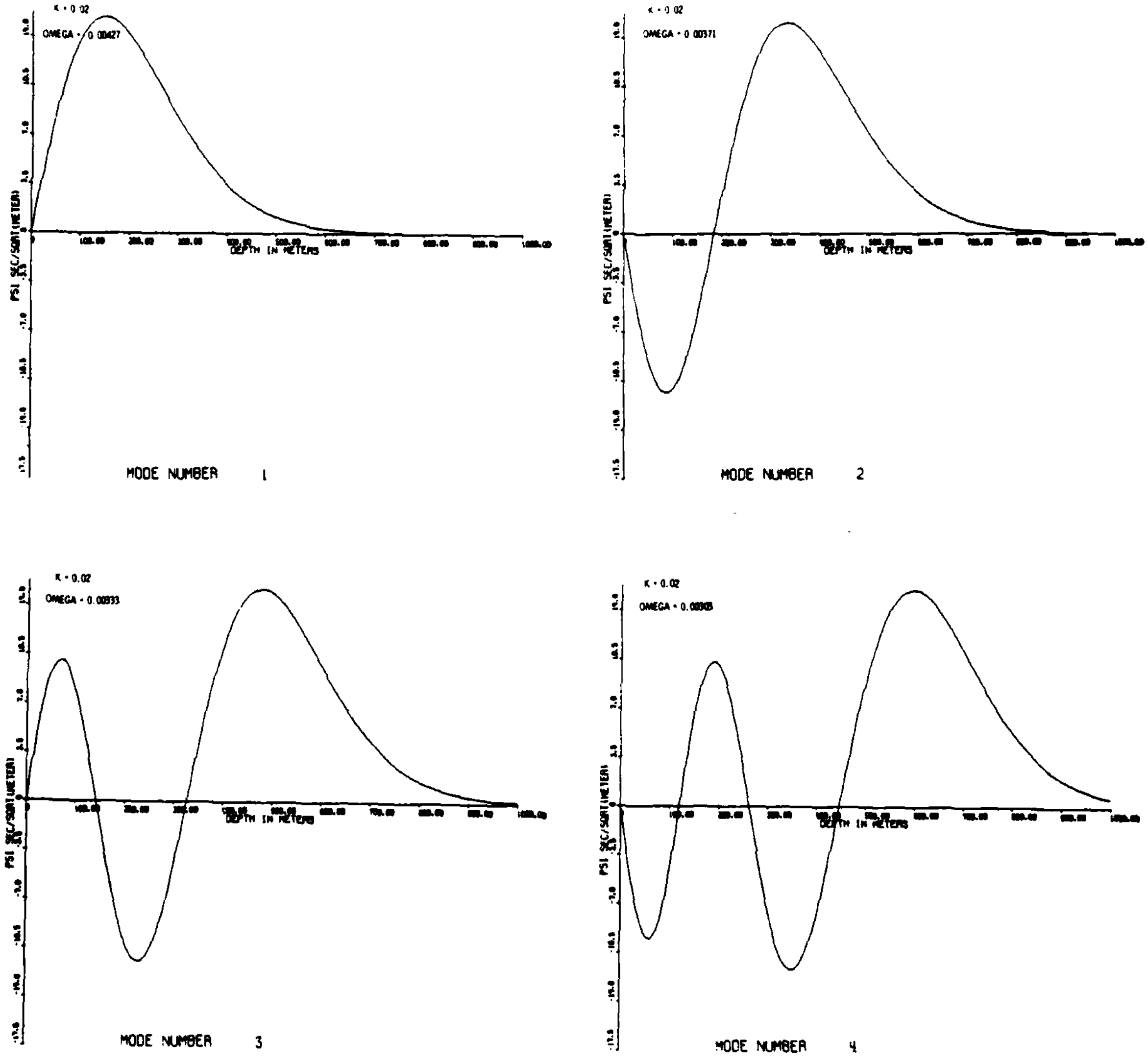


Figure 6. Internal wave mode dispersion curves for an exponentially decreasing Väisälä frequency profile.



18-2-88-13

FIGURE 7. The first four internal wave modes for the exponentially decreasing Väisälä frequency profile (λ ≃ 628 meters).



10-2-80-14

FIGURE 8. The first four internal wave modes for the exponentially decreasing Väisälä frequency profile ($\lambda \simeq 314$ meters).

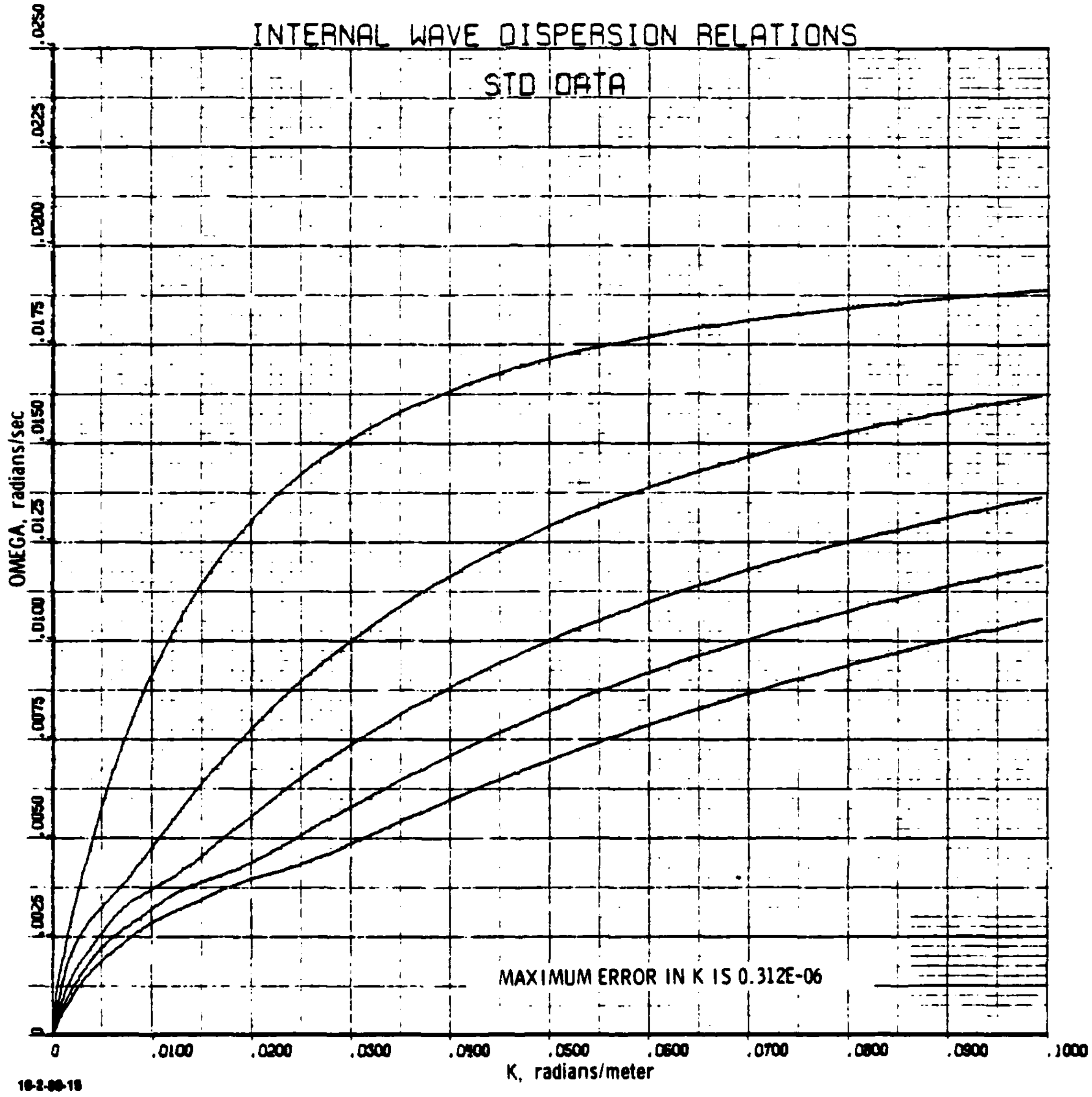


FIGURE 9. Internal wave dispersion relations -- STD data.

