REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for Public Release: Distribution Unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) NUSC TR 7943	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Underwater Systems Center 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) New London Laboratory New London, CT 06320	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING / SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. A65020 A65022 / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
FOR3D: A Computer Model for Solving the LSS Three-Dimensional, Wide Angle Wave Equation

12. PERSONAL AUTHOR(S)
George Botseas (NUSC), Ding Lee (NUSC), David King (NORDA)

13a. TYPE OF REPORT	13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
		14 August 1987	66

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Boundary conditions; Numerical methods; Range independent; Computer model; Propagation loss; Finite difference; Range dependent

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

(U) A computer model is developed for implementing the Lee-Saad-Schultz (LSS) method for solving the LSS three-dimensional wide angle wave equation. The model is designed to predict propagation loss in range-, depth-, and azimuthal-dependent ocean environments. Computational speed is favorable since the Lee-Saad-Schultz method requires only solving two tri-diagonal systems of equations for each step marched forward in range. A test problem is included for demonstrating accuracy and the capabilities of the model. The model is written in Fortran for a VAX 11/780 computer.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ OTIC USERS	UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Ding Lee	22b. TELEPHONE (Include Area Code) (203) 440-4438
	22c. OFFICE SYMBOL 3332

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

10 (Cont'd):

A65023
A63200

18 (Cont'd):

Three dimensions
Underwater acoustics
Wave equation
Wide angle

NUSC Technical Report 7943
14 August 1987

FOR3D: A Computer Model for Solving the LSS Three-Dimensional, Wide Angle Wave Equation

George Botseas
Ding Lee
Surface Ship Sonar Department

David King
Naval Ocean Research and
Development Activity
NSTL Station, MS 39529

Naval Underwater Systems Center
Newport, Rhode Island / New London, Connecticut

PREFACE

This report was prepared jointly under the following three projects:

NUSC Project No. A65020, "Numerical Solutions to Acoustic Wave Propagation," Principal Investigator, Dr. D. Lee (Code 3332). The sponsoring activity was DNL, Program Manager, G.W. Morton (Code 05), Program Element 61152N, Navy Subproject/Task ZR00000101, Inhouse Laboratory Independent Research.

NUSC Project Numbers A65022 and A65023, "New Methods in Computational Ocean Acoustics," Principal Investigator, Dr. D. Lee (Code 3332). The sponsoring activity was ONR (Grant No. N00014-86-WR-24233), Program Managers, Dr. Richard L. Lau and Robert L. Sternberg.

NUSC Project No. A63200, "Three-Dimensional IFDPE," Principal Investigator, Dr. D. Lee (Code 3332). The sponsoring activity was NORDA, Program Manager, Robert W. McGirr, Program Element 63704N.

The technical reviewer for this report was Dr. W.A. Von Winkle (Code 10).

Reviewed and Approved: 14 August 1987

W.A. Von Winkle
Associate Technical Director for Technology

The principal investigator of this report is located at the
New London Laboratory, Naval Underwater Systems Center,
New London, Connecticut 06320.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont'd)

LIST OF ILLUSTRATIONS

SECTION 1. INTRODUCTION

Considerable effort has been invested in developing methods and models for predicting acoustic wave propagation in an ocean environment in which variability in only two dimensions, range and depth, is considered. Since direction, speed, and intensity of an acoustic wave in seawater are affected by three-dimensional environmental conditions such as seamounts, fronts, and eddies, variability must be accounted for in the third dimension (azimuth). A three-dimensional parabolic equation (PE) that accounts for azimuthal variability in an ocean environment was developed by Tappert[1] during the last decade. Several years later, Baer and Perkins[2,3] introduced methods that use the fast Fourier transform to implement the three-dimensional PE. Later still, Siegmann, Kriegsmann, and Lee[4,5] (SKL) developed a three-dimensional, wide angle wave equation of which the three-dimensional parabolic approximation introduced by Tappert is a special case. Mathematical models for solving the SKL three-dimensional, wide angle wave equation were then developed by Schultz, Lee, and Jackson[6], and by Lee and Siegmann[7].

Because existing three-dimensional models are limited in capabilities and/or are slow due to the nature of the numerical methods employed, the search continues for an accurate and efficient three-dimensional model with a variety of useful capabilities.

A mathematical model recently developed by Lee, Saad, and Schultz[8] (LSS) offers an accurate and efficient approach to solving a new version of the three-dimensional, wide angle wave equation. Their approach is based on the successful application of an Implicit Finite Difference (IFD)[9,10,11] scheme for solving two-dimensional acoustic wave propagation problems. Lee, Saad, and

Schultz extended the utility of the IFD technique to include three-dimensional effects.

In this report, we develop a computer model (FOR3D) that implements the LSS method for solving the LSS version of the three-dimensional, wide angle wave equation. The model is capable of predicting acoustic propagation loss in range-, depth-, and azimuthal-dependent ocean environments. An important feature of the model is that it is also capable of accurately treating wide angle wave propagation in the vertical plane. Computational speed is favorable since the LSS method requires only solving two tri-diagonal systems of equations for each step marched forward in range.

The acronym FOR3D selected for the model represents the numerical methods employed; i.e, finite (F) differences methods, ordinary (O) differential equations, and rational (R) function approximations to solve the LSS three-dimensional (3D) wide angle wave equation.

The derivations of the SKL and LSS three-dimensional, wide angle wave equations are presented in sections 2 and 3, respectively. Section 3 also includes the mathematical formulation of the LSS method for solving the LSS three-dimensional, wide angle wave equation. Section 4 presents the finite differences methods used to employ the LSS method, while Section 5 summarizes the algorithm and computations programmed in the computer model. The structure of the computer model, including a brief description of each subroutine, is given in Section 6. Section 7 is a miniature users' guide and is devoted to input and output formats. A test problem for demonstrating accuracy and the capabilities of the model is included in Section 8.

The model is written in Fortran for a VAX 11/780 computer and has been designed such that additional capabilities may easily be incorporated. Preliminary test runs have been made on other computers, such as the VAX 8600/FPS 164 and the CRAY XMP 12.

User contributions that will enhance the model are invited. The development of the model continues.

SECTION 2. SKL THREE-DIMENSIONAL, WIDE ANGLE WAVE EQUATION

The formulation of the SKL three-dimensional, wide angle wave equation is summarized below. A complete development of the equation, including discussions on approximations and conditions, can be found in references 4 through 7.

2.1 DERIVATION

If we assume a geometry which is cylindrically symmetrical as shown in figure 1,

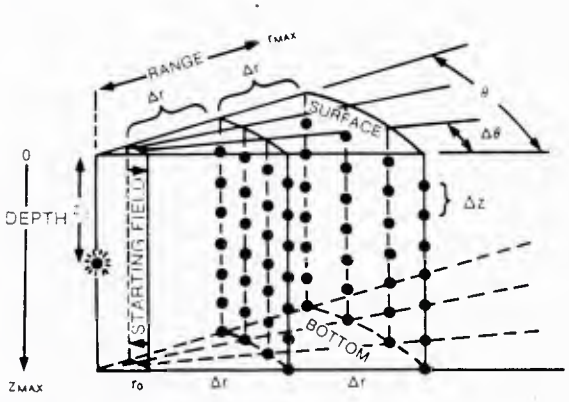


Figure 1. Geometry of Cylindrical Wedge

the three-dimensional Helmholtz equation can be written as

$$\frac{\partial^2 p}{\partial r^2} + \frac{1}{r}\frac{\partial p}{\partial r} + \frac{1}{r^2}\frac{\partial^2 p}{\partial \theta^2} + \frac{\partial^2 p}{\partial z^2} + k_0^2 n^2 p = 0. \qquad (2.1)$$

where

$p = p(r,\theta,z)$ = acoustic pressure

$n = n(r,\theta,z) = c_0/c(r,\theta,z)$ = index of refraction

c_0 = reference sound speed

$c(r,\theta,z)$ = sound speed in the ocean

$k_0 = \omega/c$ = reference wave number

$\omega = 2\pi f$

and

f = source frequency.