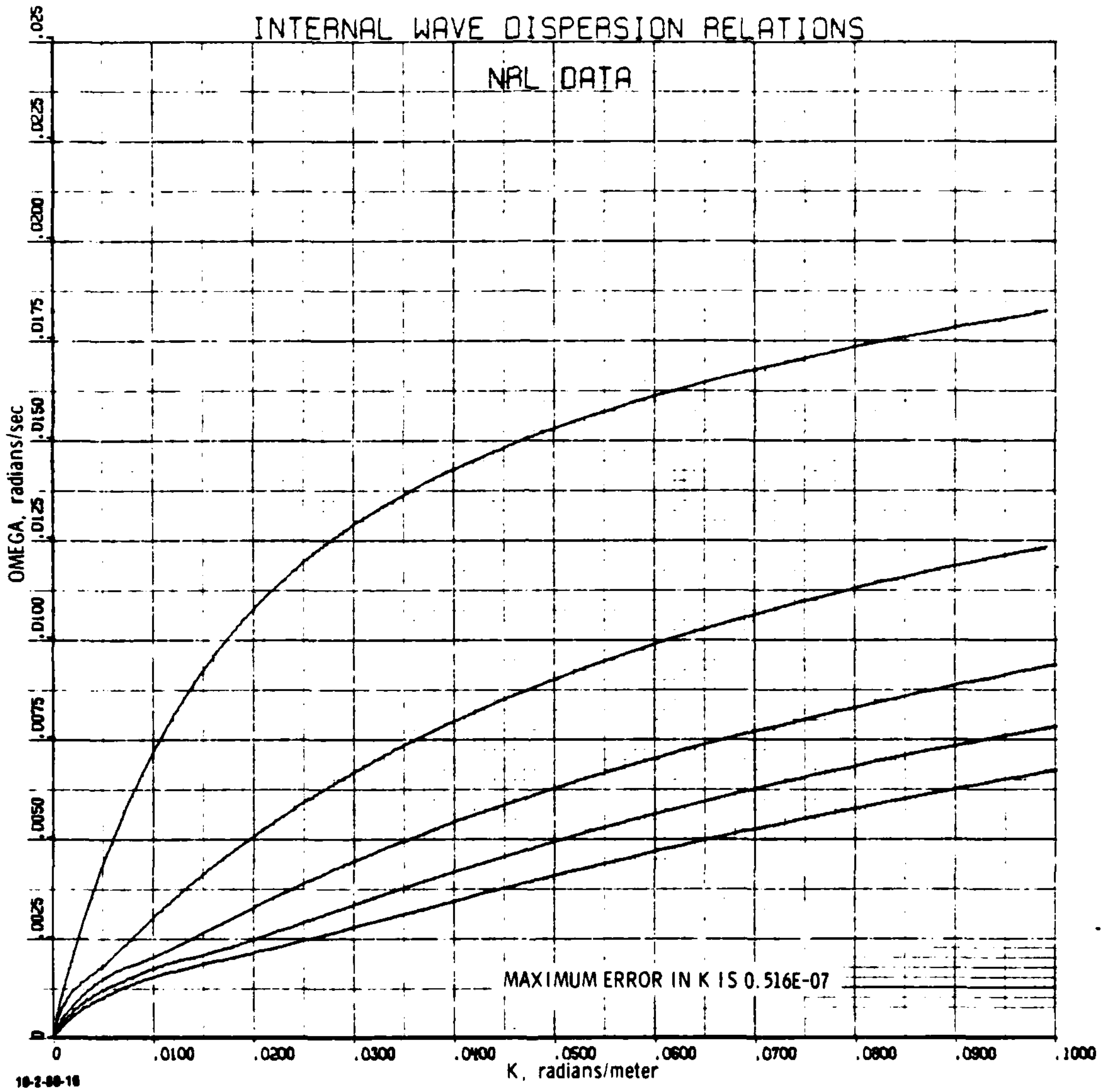


FIGURE 10. Internal wave dispersion relations -- NRL data.

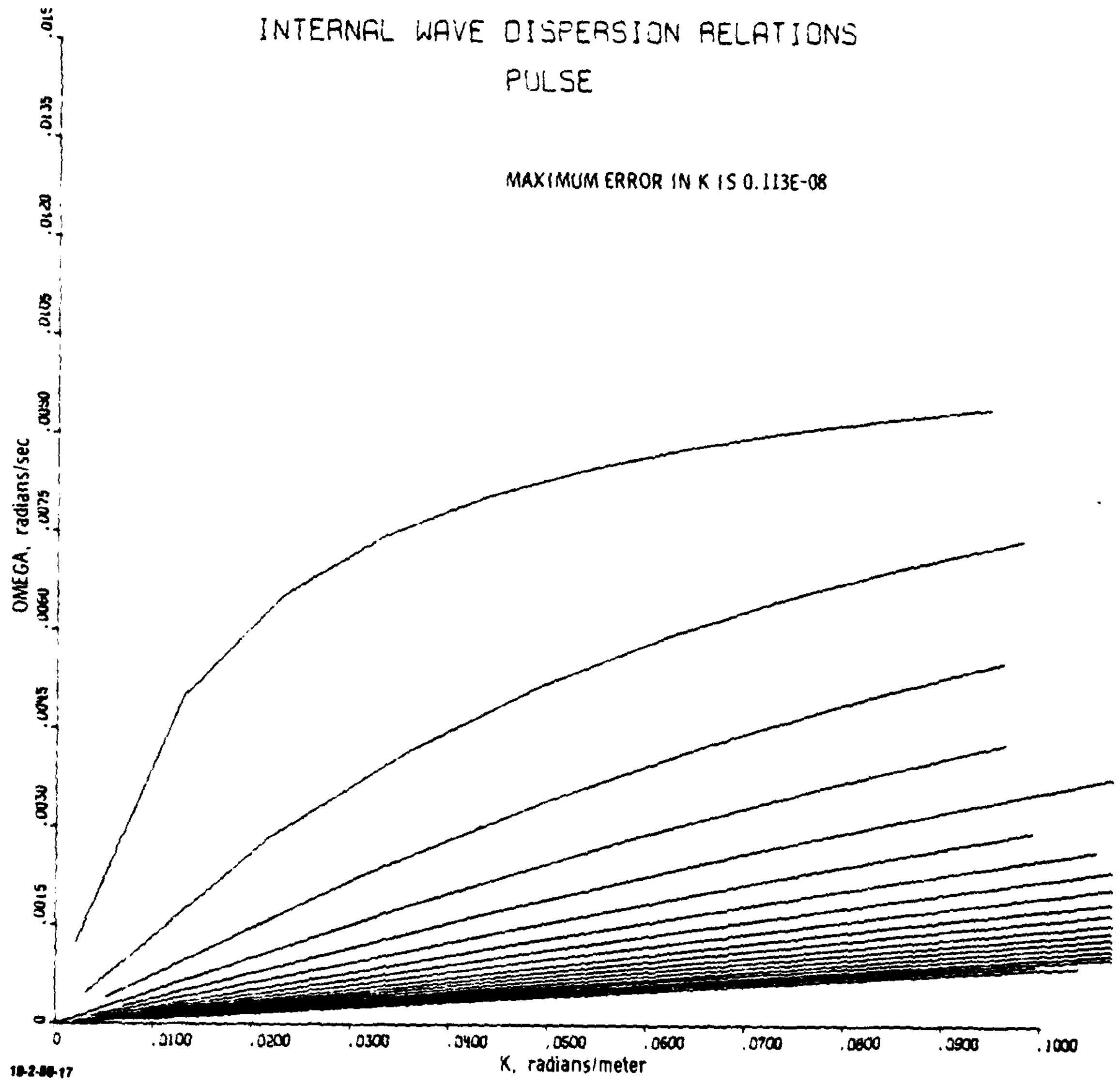


FIGURE 11. Internal wave dispersion relations -- pulse.

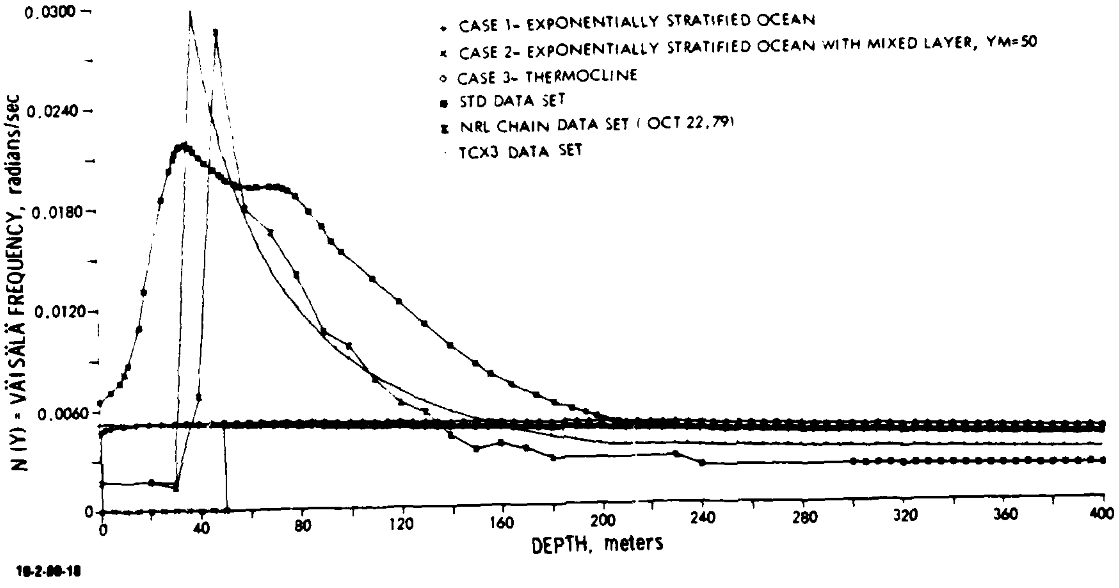


FIGURE 12. Väisälä Frequency Profiles.

REFERENCES

1. Institute for Defense Analyses, *Electromagnetic Fields Induced by Ocean Currents*, IDA Paper P-1399, W. Wasylkiwskyj, July 1979.

2. Institute for Defense Analyses, *Ocean Internal Wave Induced Magnetic Fields Within A Submerged Buoy*, IDA Paper P-1464, I. W. Kay and W. Wasylkiwskyj, January 1980.

3. D. M. Milder, *Partitioning of Energy, Vorticity, and Strain in Upper-Ocean Internal Waves*, Arete´ Associates Report.

4. E. C. Titchmarsh, *Eigenfunction Expansions Associated With Second-Order Differential Equations*, Clarendon Press, Oxford, 1962.

5. R. Courant and D. Hilbert, *Methoden Der Mathematischen Physik*, Volume I, Springer, Berlin, 1931.

6. K. S. Kunz, *Numerical Analysis*, McGraw-Hill, New York 1957.

7. C. J. R. Garrett and W. H. Munk, *Space-Time Scales of Internal Waves*, Geophys. Fluid Dyn., 2, pp. 225-264, 1972a.

8. K. W. Nelson and D. M. Milder, *Internal Wave Displacement and Vertical Coherence Spectra from Towed Thermistor Chain Measurements in the Upper Thermocline*, Arete´ Associates, Santa Monica, CA, August 1977.