Using Tappert's parabolic decomposition technique, we let

$$p(r,\theta,z) = u(r,\theta,z)\, v(r), \tag{2.2}$$

in which the factor v(r) represents the rapid variation of the pressure that is modulated by $u(r,\theta,z)$. Substituting equation (2.2) into equation (2.1), we find that

$$[v_{rr} + \frac{1}{r}v_r]u + [u_{rr} + (\frac{1}{r} + \frac{2}{v}v_r)u_r + u_{zz} + \frac{1}{r^2}u_{\theta\theta} + k_0^2 n^2(r,\theta,z)u]v = 0. \tag{2.3}$$

Using k_0^2 as a separation constant, and setting the expression in the first set of square brackets in equation (2.3) equal to $-k_0^2 v$ and the expression in the second set of square brackets equal to $k_0^2 u$, we obtain

$$v_{rr} + \frac{1}{r}\, v_r + k_0^2 v = 0, \tag{2.4}$$

and

$$u_{rr} + (\frac{1}{r} + \frac{2}{v}\, v_r)\, u_r + u_{zz} + \frac{1}{r^2}\, u_{\theta\theta} + k_0^2(n^2(r,\theta,z) - 1)u = 0. \tag{2.5}$$

Equation (2.4) is the zero-order Bessel equation for which the solution for v in terms of out-going waves is given by the zeroth-order Hankel function of the first kind

$$v(r) = H_0^{(1)}(k_0 r) \tag{2.6}$$

Introducing the far-field approximation

$$k_0 r \gg 1, \tag{2.7}$$

enables us to replace the Hankel function by its asymptotic expansion in equation (2.6), giving

$$v(r) = H_0^{(1)}(k_0 r) \simeq \sqrt{\frac{2}{\pi k_0 r}}\, e^{i(k_0 r - \pi/4)}. \tag{2.8}$$

Using equation (2.8) to simplify the coefficient $(1/r + (2/v)\, v_r)$ in equation (2.5), we obtain a partial differential equation, which is

$$u_{rr} + 2ik_0 u_r + u_{zz} + \frac{1}{r^2} u_{\theta\theta} + k_0^2 (n^2(r,\theta,z) - 1)u = 0. \tag{2.9}$$

Expressing equation (2.9) in operator form, we find

$$\left(\frac{\partial}{\partial r} + ik_0 - ik_0 \sqrt{1 + (n^2 - 1) + \frac{1}{k_0^2}\frac{\partial^2}{\partial z^2} + \frac{1}{(k_0 r)^2}\frac{\partial^2}{\partial \theta^2}}\right)$$

$$\times \left(\frac{\partial}{\partial r} + ik_0 + ik_0 \sqrt{1 + (n^2 - 1) + \frac{1}{k_0^2}\frac{\partial^2}{\partial z^2} + \frac{1}{(k_0 r)^2}\frac{\partial^2}{\partial \theta^2}}\right) u = 0. \tag{2.10}$$

The first operator in equation (2.10) represents the out-going wave, and the second operator represents the in-coming wave. Considering only the out-going wave, we obtain

$$\left(\frac{\partial}{\partial r} + ik_0 - ik_0 \sqrt{1 + (n^2 - 1) + \frac{1}{k_0^2}\frac{\partial^2}{\partial z^2} + \frac{1}{(k_0 r)^2}\frac{\partial^2}{\partial \theta^2}}\right) u = 0. \tag{2.11}$$

Defining the operators,

$$x = (n^2(r,\theta,z) - 1) + \frac{1}{k_0^2}\frac{\partial^2}{\partial z^2}, \tag{2.12}$$

and

$$y = \frac{1}{(k_0 r)^2}\frac{\partial^2}{\partial \theta^2}, \tag{2.13}$$

the three-dimensional out-going wave equation (2.11) may be rewritten as

$$\left(\frac{\partial}{\partial r} + ik_0 - ik_0\sqrt{1 + x + y}\right)u = 0. \tag{2.14}$$

Using a rational function approximation for the square root operator, we have

$$\sqrt{1 + x + y} = \frac{1 + p_1 x + p_2 y}{1 + q_1 x + q_2 y}, \tag{2.15}$$

where p_1, p_2, q_1, and q_2 are constants which influence the size of the propagation angle in the vertical plane. Claerbout's[12] coefficients, $p_1=p_2=3/4$ and $q_1=q_2=1/4$, ensure an approximation of the square root to second order in x and y. Substituting equation (2.15) into equation (2.14), we have the SKL wide angle, three-dimensional parabolic equation,

$$\frac{\partial}{\partial r}u = \left(-ik_0 + ik_0\frac{1 + p_1 x + p_2 y}{1 + q_1 x + q_2 y}\right)u. \tag{2.16}$$

derived by Siegmann, Kriegsmann, and Lee.

2.2. SOLUTION

Solutions for the SKL three-dimensional, wide angle wave equation can be found in references 4 through 7, and are not discussed here since this report is devoted to solving the LSS equation.

SECTION 3. LSS THREE-DIMENSIONAL, WIDE ANGLE WAVE EQUATION

In the previous section it was shown how Siegmann, Kriegsmann, and Lee derived their equation by substituting a rational function approximation for the square root operator in the one-way, out-going wave equation (2.14). By using a higher order approximation for the square root operator, Lee, Saad, and Schultz derived another version of the three-dimensional, wide angle wave equation that may then be solved by successive tri-diagonal systems in the resulting finite difference equations. Their derivation and method of solution are summarized below. For complete details, the reader is referred to reference 8.

3.1 DERIVATION

Lee, Saad, and Schultz begin their development with the three-dimensional out-going wave equation (2.14),

$$(\frac{\partial}{\partial r} + ik_0 - ik_0 \sqrt{1 + x + y})u = 0 \tag{3.1}$$

as previously derived by Siegmann, Kriegsmann, and Lee. It is at this point in the development of the three-dimensional, wide angle wave equation that the approach introduced by Lee, Saad, and Schultz varies from that of Siegmann, Kriegsmann, and Lee.

Based on the assumption that azimuthal variation is small but not negligible, Lee, Saad, and Schultz introduced the higher order approximation,

$$\sqrt{1 + x + y} \simeq (1 + \frac{1}{2}x - \frac{1}{8}x^2) + \frac{1}{2}y \tag{3.2}$$

where the terms inside the parentheses on the right-hand side of the equation give wide angle capability in the vertical (r,z) plane. Substituting equation (3.2) in (3.1), we obtain the LSS three-dimensional, wide angle wave equation,

$$\frac{\partial}{\partial r} u = (-ik_0 + ik_0 [1 + \frac{1}{2}x - \frac{1}{8}x^2 + \frac{1}{2}y])u. \tag{3.3}$$

Equation (3.3) is the equation that is solved by the computer model presented in this report.

3.2 SOLUTION

Locally, the solution of equation (3.3) can be expressed analytically by an ordinary-differential equation solution,

$$u(r + \Delta r,\theta,z) = e^{-ik_0 \Delta r} e^{ik_0(1 + \frac{1}{2}x - \frac{1}{8}x^2 + \frac{1}{2}y) \Delta r} u(r,\theta,z) \tag{3.4}$$

Assuming that $n(r,\theta,z)$ varies slowly with respect to θ, the operators x and y are nearly commutative, allowing equation (3.4) to be rewritten in the form,

$$u(r + \Delta r,\theta,z) = e^{-ik_0\Delta r} e^{ik_0(1 + \frac{1}{2}x - \frac{1}{8}x^2)\Delta r} e^{ik_0 \frac{1}{2}y\Delta r} u(r,\theta,z) \tag{3.5}$$

Setting $\delta = ik_0\Delta r$ and rewriting equation 3.5, we have the iterative scheme,

$$u(r + \Delta r) = e^{-\delta} e^{\delta(1 + \frac{1}{2}x - \frac{1}{8}x^2)} e^{\frac{\delta}{2}y} u(r) \tag{3.6}$$

developed by Lee, Saad, and Schultz, in which the exponentials are approximated by rational function approximations.

Substituting the approximations,

$$e^{\delta(1 + \frac{1}{2}x - \frac{1}{8}x^2)} \simeq e^{\delta} \left[\frac{1 + (\frac{1}{4} + \frac{\delta}{4})x}{1 + (\frac{1}{4} - \frac{\delta}{4})x}\right] \tag{3.7}$$

and

$$e^{\frac{\delta}{2}y} \simeq \left[\frac{1 + \frac{\delta}{4}y}{1 - \frac{\delta}{4}y}\right] \tag{3.8}$$

into equation (3.6) results in the final expression,

$$u(r + \Delta r) = \left[\frac{1 + (\frac{1}{4} + \frac{\delta}{4})x}{1 + (\frac{1}{4} - \frac{\delta}{4})x}\right] \left[\frac{1 + \frac{\delta}{4}y}{1 - \frac{\delta}{4}y}\right] u(r) \tag{3.9}$$

The advantage of using the approximations (3.7) and (3.8) is that, since x and y are real and small, then both,

$$[1 + (\frac{1}{4} - \frac{\delta}{4})\ x] \tag{3.10}$$

and

$$[1 - \frac{\delta}{4}\ y] \tag{3.11}$$

are non-zero, thus assuring that both,

$$|| \ e^{\delta} \frac{1 + (\frac{1}{4} + \frac{\delta}{4})x}{1 + (\frac{1}{4} - \frac{\delta}{4})x} \ || \tag{3.12}$$

and

$$|| \ \frac{1 + \frac{\delta}{4}y}{1 - \frac{\delta}{4}y} \ || \tag{3.13}$$

are unitary. This property assures the stability of equation (3.9).

In the next section, finite difference methods are used to develop a numerical solution for equation (3.9).

SECTION 4. SOLUTION TO THE LSS EQUATION BY FINITE DIFFERENCES

In the previous section, the derivation of the LSS three-dimensional, wide angle wave equation (3.3), and a method (equation (3.9)) for solving the LSS equation were presented. In this section, finite difference methods are employed to obtain a numerical solution for equation (3.9).