APPENDIX A

INTMODE SUBROUTINES

Contents

```
*DECK DISPER
      PROGRAM DISPER(INPUT,OUTPUT,TAPE2,TAPE3,TAPE5=INPUT)
C
C
C        ************************************************
C        *        A CDC 6400 SUBROUTINE                 *
C        *                                              *
C        *    PROGRAMMER                                *
C        *        P. J. DRAPER                          *
C        *        INSTITUTE FOR DEFENSE ANALYSIS        *
C        *        ARLINGTON, VA                         *
C        *        MARCH 5, 1980                         *
C        *                                              *
C        *                                              *
C        *   DISPER CALCULATES AND GRAPHS DISPER-       *
C        *   SION CURVES ACCORDING TO NUMERICAL         *
C        *   TECHNIQUES DEVELOPED BY DRS. I. KAY        *
C        *   AND W. WASYLKIWSKYJ OF IDA                 *
C        *                                              *
C        *   PROBLEM                                    *
C        *        GIVEN A FUNCTION N(X) DEFINED         *
C        *   NUMERICALLY FROM X = 0 TO X = X0 AND       *
C        *   BY THE RELATION                            *
C        *                                              *
C        *        N(X) = N0 * EXP(-(X - X0) / B)        *
C        *                                              *
C        *   FOR X .GE. X0 AND A DIFFERENTIAL           *
C        *   EQUATION                                   *
C        *                                              *
C        *   D**2(PSI)/D(X)**2 =                        *
C        *                                              *
C        *        (K**2 - MU**2 * N(X)**2) * PSI        *
C        *                                              *
C        *   WITH PARAMETERS MU AND K THAT TAKE         *
C        *   ON VARIOUS REAL VALUES                     *
C        *                                              *
C        *   IT IS KNOWN THAT IN THE REGION WHERE       *
C        *   X .GE. X0 THAT ONE SOLUTION OF THE         *
C        *   DIFFERENTIAL EQUATION IS THE BESSEL        *
C        *   FUNCTION WITH INDEX K * B AND ARGU-        *
C        *   MENT MU * B * N0 * EXP(-(X - X0)/B)        *
C        *                                              *
C        *   BY VARYING THE PARAMETERS MU AND K         *
C        *                                              *
C        *   ONE CAN FIND DIFFERENTIAL EQUATIONS, ONE OF *
C        *   WHOSE SOLUTIONS MATCH THE BESSEL           *
C        *   FUNCTION SOLUTION WHEN X .GE. X0 AND       *
C        *   ALSO VANISH AT X = 0.                      *
C        *                                              *
C        *   THIS PROGRAM DETERMINES THE RELATION       *
C        *                K = K(MU)                     *
C        *   THAT PROVIDES JUST SUCH DIFFERENTIAL       *
C        *   EQUATIONS                                  *
C        *                                              *
C        *   IT THEN GRAPHS  (K / MU) VS. K             *
C        *                                              *
C        ************************************************
```

```
C          *                                                        *
C          *     DESCRIPTION OF ARGUMENTS                           *
C          *          THE FOLLOWING PARAMETERS ARE                  *
C          *          MEMBER OF THE NAMELIST GROUP                  *
C          *          CALLED PARAM                                  *
C          *     X0             REAL                                *
C          *                    THE X-COORDINATE OF THE             *
C          *                    LAST POINT OF THE NUMERI-           *
C          *                    CAL DATA                            *
C          *                                                        *
C          *     B              REAL                                *
C          *                    THE DECAY CONSTANT                  *
C          *                                                        *
C          *     ND             INTEGER                             *
C          *                    THE NUMBER OF NUMERICALLY           *
C          *                    DEFINED POINTS + 1                  *
C          *                                                        *
C          *     STOPK          REAL                                *
C          *                    THE MAXIMUM K VALUE TO BE           *
C          *                    CALCULATED FOR EACH MODE            *
C          *                                                        *
C          *     STOPMU         INTEGER                             *
C          *                    THE NUMBER OF MODES TO BE           *
C          *                    CALCULATED                          *
C          *                                                        *
C          *     ESP            REAL                                *
C          *                    ERROR CRITERIA IMPOSED ON           *
C          *                    THE SOLUTION TO THE DIFFE-          *
C          *                    RENTIAL EQUATION                    *
C          *                                                        *
C          *                                                        *
C          *                                                        *
C          *     THE LAST IMPUTS TO THE PROGRAM ARE                 *
C          *     THE NUMERICAL VALUES THAT DEFINE                   *
C          *     N(X) FROM 0 TO X0. THEY ARE READ                   *
C          *     FROM CARDS.  THE VALUES MUST BE SO                 *
C          *     PUNCHED AS TO BE READ UNDER THE                    *
C          *     FORMAT(8F10.5)                                     *
C          *                                                        *
C          *     N              REAL ARRAY                          *
C          *                    DIMENSION 500                       *
C          *                                                        *
C          **********************************************************
C
C
C
C
      COMMON/INPUTS/B,ND,X0,EPS,STOPMU,STOPK, STOPX
      COMMON PSI(500),N(500),N0,DELTAX,NT,ITITLE
      COMMON /OUT/ SAVEK(500), SAVEMU(500)
      DIMENSION TEMPK(500),TEMPMU(500)
      REAL N, N0, MU, K0, KMAX, MU0, K1
      INTEGER STOPMU
      NAMELIST/PARAM/B,ND,X0,STOPMU,STOPK,EPS
      MODE = 1
      ERRMAX = 0.
      PI = ACOS(-1.)
      IERR = 6
```

```
C
C     READ PARAMETERS
C
    5 READ PARAM
      PRINT PARAM
C
C     READ NUMERICAL VALUES FOR N AND INTEGRATE N
C
      CALL INITIAL(SUM1)
      PRINT 7000, SUM1
 7000 FORMAT(* SUM1*,5X,F22.7)
      PRINT 100,(N(I),I=1,ND)
C
C
C     WRITE NAMELIST AND N TO TAPE2
C
C
      CALL OUTPUTK (1)
C     FIND MAXIMUM VALUE FOR N
C
      KMAX=N(1)
      DO 10 I=2,ND
   10 KMAX=AMAX1(KMAX,N(I))
C
C     CALCULATE MU0 FOR THE FIRST BRANCH
      DY = .1/SUM1
      MU0 = XMU0(0.,DY)
C
C     CALCULATE INITIAL ESTIMATE FOR THE INCREMENT IN MU
C     ALONG A MU0 BRANCH
C
      TDELT = .5 / SUM1
      XDELT = TDELT
C
C     START THE PROCESS OF APPROXIMATING K
C
      K0 = 0.
      MU=MU0
      KOUNT = 0
  200 CONTINUE
C
C     CALCULATE THE UPPER ESTIMATE OF K,K1
C
      K1 = MU * KMAX
C
C
C     CALL SUBROUTINE GUESS TO IMPROVE THE ESTIMATE K
C     K1 IS UPPER ESTIMATE. K0 IS LOWER ESTIMATE. THE
C     IMPROVED ESTIMATE IS RETURNED IN K1.
C
C
C
      CALL GUESS(K0,K1,MU)
C
C
C     CHECK FOR ERROR CODES RETURNED IN NT
C
```

```
C      NT = 9          INDICATES THE ALGORITM COULD NOT PRO-
C                      DUCE A K FOR MU0 ON THE FIRST BRANCH.
C                      K1 IS NOT ABOVE THE CURVE.
C
C
       IF(NT .EQ. 9)CALL ERRPRO(5)
C
C
C
C
C
C      THE MAXIMUM ESTIMATED ERROR FOR K IS
C      RETURNED BY GUESS IN THE VARIABLE K0
C
C
       EPSK = K0
       ERRMAX = AMAX1(EPSK,ERRMAX)
C
C
C      INCREMENT KOUNT. IF 500 (MU, K) PAIRS HAVE
C      BEEN CALCULATED AND STOPK NOT REACHED AN
C      ERROR MESSAGE IS PRINTED, A FTN ERROR 52
C      GENERATED, AND THE JOB ABORTED. EITHER RE-
C      DUCE STOPK OR INCREASE THE DIMENSION OF
C      SAVEK(500),SAVEMU(500,)TEMPK(500),TEMPMU(500),
C      THEN CHANGE THE TEST ON KOUNT
C
       KOUNT = KOUNT + 1
       IF(KOUNT .GE. 500)CALL ERRPRO(IER)
       SAVEK(KOUNT) = K1
       SAVEMU(KOUNT) = MU
C
C      CALCULATE THE LOWER K ESTIMATE FOR THE NEXT
C      MU VALUE
C      INCREMENT MU
C
       K0 = K1 + TDELT * K1 / MU
       IF(K0 .GT. STOPK)GO TO 300
       MU = MU + TDELT
    96 FORMAT(1X,I3,2(5X,F20.5))
       GO TO 200
C
C      WE ARE FINISHED WITH THIS BRANCH
C
   300 CONTINUE
       NPTS=KOUNT
C
C      SAVE CURRENT (MU,K) PAIRS TO USE IN ESTIMATING
C      K1 ALONG THE NEXT MODE
C
C
       DO 900 I=1,KOUNT
       TEMPK(I)=SAVEK(I)
   900 TEMPMU(I)=SAVEMU(I)
       CALL OUTPUTK (KOUNT)
C
C
C      PRINT THE MAXIMUM ERROR ESTIMATED FOR K FOR
```

```
            C     THIS MODE
            C
                  PRINT 98, MODE, ERSK
               98 FORMAT(* ESTIMATED MAXIMUM ERROR IN K FOR MODE *, I4, * IS  *,
                 1E22.7)
            C
            C     BEGIN THE NEXT BRANCH
                  MODE = MODE + 1
                  IF(MODE .GT. STOPMU)GO TO 400
             1000 CONTINUE
                  NT = 0
            C
                  MU0 = XMU0(MU0,DY)
                  K0=0.
                  MU=MU0
                  KOUNT = 0
                  TDELT = XDELT
                  LCOUNT = 0
              201 CONTINUE
            C
            C     FIRST APPROXIMATION OF K FOR THIS BRANCH
            C     BEGINS BY FINDING THE K FOR THE LAST BRANCH
            C
                  DO 20 I=1,NPTS
                  IF(TEMPMU(I) .GE. MU)GO TO 30
               20 CONTINUE
                  I = NPTS + 1
               30 SLOPE = TEMPK(I-1) / TEMPMU(I-1)
            C
            C     CALCULATE THE UPPER ESTIMATE OF K,K1
            C
            C
                  K1 = SLOPE * MU
            C
            C     CALL SUBROUTINE GUESS TO IMPROVE THE ESTIMATE K
            C     K1 IS UPPER ESTIMATE, K0 IS LOWER ESTIMATE. THE
            C     IMPROVED ESTIMATE IS RETURNED IN K1.
            C
                  CALL GUESS(K0,K1,MU)
            C
            C     CHECK FOR ERROR CODES RETURNED IN NT
            C
            C
            C     NT = 9          A BRANCH WE ANTICIPATED DID NOT
            C                     OCCUR HERE. A RESULT OF OUR ESTI-
            C                     MATION PROCEDURE FOR MU.
            C
            C
                  IF(NT .EQ. 9)GO TO 1000
            C
              221 CONTINUE
            C
            C
            C
            C     THE MAXIMUM ESTIMATED ERROR FOR K IS
            C     RETURNED BY GUESS IN THE VARIABLE K0
            C
```

```
      C
            ERSK = K0
            ERRMAX = AMAX1(ERSK,ERRMAX)
      C
      C
      C
      C     INCREMENT KOUNT. IF 500 (MU, K) PAIRS HAVE
      C     BEEN CALCULATED AND STOPK NOT REACHED AN
      C     ERROR MESSAGE IS PRINTED. A FTN ERROR 52
      C     GENERATED. AND THE JOB ABORTED. EITHER RE-
      C     DUCE STOPK OR INCREASE THE DIMENSION OF
      C     SAVEK(500),SAVEMU(500,)TEMPK(500),TEMPMU(500).
            KOUNT=KOUNT + 1
            IF(KOUNT .GE. 500)CALL ERRPOR(IER)
            SAVEK(KOUNT) = K1
            SAVEMU(KOUNT) = MU
      C
      C     CALCULATE THE LOWER K ESTIMATE FOR THE NEXT
      C     MU VALUE
      C     INCREMENT MU
      C
            K0 = K1 + TDELT * K1 / MU
            IF(K0 .GT. STOPK) GO TO 300
            MU = MU + TDELT
            GO TO 201
      C
      C
        400 CONTINUE
            LMU = SAVEMU(KOUNT)
            ENDFILE 2
            REWIND 2
            CALL PLOTER(NPTS,1,KMAX)
      C
      C
      C     POSITION THE FILE TO THE BEGINNING OF
      C     THE FIRST MODE
      C
            READ(2,95)R,X0,STOPK,EPS,ND,STOPMU
         95 FORMAT(4E15.7,2I5//)
            PRINT PARAM
            ISTOP = ND -1
            READ(2,94) (N(I), I = 1, ISTOP)
         94 FORMAT(8F10.5)
            DO 6000 J = 1, STOPMU
      C
      C
      C     NOW TO CALCULATE OMEGA FOR EACH MODE AND
      C     PLOT OMEGA VS K
      C
      C
      C     USE ARRAY SAVEMU TO HOLD THE OMEGA VALUES
      C
            READ(2, 99)NPTS
            DO 5000 I = 1, NPTS
            READ(2, 100) SAVEK(I), MU
            SAVEMU(I) = SAVEK(I) / MU
         99 FORMAT(//,I5)
        100 FORMAT(2E22.7)
```

```
PROGRAM DISPER

          101 FORMAT(1X,2E22.7)
         5000 CONTINUE
              CALL PLOTER(NPTS,2, KMAX)
         6000 CONTINUE
              CALL PLOTER(NPTS,3,ERRMAX)
              CALL ENDPLOT
        C
        C     JOB IS COMPLETE
        C
              STOP
              END
```