4.1 AN ALTERNATE FORMULA

Lee, Saad, and Schultz began by rewriting equation (3.9) in the form,

$$u^{j+1} = [1 + (\frac{1}{4} - \frac{\delta}{4})x]^{-1} [1 + (\frac{1}{4} + \frac{\delta}{4})x] [1 - \frac{\delta}{4}y]^{-1} [1 + \frac{\delta}{4}y]u^{j} \tag{4.1}$$

where u^{j} represents the acoustic field $u(r,\theta,z)$ at range r, u^{j+1} represents the acoustic field $u(r+\Delta r,\theta,z)$ at range $r+\Delta r$, and, as previously defined,

$$\delta = ik_0\Delta r, \tag{4.2}$$

$$x = (n^2(r,\theta,z) - 1) + \frac{1}{k_0^2}\frac{\partial^2}{\partial z^2} \quad \text{and} \tag{4.3}$$

$$y = \frac{1}{(k_0 r)^2}\frac{\partial^2}{\partial \theta^2} \tag{4.4}$$

By exploiting the near commutativity of x and y, Lee, Saad, and Schultz derived an alternate formula to equation (4.1),

$$[1 + (\frac{1}{4} - \frac{\delta}{4})x] [1 - \frac{\delta}{4}y]u^{j+1} = [1 + (\frac{1}{4} + \frac{\delta}{4})x] [1 + \frac{\delta}{4}y]u^{j} \tag{4.5}$$

that, for reasons discussed in reference 8, is unconditionally stable.

4.2 TWO-STEP SOLUTION

Given the acoustic field u^j at the present range r, the object is to solve for the field u^{j+1}, one step forward, at the advanced range $r+\Delta r$. This can be accomplished by solving equation (4.5) in two steps.

Writing RHS (right-hand side) for

$$[1 + (\frac{1}{4} + \frac{\delta}{4})x]\ [1 + \frac{\delta}{4}y]u^j, \tag{4.6}$$

and

$$v^{j+1} \text{ for } [1 - \frac{\delta}{4}y]u^{j+1}, \tag{4.7}$$

the two-step marching process is as follows:

Step 1

Compute RHS and solve for v^{j+1} in the following system of equations:

$$[1 + (\frac{1}{4} - \frac{\delta}{4})x]v^{j+1} = \text{RHS}. \tag{4.8}$$

Step 2

Now that v^{j+1} is known, solve for the unknown u^{j+1} in the following system of equations:

$$[1 - \frac{\delta}{4}y]u^{j+1} = v^{j+1}. \tag{4.9}$$

The distinct advantage of this two-step procedure is that only two tri-diagonal systems need to be solved for each step marched forward in range. Consequent-

ly, less memory and fewer computations are required than were for previous methods.

4.3 DISCRETIZATION OF OPERATORS

Before the two-step procedure previously explained can be performed, the operators x and y are discretized by central differences. In equation (4.8) of step 1, the x operator is replaced by its discrete analogue as follows:

Writing equation (4.8),

$$[1 + (\frac{1}{4} - \frac{\delta}{4})x]v^{j+1} = RHS, \tag{4.10}$$

and substituting $(n^2(r,\theta,z)-1) + \frac{1}{k_0^2}\frac{\partial^2}{\partial z^2}$ for the operator x, we have

$$(1 + (\frac{1}{4} - \frac{\delta}{4})[n^2(r,\theta,z) - 1 + \frac{1}{k_0^2}\frac{\partial^2}{\partial z^2}])v^{j+1} = RHS \tag{4.11}$$

Replacing $\frac{\partial^2 v^{j+1}}{\partial z^2}$ by its discrete analogue,

$$\frac{\partial^2 v^{j+1}}{\partial z^2} = \frac{1}{(\Delta z)^2}\left(v^{j+1}_{m+1,\ell} - 2v^{j+1}_{m,\ell} + v^{j+1}_{m-1,\ell} \right), \tag{4.12}$$

and rewriting equation (4.11), we have

$$(1 + (\frac{1}{4} - \frac{\delta}{4})\ (n^2(r,\theta_\ell,z_m)-1))\ v^{j+1}_{m,\ell} +$$

$$+ (\frac{1}{4} - \frac{\delta}{4})\ \frac{1}{k_0^2(\Delta z)^2}\ (v^{j+1}_{m+1,\ell} - 2v^{j+1}_{m,\ell} + v^{j+1}_{m-1,\ell}) = RHS \tag{4.13}$$

where Δz is the depth increment, and j, m, and ℓ are, respectively, indices in range, depth, and azimuth. Equation (4.13) may then be expressed in the matrix form,

$$A^{j+1} v^{j+1}_{m,\ell} = RHS, \tag{4.14}$$

where the tri-diagonal matrix A^{j+1} contains the diagonal elements:

$$\text{upper diagonal:} \quad (\frac{1}{4} - \frac{\delta}{4}) \frac{1}{k_0^2(\Delta z)^2} \tag{4.15}$$

$$\text{main diagonal:} \quad 1 + (\frac{1}{4} - \frac{\delta}{4})(n^2 - 1) - 2(\frac{1}{4} - \frac{\delta}{4}) \frac{1}{k_0^2(\Delta z)^2} \tag{4.16}$$

$$\text{lower diagonal:} \quad (\frac{1}{4} - \frac{\delta}{4}) \frac{1}{k_0^2(\Delta z)^2} \tag{4.17}$$

The solution vector $v^{j+1}_{m,\ell}$ contains L*M elements where m = 1,2,...,M for each ℓ = 1,2,...,L. M is the number of acoustic sensors in the vertical column, and L is the number of vertical columns as varied from port to starboard in azimuth.

In equation (4.9) of step 2, the y operator is replaced by its discrete analogue as follows:

Writing equation (4.9),

$$[1 - \frac{\delta}{4} y] u^{j+1} = v^{j+1} \tag{4.18}$$

and substituting $\frac{1}{k_0^2 r^2} \frac{\partial^2}{\partial \theta^2}$, for the operator y, we have

$$(1 - \frac{\delta}{4} (\frac{1}{(k_0 r)^2} \frac{\partial^2}{\partial \theta^2}))u^{j+1} = v^{j+1}. \tag{4.19}$$

Replacing $\frac{\partial^2 u^{j+1}}{\partial \theta^2}$ by its discrete analogue,

$$\frac{\partial^2 u^{j+1}}{\partial \theta^2} = \frac{1}{(\Delta\theta)^2} (u^{j+1}_{m,\ell+1} - 2u^{j+1}_{m,\ell} + u^{j+1}_{m,\ell-1}), \tag{4.20}$$

and rewriting equation (4.19), we have

$$u^{j+1}_{m,\ell} - \frac{\delta}{4} \frac{1}{(k_0 r)^2} \frac{1}{(\Delta\theta)^2} (u^{j+1}_{m,\ell+1} - 2u^{j+1}_{m,\ell} + u^{j+1}_{m,\ell-1}) = v^{j+1}_{m,\ell} \tag{4.21}$$

where $\Delta\theta$ is the azimuthal increment, and j, m, and ℓ are, respectively, indices in range, depth, and azimuth. Equation (4.21) may then be expressed as a system of equations in the matrix form,

$$B^{j+1} u^{j+1}_{m,\ell} = v^{j+1}_{m,\ell} \tag{4.22}$$

where the tri-diagonal matrix B^{j+1} contains the diagonal elements:

$$\text{upper diagonal:} \quad - \frac{\delta}{4} \frac{1}{k_0^2 r^2} \frac{1}{(\Delta\theta)^2} \tag{4.23}$$

$$\text{main diagonal:} \quad 1 + \frac{\delta}{2} \frac{1}{k_0^2 r^2} \frac{1}{(\Delta\theta)^2} \tag{4.24}$$

$$\text{lower diagonal:} \quad - \frac{\delta}{4} \frac{1}{k_0^2 r^2} \frac{1}{(\Delta\theta)^2} \tag{4.25}$$

The solution vector $u^{j+1}_{m,\ell}$ contains L*M elements where m = 1,2,...,M for each ℓ = 1,2,...,L. M is the number of acoustic sensors in the vertical column, and L

is the number of vertical columns as varied from port to starboard in azimuth. Solution vector $u^{j+1}_{m,\ell}$ represents the acoustic field at $(j+1)*\Delta r$, $m*\Delta z$, and $\ell*\Delta\theta$, i.e., range, depth, and azimuth, respectively.

Observing that the two terms in square brackets on the left-hand side of equation (4.5) are conjugates of the corresponding two terms in square brackets on the right-hand side, equation (4.5) may be expressed in the matrix form,

$$A^{j+1}B^{j+1}u^{j+1} = A^{j}B^{j}u^{j} \tag{4.26}$$

or,

$$A^{*}B^{*}u^{j+1} = A\ B\ u^{j} \tag{4.27}$$

where A^{*} and B^{*} are conjugates of A and B. Evaluating the A and B matrices midway between u^{j} and u^{j+1} at range $r+\Delta r/2$ ensures that the operators A and A^{*} and B and B^{*} are conjugates of each other.

SECTION 5. COMPUTER ALGORITHM

In the previous section, it was shown how Lee, Saad, and Schultz applied finite difference techniques to develop a method for numerically solving the LSS three-dimensional, wide angle wave equation. In this section we will discuss the algorithm that was used to implement their method into computer code.