FUNCTION XMU0

```
      FUNCTION XMU0(Y,DY)
      COMMON PSI(500),X(500)
      REAL MU
      MU = Y + DY
      CALL DIFF(MU,0.,PSIMAX)
      KOUNT = 0
 1000 CONTINUE
      TEST = PSI(1)
      KOUNT = KOUNT + 1
      IF(KOUNT .GT. 50)CALL ERRPRO(7)
      MU = MU + DY
      CALL DIFF(MU,0.,PSIMAX)
      IF(TEST * PSI(1) .GT. 0.)GO TO 1000
      XMU0 = MU
      RETURN
      END
```

SUBROUTINE GUESS

```
        *DECK GUESS
              SUBROUTINE GUESS(K0,K1,MU)
              COMMON/INPUTS/A,ND,XA,EPS,STOPMU,STOPK, STUPA
              COMMON PSI(500),N(500),NO,DELTAX,NT,ITITLE
              REAL NU,N,MU,K1,K0,K2
              IERR = 2
        C
        C
        C     THE MAXIMUM ESTIMATED ERROR FOR K IS
        C     RETURNED BY GUESS IN THE VARIABLE K0
        C
        C
        C
              IF(K0 .EQ. 0. ) EPSK = 0.
        C
        C     SOLVE DIFF EQUATION WITH MU, K0
        C     IF K0 CLOSE ENOUGH, RETURN WITH K1 = K0
        C
        C
        C
        C
        C     ATTEMPT TO FIND AN UPPER BOUND THAT IS
        C     CLOSER TO K0. THIS IS MORE EFFICIENT FOR
        C     THOSE CASES WHERE K0 IS A MUCH CLOSER ESTI-
        C     MATE THAN K1.
        C
              KOUNT = 0
              CALL DIFF(MU , K0, PSIMAX)
              IF(ABS(PSI(1) / PSIMAX) .GE. EPS)GO TO 10
              K1 = K0
        C
        C     NO ESTIMATE OF THE ERROR IN K POSSIBLE
        C
        C
              PRINT 98, K0, MU
           98 FORMAT(* NO ERROR ESTIMATE IN K POSSIBLE HERE*, / * K0 = *, F22.7,
             1     5X, * MU = *, F22.7)
              K0 = EPSK
              RETURN
           10 CONTINUE
              TEMPMX = ABS(PSI(1) / PSIMAX)
              TEMPK = K0
              TEST = PSI(1)
              STEP1 = (K1 - K0) / 10.
              K1 = K0
              DO 20 I = 1, 10
              K1 = K1 + STEP1
              CALL DIFF(MU,K1,PSIMAX)
              TERM = ABS(PSI(1) / PSIMAX)
              IF(TERM .GE. EPS) GO TO 20
              TERM1 = ABS((K1 - TEMPK) / (TERM - TEMPMX)) * EPS
              IF (TERM1 .GT. EPSK)EPSK = TERM1
              K0 = EPSK
              K1 = K1 - TERM1
              RETURN
           20 CONTINUE
              TEMPK = K1
              TEMPMX = TERM
```

```
SUBROUTINE GUESS

      IF(PSI(1) * TEST .LT. 0.)GO TO 201
  202 CONTINUE
C
C
C     ERROR, NO SIGN CHANGE
      PRINT 99,MU,K0,K1,TEST,PSI(1)
   99 FORMAT(1X,3F20.15)
      IF(K0 .NE. 0.)GO TO 250
      NT = 9
      RETURN
C
C
C
C
C
C     NOW USE THE REGULA FALSI METHOD
C
  201 CONTINUE
      KOUNT =0
      NCHEC0 = 0
      NCHEC1 = 0
  200 K2=K0 + TEST * (K1-K0) / (TEST - PSI(1))
      KOUNT = KOUNT + 1
      IF(KOUNT .GT. 20)GO TO 240
      SAVE = PSI(1)
      CALL DIFF(MU,K2,PSIMAX)
      TERM = ABS(PSI(1) / PSIMAX)
      IF (TERM .GE. EPS) GO TO 210
      TERM1 = ABS((K2 - TEMPK) /(TERM - TEMPMX)) * EPS
      IF(TERM1 .GT. EPSK) EPSK = TERM1
      K0 = EPSK
      K1 = K2 - TERM1
      RETURN
  210 CONTINUE
      TEMPK = K2
      TEMPMX = TERM
      IF (TEST * PSI(1)) 230,220,220
C
  220 CONTINUE
  226 CONTINUE
      K0 = K2
      TEST = PSI(1)
      PSI(1) = SAVE
      GO TO 200
  230 CONTINUE
  236 CONTINUE
      K1 = K2
      GO TO 200
  240 CONTINUE
      SAVE = PSI(1)
C
C     INTERVAL HALVING
C
      KOUNT = 0
  285 CONTINUE
      KOUNT = KOUNT + 1
      IF(KOUNT .GT. 54)GO TO 290
```

```
SUBROUTINE GUESS

              TEMPK = K2
              TEMPMX = TERM
              K2 = (K1 + K0)/2.
              CALL DIFF(MU, K2, PSIMAX)
              TERM = ABS(PSI(1)/PSIMAX)
              IF(TERM .LT. EPS)GO TO 300
              TERM1= ABS(K2-TEMPK)/K2
              IF(TERM1.LT. 1.E-6)GO TO 300
              IF(TEST * PSI(1) .LT. 0.)GO TO 280
      C
      C       K0 AND K2 ARE ON THE SAME SIDES
      C
              K0 = K2
              TEST = PSI(1)
              GO TO 285
          280 CONTINUE
      C
      C       K0 AND K2 ARE ON OPPOSITE SIDES
      C
              K1 = K2
              SAVE = PSI(1)
              GO TO 285
          290 CONTINUE
      C
      C       ERROR PROCESSING
      C
      C
      C       NT = 99         INDICATES OVER 50 ITERATIONS
      C                       HAVE OCCURED, BUT NO SATISFACTORY
      C                       ESTIMATE OF K REACHED
      C
              NT = 99
              RETURN
          250 CONTINUE
      C
      C       NO SIGN CHANGE HAS OCCURED. EITHER
      C       K0 IS TOO LARGE OR K1 IS TOO SMALL.
      C       THEY LIE ON THE SAME SIDE OF THE
      C       CURVE
      C
              CALL ERRPRO(8)
      C
          300 CONTINUE
      C
      C       NORMAL RETURN
      C
              TERM1 = ABS((K2 - TEMPK)/(TERM - TEMPMX)) * EPS
              IF(TERM1 .GT. EPSK)EPSK = TERM1
              K0 = EPSK
              K1 = K2 - TERM1
              RETURN
              END
```

```
*DECK MODE
      PROGRAM MODE(INPUT,OUTPUT,TAPE5)
***************************************************************
*                    A CDC 6400 PROGRAM                       *
*                                                             *
*        PROGRAMMER                                           *
*                    P. L. DRAPER                             *
*                    INSTITUTE FOR DEFENSE ANALYSIS           *
*                    ARLINGTON, VA.                           *
*                    MARCH 25, 1980                           *
*                                                             *
*        MODE CALCULATES AND GRAPHS MODE                      *
*        FUNCTIONS FOR DISPERSION CURVES                      *
*        CREATED BY COMPANION PROGRAM                         *
*        DISPER.                                              *
*        PROBLEM                                              *
*                    GIVEN                                    *
*              (1)         THE DIFFERENTIAL EQUATION          *
*                                                             *
*                    D**2 (PSI) / D(X) ** 2 =                 *
*                          (K**2 - MU**2 * N(X)**2) * PSI     *
*                                                             *
*                                                             *
*                          N0 * EXP (-(X-X0)/B)               *
*                                FOR X .GE. X0                *
*              (2)       N(X)=                                *
*                                NUMERICAL DATA               *
*                                     FOR 0 .LE. X .LE. X0    *
*              (3)       BOUNDARY CONDITION                   *
*                          PSI(0) = 0                         *
*              (4)       K                                    *
*              (5)       MU                                   *
*                    FIND THE SOLUTION TO                     *
*                    THE DIFFERENTIAL EQUATION                *
***************************************************************
C
C     INPUTS
C                    ND          INTEGER   NUMBER OF DATA POINTS
C                    B           REAL      DECAY CONSTANT
C                    X0          REAL      LAST X COORDINATE OF
C                                          NUMERICALLY DEFINED
C                                          DATA N(X)
C                    EPS         REAL      ERROR CRITERIA FOR
C                                          SOLUTION OF DIFFE-
C                                          RENTIAL EQUATION
C                    K           REAL      THE VALUE OF K OF
C                                          INTEREST
C                    STOPX       REAL      HOW FAR OUT THE X -
C                                          AXIS MODE FUNCTIONS
C                                          ARE TO BE CALCULATED
C                    N           REAL      ARRAY THAT HOLDS
C                                          THE NUMERICAL DATA
C                    DIMENSIONED 500
C                    IFIRST      INTEGER   FIRST MODE OF INTE-
C                                          REST
C                    LAST INTEGER          LAST MODE OF INTE-
C                                          REST
```

```
C
C          OUTPUT
C                    PSI        REAL ARRAY THAT HOLDS THE
C                                          MODE FUNCTION
C                    DIMENSIONED 500
C
      COMMON/INPUTS/B,ND,X0,EPS,STOPMU,STOPK, STOPX
      COMMON PSI(500),N(500),NO,DELTAX,NT,ITITLE
      REAL K, MU, K1, K0, N, MU0, MU1, N0
      INTEGER STOPMU
      NAMELIST/INPUT/B, ND, X0, EPS, IFIRST, LAST, K, STOPX
      PI = ACOS(-1.)
      IER=9
C
C
C     READ THE IDENTIFYING PARAMETERS
C     FROME TAPE5
C
C
      READ (5,96) B,X0,STOPK,EPS,ND,STOPMU
C
C
C     READ INPUT
C     K,STOPX,IFIRST,LAST
C
      READ INPUT
C
C
C     PRINT NAMELIST/INPUT/ FOR VERIFI-
C     CATION, THEN PRINT N(X)
C
      PRINT INPUT
C
      IF(K.GT. STOPK) CALL ERRPRO(IER,1)
      IF(LAST .GT. STOPMU) CALL ERRPRO(IER,2)
C
C     CALL INITIAL TO READ THE VALUES
C     OF N(X) OFF TAPE 5
C     GET N(X) DATA, AND CALCULATE DELTAX
C
C
      CALL INITIAL(SUM1)
      PRINT 101,(N(I),I=1,ND)
      PRINT 99
C
      ISTOP = ND - 1
C
      DELTAMU = PI / SUM1
      IF(IFIRST .EQ. 1)GO TO 505
C
C
C     ADVANCE POINTER TO THE
C     FIRST MODE OF INTEREST, IFIRST
C
C
      ISTOP = IFIRST - 1
      DO 500 I = 1, ISTOP
      READ(5,99)NPTS
      DO 500 J = 1, NPTS
      READ(5,100)DUMMY,DUMMY
  500 CONTINUE
  505 CONTINUE
      CALL PLOTS(100.,0)
      DO 510 I = IFIRST, LAST
C
C
C     FOR EACH OF THE MODES
C
      READ(5,99)NPTS
      K0 = 0.
      MU0 = (I + .5) * DELTAMU
      DO 520 J = 1, NPTS
C
C     READ EACK (K, OMEGA) PAIR TO FIND K INTERVAL
C
      READ(5,100)K1,MU1
      IF(K1 .GT. K)GO TO 530
      K0 = K1
      MU0 = MU1
  520 CONTINUE
  530 CONTINUE
C
C     DO LINEAR INTERPOLATION
C
      SLOPE = (K1 - K0) / (MU1 - MU0)
      YINT = K1 - SLOPE * MU1
      MU = (K - YINT) / SLOPE
      PRINT 99
      PRINT 101 ,MU,K
C
C     SOLVE DIFFERENTIAL EQUATION
C
      CALL DIFF(MU, K, PSIMAX)
      CALL INTNP(K, MU, SUM3)
      AMAXN = PSIMAX / SQRT(SUM3)
      ND = STOPX / DELTAX + 1
C
C     ERROR CONDITION
C     CHECK THAT NUM NOT GREATER THAN THE DIMENSION OF N
C
      IF(NUM .GT. 500)CALL ERRPRO(IER)
C
C     FILL IN THE REST OF PSI
C
      NT = NUM - ND
      TINDX = K * B
      ARG = MU * B * NO
      DO 560 L = 1, NT
      CALL JBESS(TINDX, 1, ARG * EXP(-(L + 1) * DELTAX / B),
     1    PSI(ND + L))
  560 CONTINUE
C
C     NORMALIZE
C
```

```
      DO 570 L = 1, NUM
      PSI(L) = PSI(L) / SQRT(SUM3)
  570 CONTINUE
C
C
C     DO PLOTTING AND OUTPUTTING HERE
C
      PRINT 98, I
      PRINT 101,(PSI(II), II = 1, NUM)
      CALL PLOTMOD(NUM,I,K,K / MU, AMAXN )
      ISTART = J + 1
CC
C     ADVANCE POINTER TO NEXT MODE
C
      DO 580 J = ISTART, NPTS
      READ(5, 100)DUMMY,DUMMY
  580 CONTINUE
  510 CONTINUE
      CALL ENDPLOT
      STOP
  101 FORMAT(1X, 4F22.7)
  100 FORMAT(2E22.7)
   99 FORMAT(//,I5)
   98 FORMAT(1H1,* MODE NUMBER *,I3)
   96 FORMAT(4E15.7,2I5//)
      END
```

SUBROUTINE DIFF

```
*DECK DIFF
      SUBROUTINE DIFF (MU,K,PSIMAX)
      COMMON/INPUTS/B,ND,X0,EPS,STOPMU,STOPK, STOPX
      COMMON PSI(500),N(500),N0,DELTAX,NT,ITITLE
      REAL NU,N,MU,K,M0
C
C
C     THIS IS THE NUMEROV - MANNING - MILLMAN METHOD
C     FOR SOLVING SECOND ORDER LINEAR DIFFERENTIAL
C     EQUATIONS
C
      F(X)=K**2 - MU**2 * Y**2
      DPSI(X,Y) = F(X)*Y
      M0=B*NU
      ISTOP = ND
C
C     DEFINE FIRST TWO VALUES OF PSI
C
      TEMP1=K*B
      TEMP2=MU*M0
      CALL JBESS (TEMP1,1,TEMP2*EXP(-DELTAX / B),PSI(1))
      CALL JBESS (TEMP1,1,TEMP2,PSI(2))
      TERM=DELTAX**2/12.
      PSIMAX = ABS(PSI(2))
C
C     SOLVE THE EQUATION BACKWARDS
C
      DO 100 I=3,ISTOP
      XNUM=2.*PSI(I-1)-PSI(I-2)
      XNUM=XNUM+DPSI(N(ND-I+4),PSI(I-2))*TERM
      XNUM=XNUM+DPSI(N(ND-I+3),PSI(I-1))*TERM*10.
      DENOM=1.-TERM*F(N(ND-I+2))
      PSI(I)=XNUM/DENOM
      IERR=LEGVAR(PSI(I))
      IF(IERR.NE.0) GO TO 900
  100 PSIMAX = AMAX1(ABS(PSI(I)),PSIMAX)
      JSTOP = ISTOP / 2
C
C     REORDER PSI
C
      DO 200 I = 1, JSTOP
      INDEX=ISTOP+1-I
      TEMP=PSI(I)
      PSI(I)=PSI(INDEX)
  200 PSI(INDEX)=TEMP
      RETURN
  900 CONTINUE
      IF(IERR .LT. 0)IERR=0
      CALL ERRPRO(IERR + 3)
      END
```

SUBROUTINE INTNP

```
*DECK INTNP
      SUBROUTINE INTNP(K,MU,SUM3)
      COMMON/INPUTS/B,ND,X0,EPS,STOPMU,STOPK, STOPX
      COMMON PSI(500),N(500),N0,DELTAX,NT,ITITLE
      COMMON /FUNC/ ARG
      REAL NU,N,MU ,K
      EXTERNAL FUNCT2, FUNCT4
      ARG = K
      SUM3 = .5*((N(1)*PSI(1))**2 + (N(ND)*PSI(NU))**2)
      TERM=MU*B*N0
      ISTOP=ND-1
      DO 100 I=2,ISTOP
  100 SUM3=SUM3+(N(I)*PSI(I))**2
      SUM3 = SUM3 * DELTAX
      TEST = (2. * K * B)**2
      TEST = ABS((TEST - 1.) * (TEST - 9.))
      TEST = TEST / 128. * 1.E3
      TEST = SQRT(TEST)
      UPPER = AMIN1(TEST, TERM)
      Z= GAUSS(4 , 0., UPPER, FUNCT2)
      IF(TERM .GT. TEST)Z = Z + GAUSS(4, TEST, TERM, FUNCT4)
      Z = Z / MU**2
      SUM3=SUM3+Z/B
      RETURN
      END
```

FUNCTION FUNCT2

```
                    FUNCTION FUNCT2(X)
                    COMMON/INPUTS/R,NO,X0,EPS,STOPMU,STOPK, STOPX
                    COMMON PSI(500),N(500),NU,DELTAX,NT,ITITLE
                    COMMON /FUNC/K
                    REAL NU,N,K
                    CALL JBESS(K*R,1.,X,BESS)
                    FUNCT2=BESS**2*X
                    RETURN
                    END
```

```
FUNCTION FUNCT4

                    FUNCTION FUNCT4(X)
                    COMMON/INPUTS/B,NO,X0,EPS,STOPMU,STOPK, STOPX
                    COMMON PST(500),N(500),NO,DELTAX,NT,ITITLE
                    COMMON /FUNCT/ K
                    REAL NO,N,K
              C
              C
              C     ASYMPTOTIC APPROXIMATION FOR J(K*B, X)**2*X
              C
              C
                    PI= ACOS(-1.)
                    TERM1=2. /PI
                    TERM2=K*B
                    ARG=X - (.5 * TERM2 + .25) * PI
                    TERM3 = (4. * TERM2 ** 2 - 1.) / (8. * X)
                    FUNCT4 = TERM1 *(COS(ARG) -TERM3 * SIN(ARG))**2
                    RETURN
                    END
```

SUBROUTINE INITIAL

```
              SUBROUTINE INITIAL (SUMI)
              COMMON/INPUTS/B,ND,X0,EPS,STOPMU,STOPK, STOPX
              COMMON PSI(500),N(500),N0,DELTAX,NT,ITITLE
              REAL N0,N
      C
      C
      C       CALCULATE DELTAX FROM X0 AND ND
      C
      C
              DELTAX = X0 / (ND - 2)
      C
      C       READ NUMERICAL VALUES
      C
              ISTOP = ND - 1.
              READ(5,450)ITITLE
          450 FORMAT(A10)
              READ(5,400) (N(I),I = 1,ISTOP)
      C
      C       CALCULATE X0+H VALUE OF N
      C
              N0 = N(ND-1)
              N(ND) = N0 * EXP(-DELTAX/H)
      C
      C       CALCULATE THE INTEGRAL OF N
      C
              SUMI= .5*(N(1)+N(ND))
              ISTOP = ND - 1
              DO 300 I = 2, ISTOP
          300 SUMI = SUMI + N(I)
              SUMI = SUMI * DELTAX
              SUMI = B * N0 + SUMI
              RETURN
      C
      C       ERROR PROCESSING
      C
          700 CALL ERRPRO(1)
              RETURN
          400 FORMAT(8F10.5)
              END
```