5.1. TWO-STEP PROCEDURE IN MATRIX FORM

Equation (4.27) may be solved in two steps, as follows. Writing equation (4.27) in the matrix form

$$A^*B^*u^{j+1} = ABu^j, \tag{5.1}$$

and substituting v^{j+1} for B^*u^{j+1}, and v^j for Bu^j, we have

$$A^*v^{j+1} = Av^j \tag{5.2}$$

Now, solve as follows:

Step 1

Compute Av^j and, using a tri-diagonal solver, solve the system

$$A^*v^{j+1} = Av^j \tag{5.3}$$

for the vector v^{j+1}.

Step 2

Now that v^{j+1} is known, use a tri-diagonal solver to solve the system

$$B^*u^{j+1} = v^{j+1} \tag{5.4}$$

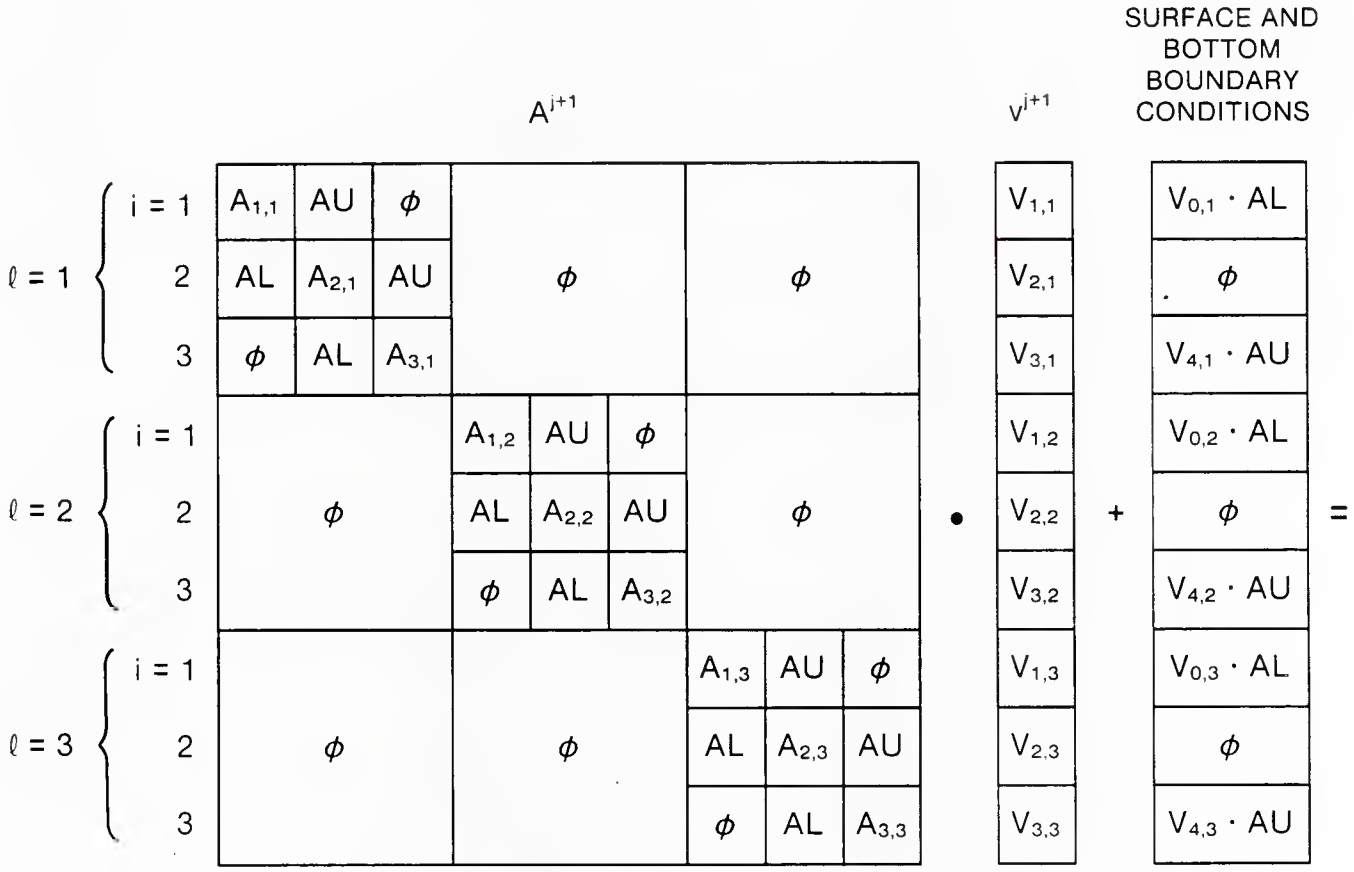
for the vector u^{j+1}. As previously stated, u^{j+1} is the acoustic field at all receivers in the cylindrical plane at the advanced range, $r+\Delta r$.

5.2 COMPUTATIONS FOR STEP 1

Figures 2 and 3 summarize the computations performed in Step 1 for a system of equations where L=3 and M=3.

5.3 COMPUTATIONS FOR STEP 2

Figure 4 summarizes the computations performed in Step 2 for a system of equations where L=3 and M=3.



ϕ = ZERO

$$AL = \left(\frac{1}{4} - \frac{\delta}{4}\right) \frac{1}{k_0^2(\Delta z)^2} = \text{lower diagonal}$$

$$AM = A_{i\ell} = 1 + \left(\frac{1}{4} - \frac{\delta}{4}\right)(n^2 - 1) - 2\left(\frac{1}{4} - \frac{\delta}{4}\right) \frac{1}{k_0^2(\Delta z)^2} = \text{main diagonal}$$

AU = upper diagonal = AL

$$\delta = ik_0\Delta r,\ n = \frac{C_0}{C_i}$$

$$k_0 = \frac{2\pi f}{C_0}$$

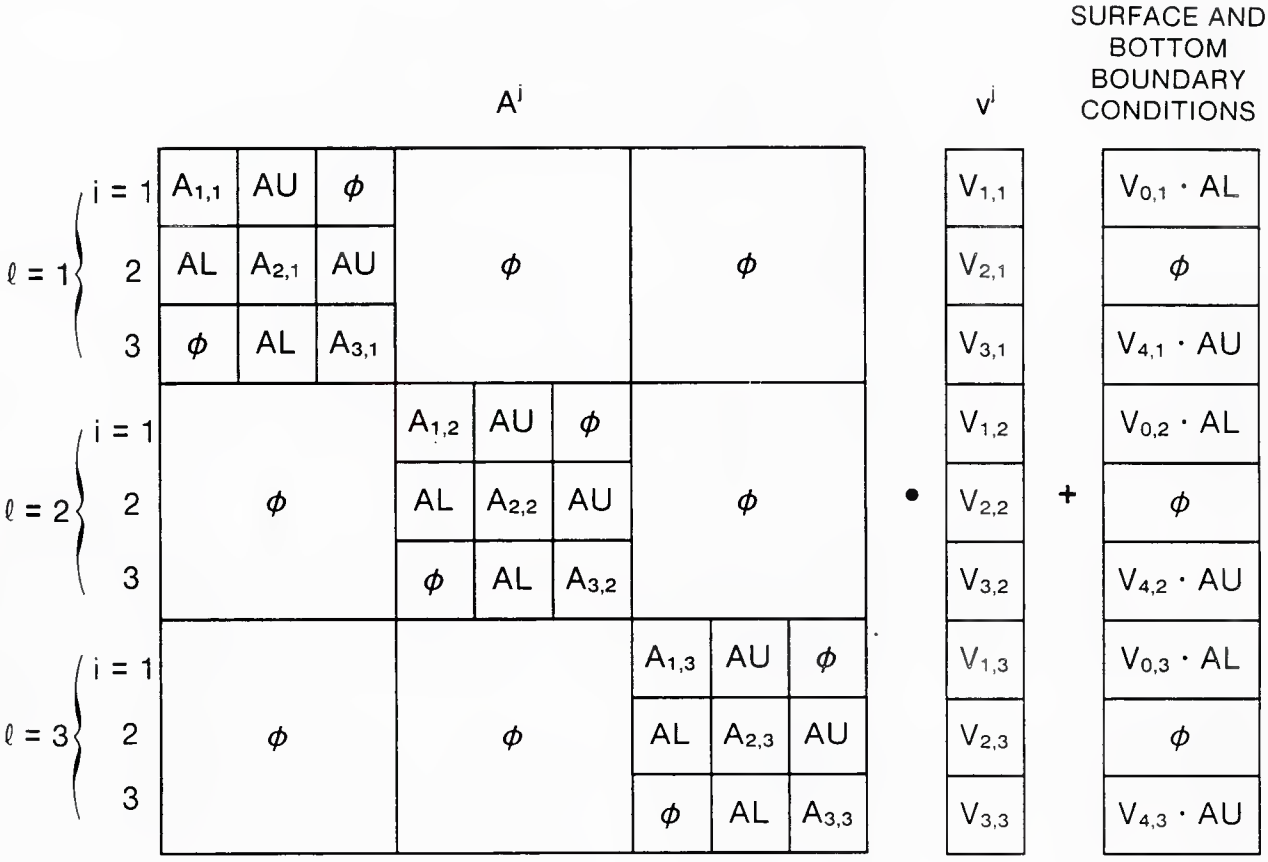
$$v_{i,\ell}^{j+1} = BL^{j+1}\, u_{i,\ell-1}^{j+1} + BM^{j+1}\, u_{i,\ell}^{j+1} + BU^{j+1}\, u_{i,\ell+1}$$

$$BL^{j+1} = -\frac{\delta}{4} \frac{1}{k_0^2 r^2} \frac{1}{(\Delta\theta)^2}$$

$$BM^{j+1} = 1 + \frac{\delta}{2} \frac{1}{k_0^2 r^2} \frac{1}{(\Delta\theta)^2}$$

$$BU^{j+1} = BL$$

Figure 2. STEP 1: Left-Hand Side of System of Equations



ϕ = ZERO

$$AL = \left(\frac{1}{4} + \frac{\delta}{4}\right) \frac{1}{k_0^2(\Delta z)^2} =$$

$$AM = A_{i,\ell} = 1 + \left(\frac{1}{4} + \frac{\delta}{4}\right)(n^2 - 1) - 2\left(\frac{1}{4} + \frac{\delta}{4}\right) \frac{1}{k_0^2(\Delta z)^2}$$

$$AU = AL$$

$$\delta = ik_0\Delta r, \; n = \frac{C_0}{C_i}$$

$$k_0 = \frac{2\pi f}{C_0}$$

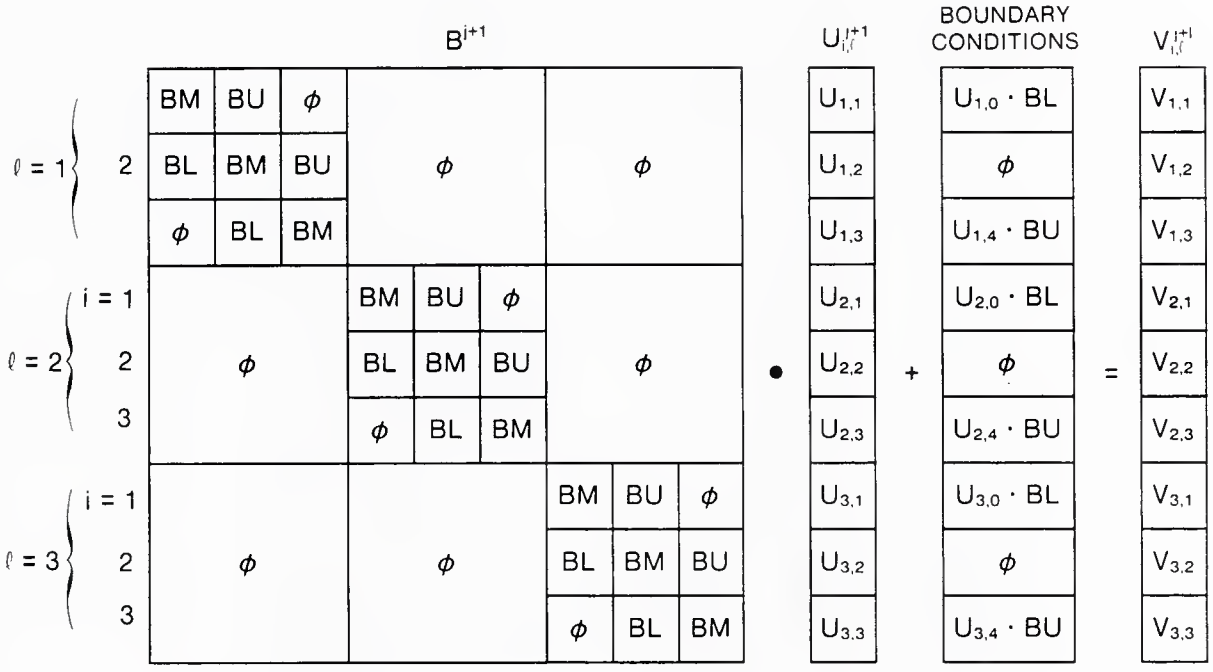
$$v^j_{i,\ell} = BL^j u^j_{i,\ell-1} + BM^j u^j_{i,\ell} + BU^j u^j_{i,\ell+1}$$

$$BL^j = \frac{\delta}{4} \frac{1}{k_0^2 r^2} \frac{1}{(\Delta\theta)^2}$$

$$BM^j = 1 - \frac{\delta}{2} \frac{1}{k_0^2 r^2} \frac{1}{(\Delta\theta)^2}$$

$$BU^j = BL^j$$

Figure 3. STEP 1: Right-Hand Side of System of Equations



ϕ = ZERO

$$BL = -\frac{\delta}{4}\frac{1}{k_0^2 r^2}\frac{1}{(\Delta\theta)^2} = \text{lower diagonal}$$

$$BM = 1 + \frac{\delta}{2}\frac{1}{k_0^2 r^2}\frac{1}{(\Delta\theta)^2} = \text{main diagonal}$$

BU = upper diagonal = BL

$\delta = ik_0\Delta r$

$$k_0 = \frac{2\pi f}{C_0}$$

$V_{i,\ell}^{i+1}$ = Computed in Step 1.

Figure 4. STEP 2: System of Equations

SECTION 6. COMPUTER MODEL

The previous section presented the two-step marching procedure for solving the LSS three-dimensional, wide angle wave equation and computations required in each of the steps. In this section, we will discuss the computer model, FOR3D, which implements the two-step procedure. FOR3D is written in Fortran using single-precision, complex arithmetic, and has been installed on a VAX-11/780 digital computer. The model was designed, in some instances at the expense of speed, such that additional capabilities may be easily incorporated. After the model has been sufficiently developed, it is anticipated that special versions of FOR3D will be generated to maximize computational speed.

6.1 CAPABILITIES

The model is capable of predicting acoustic propagation loss in range-, depth-, and azimuthal-dependent ocean environments. However, in its present stage of development, the model is only capable of handling a flat bottom. Boundary conditions are supplied by either the user or the model depending on flags set in the input runstream. Additional optional boundary conditions will be incorporated into the model as development continues. Methods for treating irregular interfaces, three-dimensional sound speed profile arrays, and other starting fields are also under consideration.

6.2 STRUCTURE OF MODEL

The hierarchical structure of the model is shown in figure 5. Those subroutines marked with an asterisk (*) are prepared by the user. Input parameters that result in control being transferred to user-supplied subroutines are shown in the figure. For example, if input parameter ISF=1, then SFLD3D transfers control to USFLD3D. If ISF=0, subroutine SFLD3D returns a Gaussian start-