SUBROUTINE OUTPUTK

```
*DECK OUTPUT
      SUBROUTINE OUTPUTK (KOUNT)
      COMMON/INPUTS/R,ND,X0,EPS,STOPMU,STOPK,SLOPK
      COMMON PSI(500),N(500),N0,DELTAX,NT,ITITLE
      INTEGER STOPMU
      REAL NU, N
      COMMON /OUT/ X(500), Y(500)
      IF(KOUNT .NE. 1) GO TO 50
C
C
C     WRITE IDENTIFYING PARAMETERS TO TAPE 2
C     FOLLOWED BY THE VALUES OF N(X)
C
C
      WRITE(2,96)R,X0,STOPK,EPS,ND,STOPMU
      ISTOP = ND - 1
      WRITE(2,95) (N(I),I=1,ISTOP)
      RETURN
   50 CONTINUE
C
C
C     PRINT NUMBER OF PAIRS AND LIST
C     THE PAIRS TO OUTPUT
C
C     WRITE THE NUMBER OF PAIRS AND
C     THE PAIRS TO TAPE 2
C
C
      PRINT 98,KOUNT
      WRITE(2,99) KOUNT
      DO 10 I=1,KOUNT
      WRITE(2,100) X(I),Y(I)
   10 PRINT 101, X(I), Y(I)
      RETURN
   98 FORMAT(1H1, I5)
   96 FORMAT(4E15.7,2I5//)
   95 FORMAT(8F10.5)
  101 FORMAT(1X,2E22.7)
   99 FORMAT(//,I5)
  100 FORMAT(2E22.7)
      END
```

SUBROUTINE ERRPRO

```
                SUBROUTINE ERRPRO (N)
          C
          C     THIS ERROR PROCESSING SUBROUTINE PRESUPPOSES
          C     AN IDA TYPE ENVIRONMENT WITH CDC 6400 EQUIP-
          C     MENT, NOS/BE1 OPERATING SYSTEM
          C
          C     THE EXTERNAL REFERENCE ABRTJOB IS A ROUTINE
          C     WRITTEN IN COMPASS TO GENERATE ERROR MESSAGES,
          C     INITIATE TRACE BACK, AND ABORT THE
          C     JOB WITH OUT A DUMP
          C
                GO TO (700,800,900,902,1001,1002,1003,1004,1005,1006,1007),N
           1005 CALL ABRTJOB(48 L MORE THAN 500 PAIRS WERE NEEDED FOR THIS BRANCH)
           1006 CALL ABRTJOB(60 L THE MAXIMUM K VALUE ON TAPES IS LESS THAN THE K
               1OF INTEREST)
           1007 CALL ABRTJOB(36 L THERE ARE NOT ENOUGH MODES ON TAPES)
            700 CALL ABRTJOB(20L END-OF-FILE, UNIT 5)
            800 CALL ABRTJOB(16L KOUNT TOO LARGE    )
            900 CALL ABRTJOB( 24L INDEFINITE OPERAND 0 / 0 )
            902 CALL ABRTJOB( 17L DIVISION BY ZERO)
           1001 CALL ABRTJOB(30L CANNOT FIND MU0 FOR THIS MODE)
           1002 CALL ABRTJOB(16L TOO MANY POINTS)
           1003 CONTINUE
           1004 CALL ABRTJOB(33L K0 AND K1 ON SAME SIDE OF CURVE )
                END
```

SUBROUTINE PLOTMOD

```
              SUBROUTINE PLOTMOD(NPTS,MODE,XK,XOM,STEP)
              COMMON/INPUTS/R,NO,XO,EPS,STOPMU,STOPK, STOPA
              COMMON PSI(500),N(500),NO,DELTAX,NT
              DIMENSION DEL(4), XX(11), YY(11), XPLOT(500), YPLOT(500)
              EQUIVALENCE(PSI,YPLOT)
              DATA DEL/0., 0., 0., 1./
              DEL(4) = STEP* 2.
              CALL PSCALE(DEL(4))
              DEL(2) = STOPX / 10.
              DO 100 I=1,11
              XX(I) = (I - 1) * DEL (2)
              YY(I) = (I - 6) * DEL(4)
          100 CONTINUE
              DO 200 I=1, NPTS
              XPLOT(I) = (I - 1) * DELTAX
          200 CONTINUE
              CALL DAXIS(0.,0.,19HPSI SEC/SQRT(METER),-19,9.5,90.,YY,1.,4HF6.1,
             1   6)
              CALL PLOT(0., 5., -1)
              CALL SYMBOL (.5, 4.5, .1, 2HK=, 0., 2)
              CALL NUMBER (.8, 4.5, .1, XK, 0., 4HF4.2, 4)
              CALL SYMBOL (.5, 4., .1, 6HOMEGA=, 0., 6)
              CALL NUMBER(1.2,4.,.1,XOM,0.,4HF7.5,7)
              CALL DAXIS(0., 0.,15HDEPTH IN METERS, 15, 10., 0., XX, 1.,4HF7.2
             1     , 7)
              CALL SYMBOL(2.,-.5, .2, 11HMODE NUMBER, 0., 11)
              CALL NUMBER (4.4,-.5, .2, MODE, 0., 2HI3, 3)
              CALL LINE(XPLOT, YPLOT, NPTS, 1, 0, 0, DEL)
              CALL PLOT(12.,-5.,-1)
              RETURN
              END
```

SUBROUTINE PLOTER

```
          *DECK PLOT
                SUBROUTINE PLOTER(NPTS,M,KMAX)
                COMMON/OUT/XPLOT(500), YPLOT(500)
                COMMON/INPUTS/B,ND,X0,EPS,STOPMU,STOPK, STOPX
                COMMON PST(500),N(500),N0,DELTAX,NT,ITITLE
                DIMENSION DEL(4),XX(11),YY(11)
                REAL KMAX
                GO TO(10,20,30)M
             10 CONTINUE
                DEL(1) = 0.
                DEL(2) = STOPK / 10.
                DEL(3) = 0.
                DEL(4) = KMAX
                CALL PSCALE(DEL(4))
                DO 100 I = 1, 11
                XX(I) = (I-1) * DEL(2)
                YY(I) = (I-1) * DEL(4)
            100 CONTINUE
                CALL PLOTS(100.,0)
                CALL DAXIS(0.,0.,19HK (RADIANS / METER),19,10.,0.,XX,1.,4HF6.4,6)
                CALL SYMBOL(2.,10.,.21,34HINTERNAL WAVE DISPERSION RELATIONS,0.,
               * 34)
                CALL SYMBOL(4.,9.5,.21,ITITLE,0.,10)
                CALL DAXIS(0.,0.,5HOMEGA,-5,10.,90.,YY,1.,4HF6.4,6)
                RETURN
             20 CALL LINE(XPLOT,YPLOT,NPTS,1,0,0,DEL)
                CALL PLOT(0.,0.,-1)
                RETURN
             30 CONTINUE
                CALL SYMBOL(4.5,.5,.21,21HMAXIMUM ERROR IN K IS,0.,21)
                F=1.
                CALL WHERE(XOLD,YOLD,F)
                CALL NUMBER(XOLD,YOLD,.21,KMAX,0.,5HE10.3,10)
                RETURN
             99 FORMAT(1X,F20.10)
                END
```

SUBROUTINE PSCALE

```
      SUBROUTINE PSCALE(SCALE)
      TERM=1.
      IF(SCALE .GE. 10) GO TO 2
    1 J=SCALE*TERM
      IF(J.GT.0) GO TO 10
      TERM = TERM * 10.
      GO TO 1
    2 J=SCALE/TERM
      IF( J .LE. 10)GO TO 20
      TERM=TERM*10.
      GO TO 2
   10 SCALE=(J+.5)/(10.*TERM)
      RETURN
   20 SCALE=(J+.5)*TERM/10.
      RETURN
      END
```