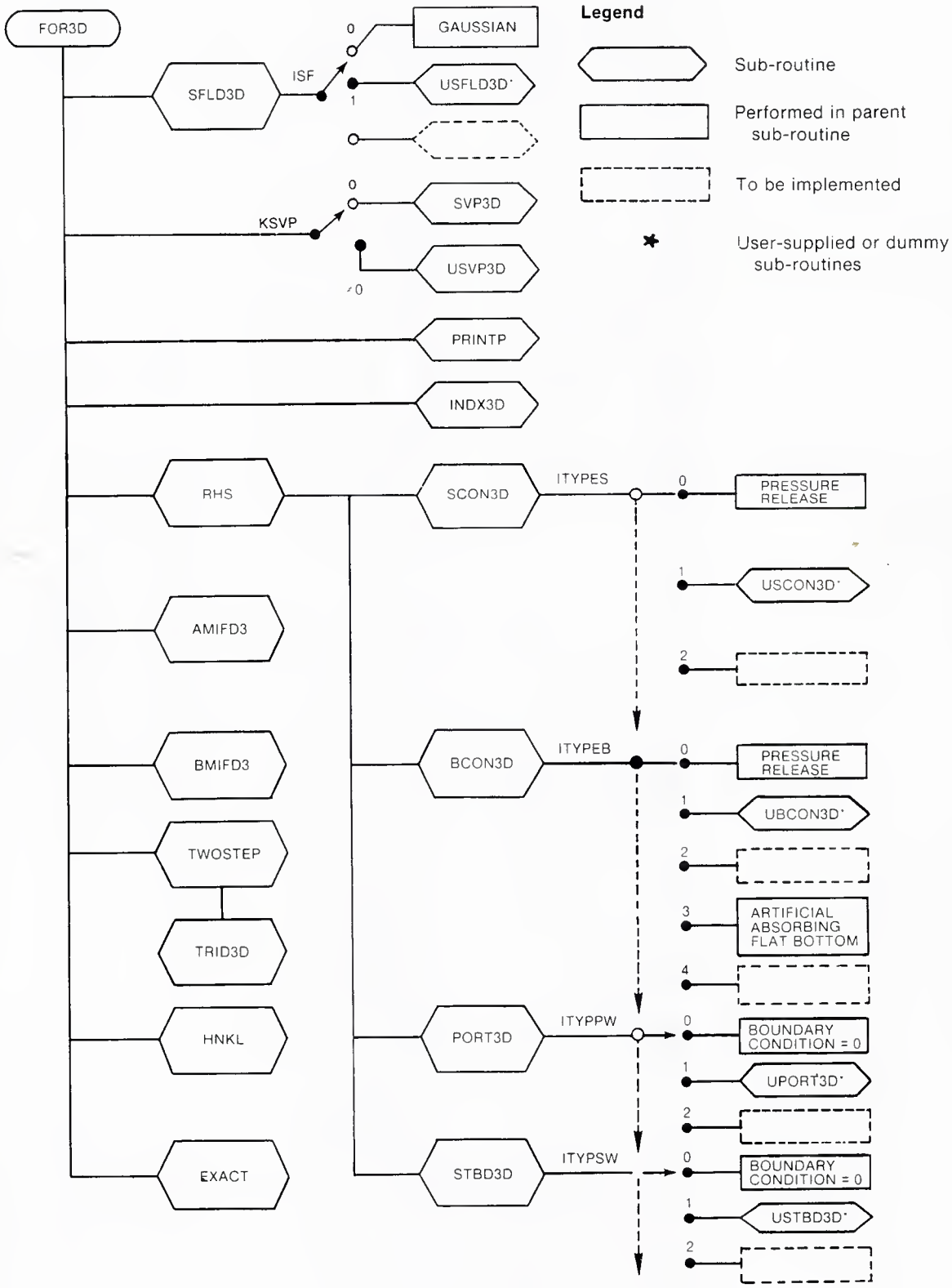


Figure 5. Hierarchical Structure of Computer Model

ing field to the main program, FOR3D. Note that the hierarchical order shown in figure 5 is not necessarily the order in which the subroutines are executed.

FOR3D can be compiled, linked, and executed by submitting the following commands:

```
$FOR  FOR3D,AMIFD3,BMIFD3,HNKL,PRINTP,TWOSTEP,RHS,TRID3D,INDX3D,SVP3D,-
      SFLD3D,SCON3D,BCON3D,PORT3D,STBD3D,USVP3D,USFLD3D,USCON3D,UBCON3D,-
      UPORT3D,USTBD3D,EXACT
$LINK FOR3D,AMIFD3,BMIFD3,HNKL,PRINTP,TWOSTEP,RHS,TRID3D,INDX3D,SVP3D,-
      SFLD3D,SCON3D,BCON3D,PORT3D,STBD3D,USVP3D,USFLD3D,USCON3D,UBCON3D,-
      UPORT3D,USTBD3D,EXACT
$RUN  FOR3D
```

6.3 GEOMETRY OF PROPAGATION

To facilitate subsequent discussions on the purpose of each sub-routine shown in figure 5, the geometry through which the acoustic wave propagates and the data structures in which that wave is contained are presented in this section.

Figure 6 illustrates the top view of a cylindrical wedge through which an acoustic wave is propagating from the cylindrical plane y to the cylindrical plane x.

Figure 7 shows a three-dimensional view of one sector of the wedge illustrated in figure 6.

Given the acoustic field along the cylindrical plane y at the present range (r), the purpose of the model is to predict the acoustic field at the cylindrical plane x at the advanced range, $r+\Delta r$. Having solved for the field at range $r+\Delta r$, the model can then solve for the field at $r+2\Delta r$. In this manner, the model marches forward, one step at a time, until maximum range has been reached.

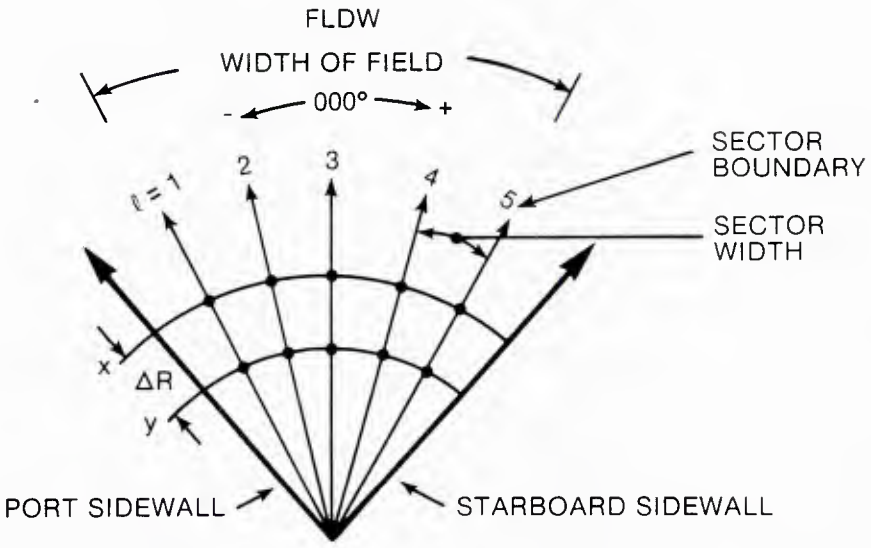


Figure 6. Top View of Cylindrical Wedge

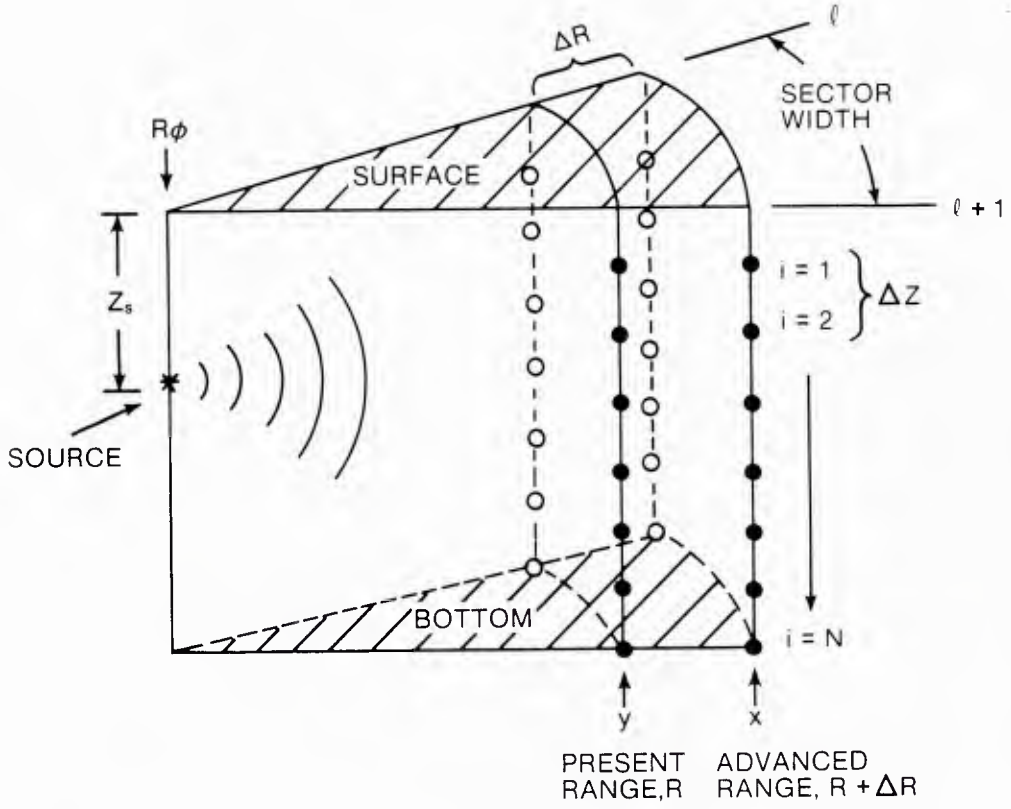


Figure 7. 3D View of One Sector

6.4 DATA STRUCTURES OF THE ACOUSTIC FIELD

The outwardly marching acoustic field along the cylindrical plane y, as viewed from the source, is as shown in figure 8. Note that the left subscript varies in depth from top to bottom while the right subscript varies in azimuth from port to starboard.

The data structures in which the acoustic field is stored are shown in figure 9.

Although the subscript N is used in both figures to point to the bottom, FOR3D will, depending on the type of bottom, use the subscript N+1 to point to the bottom condition.

6.5 MAIN PROGRAM FOR3D

The main program, FOR3D, controls the execution of the various subroutines that make up the model. Initially, FOR3D reads input parameters and environmental data from input file, FOR3D.IN, and performs required initialization. Depending on input parameter ISF, FOR3D then calls on either subroutine SFLD3D or subroutine USFLD3D to generate the starting field that is to be marched out in range. Selected problem parameters are then printed and, if requested, written in an output file for subsequent use. FOR3D then calls on subroutine INDX3D to compute an index of refraction table that is to be used in the computation of the main diagonals of the tri-diagonal matrices A and B.

After these preliminary procedures have been accomplished, FOR3D enters the program's main loop and continues to cycle in the loop until the solution has been marched out to maximum range, as requested.

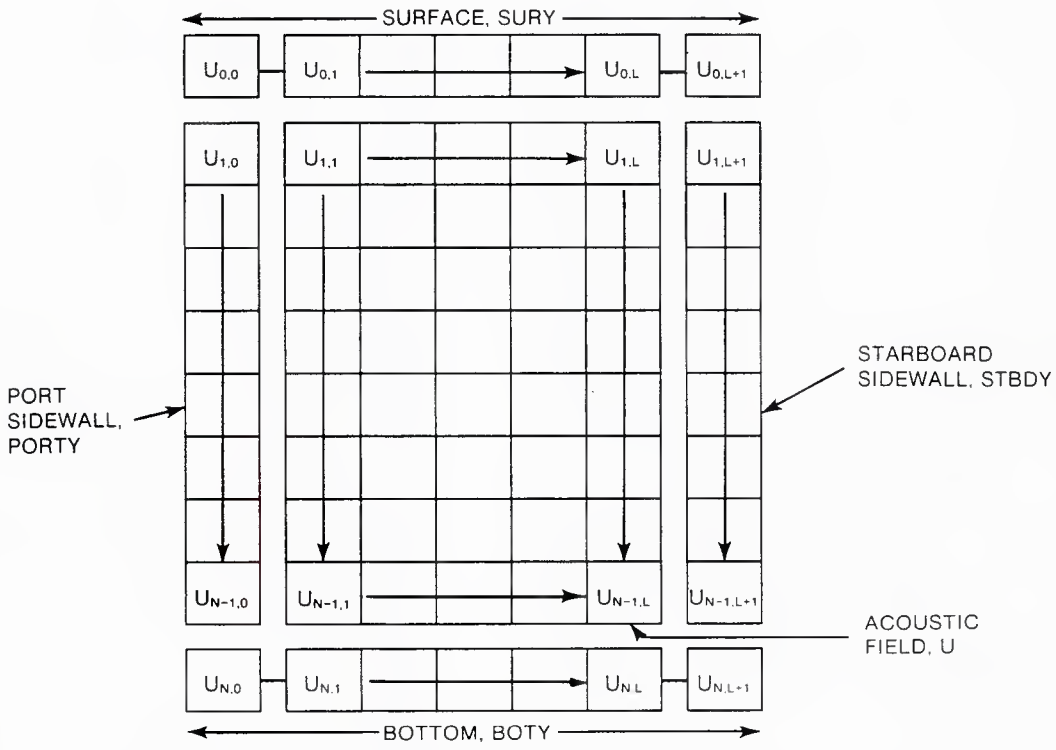


Figure 8. Pictorial View of Outwardly Propagating Wave

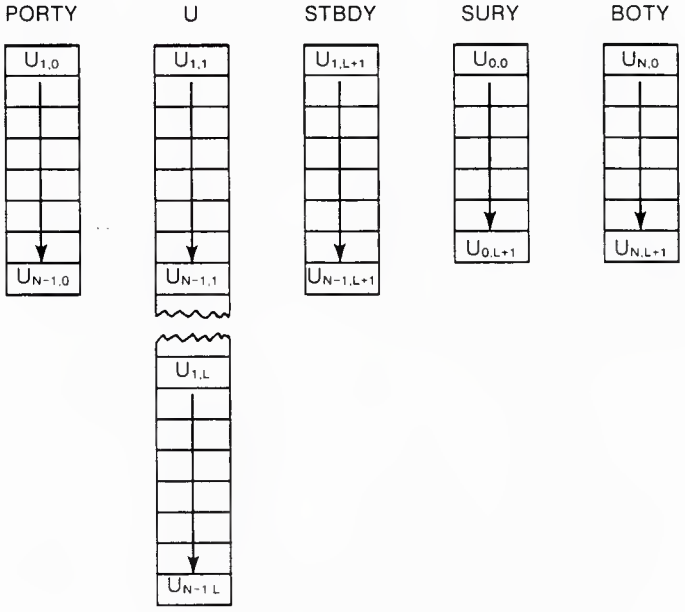


Figure 9. Data Structures of Acoustic Field

At the beginning of each pass through the loop, FOR3D updates environmental parameters, as required. FOR3D then calls on subroutine RHS to compute the right-hand side of the system of equations to be solved. Subroutines INDX3D, AMIFD3, and BMIFD3 are then called to compute the index of refraction table and the diagonals in the A and B matrices, respectively. FOR3D then calls on subroutine TWOSTEP to march the solution one range step (Δr) forward. The solution returned by TWOSTEP is then printed and written in output file, FOR3D.OUT, as requested by the user. When the solution has been marched out to maximum range, the program is terminated. If the range of the solution has not reached maximum range, FOR3D returns control to the top of the main loop and repeats the above procedures. The flow-chart in figure 10 summarizes these procedures.

6.6 SUBROUTINES

A brief description of each of the subroutines that make up FOR3D is presented in this section.

6.6.1 Subroutine SFLD3D

If input parameter ISF=0, main program FOR3D calls on subroutine SFLD3D to generate a Gaussian starting field at range zero. The Gaussian starting field[1,13] is

$$U(M+I) = CMPLX(PR,0.0),$$

where

M = (J-1)*N
J = 1,2,...,NSOL
I = 1,2,...,N
N = number of equispaced points in the vertical column
NSOL = number of columns in azimuth

$$PR = GA\left[e^{-\left(\frac{ZM-ZS}{GW}\right)2} - e^{-\left(\frac{-ZM-ZS}{GW}\right)2}\right]$$

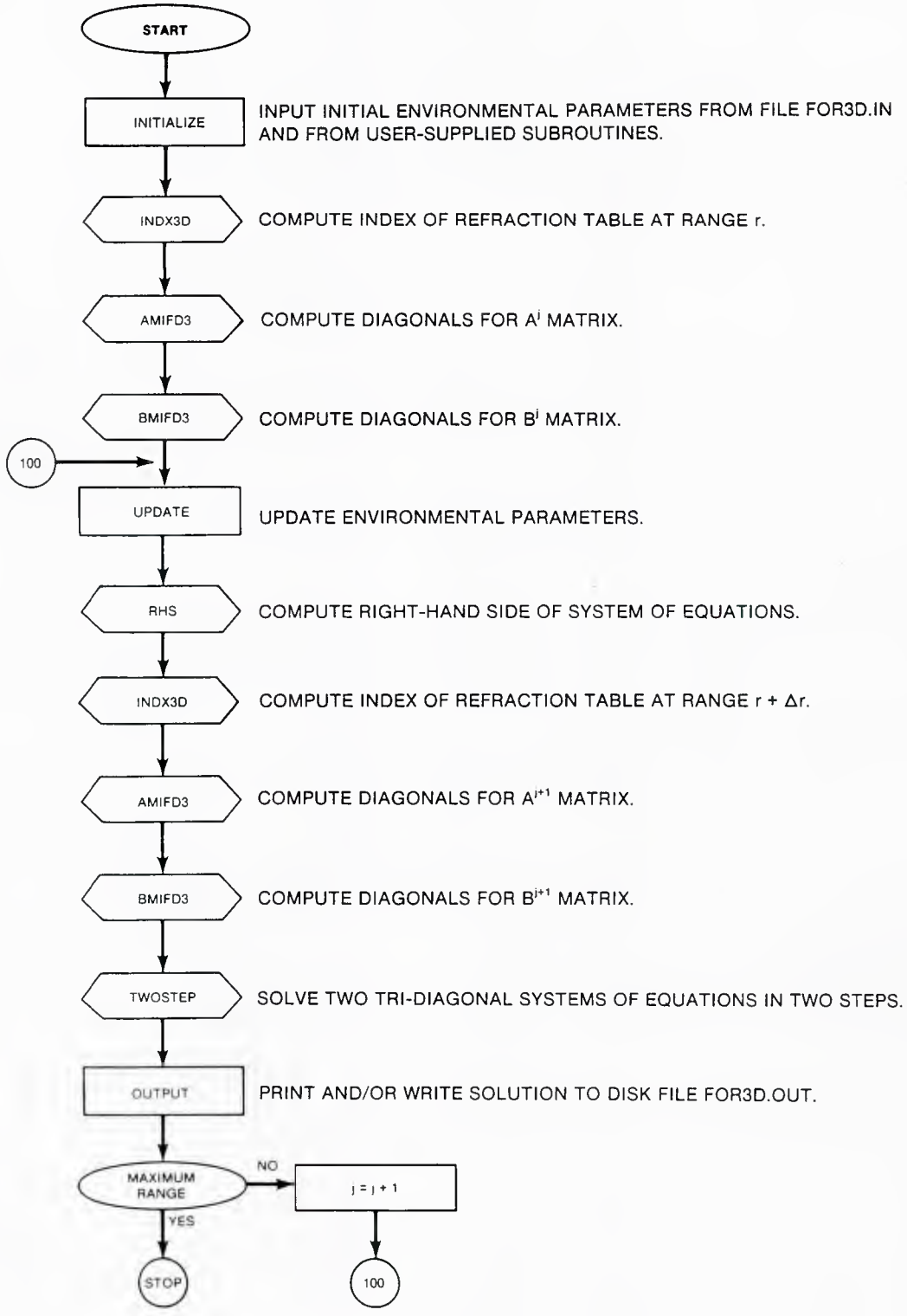


Figure 10. Flow Chart of Computer Model

ZM = depth of point = I*ΔZ (meters)
ΔZ = depth increment (meters)
ZS = source depth (meters)
GW = Gaussian width = 2/FK
FK = reference wavenumber = 2*pi*FRQ/C0
FRQ = frequency (hz)
C0 = reference sound speed (meters/sec)
GA = Gaussian amplitude = (1/GW)*(2/FK)**0.5

6.6.2 Subroutine UFLD3D

If input parameter ISF=1, main program FOR3D calls on user-written subroutine UFLD3D to generate the starting field. UFLD3D must load N*NSOL values of of the complex starting field in array U.

If ISF=0, UFLD3D is not called and may be a dummy subroutine.

If the user-supplied starting field is an elliptic solution, then the starting field must be divided by the Hankel function. The solution field obtained must then be multiplied by the Hankel function before computing transmission loss. This procedure can be accomplished automatically by setting input parameter IHNK = 1.

6.6.3 Subroutine SVP3D

When the range (RA) of the next solution to be obtained is equal to the range of the next set of sound speed profiles (RSVP), the next set of profiles is input. If input parameter KSVP=0, then subroutine SVP3D is called to read the profiles from the input runstream. SVP3D then loads the six arrays shown in figure 11.

Array ZLYR contains the maximum depth of each layer as measured from the surface. Arrays RHO and BETA contain density and attenuation, respectively, in each layer. Variable NLYR is the number of layers in the profile. The profile

in each layer is stored in arrays ZSVP and CSVP. NSVP is the total number of points in the profile. If an error is detected while inputting the profiles, NSVP is reset to zero. Array IXSVP contains indices that point to the last sound speed in each layer.

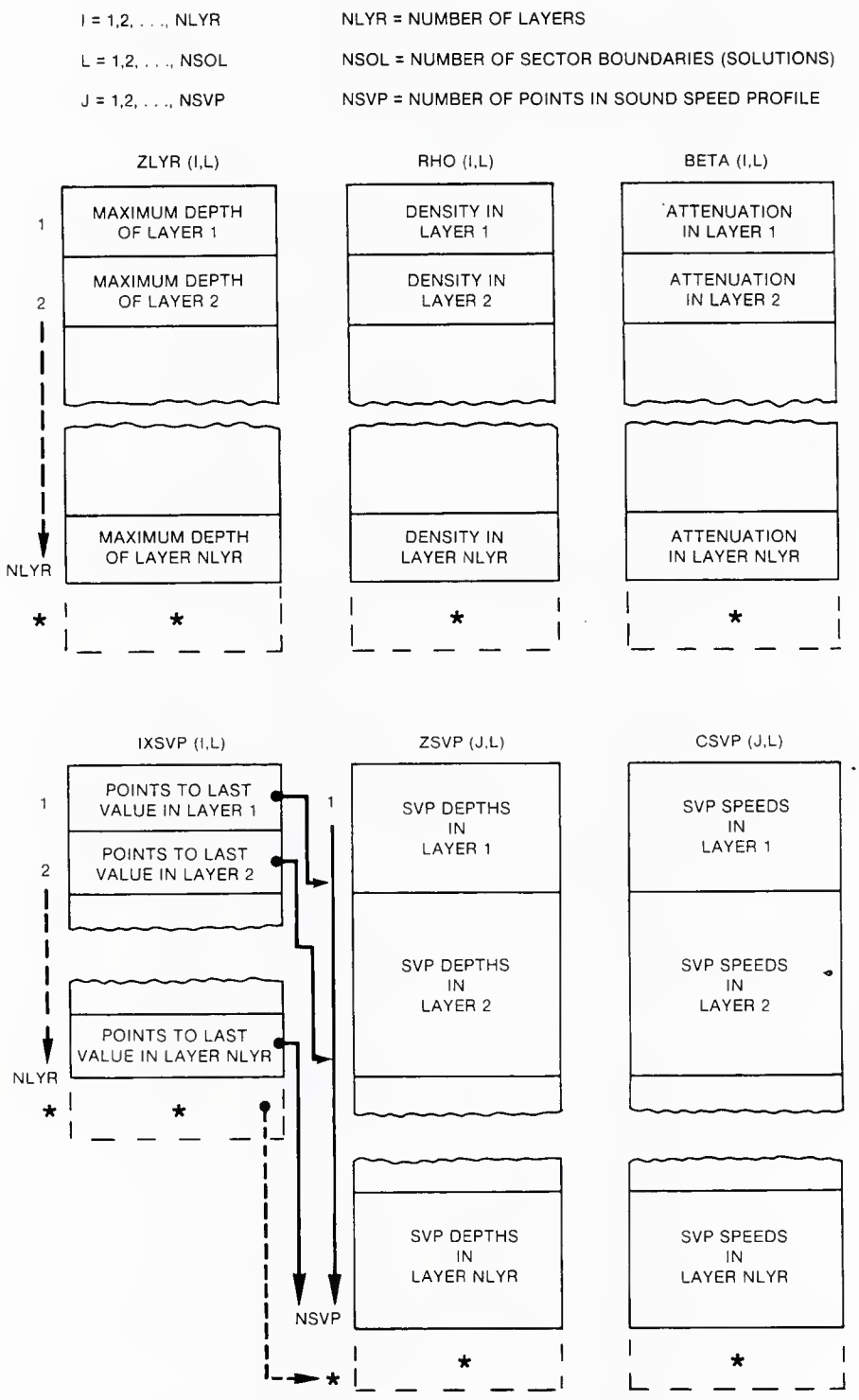
At the present stage of development, interpolation is performed only in depth. Profile changes in range are abrupt, and NSOL profiles, one for each sector boundary, must be provided since interpolation in azimuth is not performed, either. Linear interpolation in depth is performed in subroutine INDX3D.

Plans for the future include the development of a sound speed profile pre-processor which will perform interpolation in all three dimensions. Given a three-dimensional array of sound speed profiles, this pre-processor will generate profiles as required by FOR3D as it marches through a three-dimensional wedge.

6.6.4 Subroutine USVP3D

If input parameter KSVP $\neq$ 0, then subroutine USVP3D will be called to supply an updated set of sound speed profiles. If KSVP is not reset to zero by USVP3D, then USVP3D will be called at each step in range. User-prepared subroutine USVP3D must load the six arrays with NSOL profiles as described in the previous section. Variables NLYR and NSVP must also be loaded by the user.

Variable KSVP may be used in a computed GOTO statement to transfer control within user program USVP3D. When the user no longer needs USVP3D, KSVP must be reset to zero within USVP3D. The last profile entered will be used until the solution range is equal to the next RSVP. If KSVP is not reset to zero, then USVP3D will be called until the solution range is equal to RSVP, the range of



* EXTENDED IF USER REQUESTS AN ARTIFICIAL ABSORBING LAYER.

Figure 11. Sound Speed Profile Arrays

the next set of NSOL profiles. With this option, the user can generate a new set of profiles at each range step. Sound speed profile values interpolated in all three dimensions may be entered by the user in this manner. When the next solution range is equal to the next RSVP, either SVP3D or USVP3D, depending on the next KSVP, is called to input the next set of profiles.

6.6.5 Subroutine PRINTP

Subroutine PRINTP prints selected input parameters.

6.6.6 Subroutine INDX3D

Subroutine INDX3D determines the values of sound speed (C_i) and attenuation (β) to be used at the receiver depth under consideration. The index of refraction squared (n^2) is then calculated where $n = C_0/C_i$, and C_0 and C_i are, respectively, a reference sound speed and the speed of sound at the depth under consideration. Linear interpolation in depth for C_i is performed as required. The index of refraction squared is then modified as follows,

$$n^2 \simeq (C_0/C_i)^2 + i(C_0/C_i)^2 * \beta/27.287527,$$

where

β = attenuation in dB/wavelength, and

27.287527 = pi*20*ln(e).

Adding the imaginary term to the index of refraction squared accounts for attenuation.[1,14] The new value of n^2 is used in subsequent calculations of the main diagonals in the A and A^* matrices.

6.6.7 Subroutine RHS

The purpose of subroutine RHS is to compute the right-hand side of the system of equations solved in STEP 1 of the two-step procedure. After calling on SCON3D, BCON3D, PORT3D, and STBD3D for surface, bottom, port, and starboard boundary conditions, RHS performs the computations summarized in figure 3. The result is stored in the array (D). If the boundary conditions at the advanced range are known, then these conditions are included in the calculation of the right-hand side. Plans for the future include handling cases where the boundary conditions at the advanced range are unknown but may be expressed as a function of the solution to be obtained. Such a case will require modifications to some of the matrix coefficients on the left-hand side of the system of equations.

6.6.8 Subroutine SCON3D

SCON3D is called upon by subroutine RHS to supply surface conditions. If input parameter, ITYPES=0, a pressure release surface condition is returned; i.e., the arrays SURY and SURX are set to zero. If ITYPES=1, SCON3D calls on USCON3D.

6.6.9 Subroutine USCON3D

If ITYPES=1, the user must write subroutine USCON3D to supply the surface condition. The values of the surface condition must be stored in arrays SURY, shown in figure 9, and in SURX, not shown. SURY is used to store the surface condition at the present range, while SURX is used to store the surface condition at the advanced range. Surface conditions, $u^{j}_{0,\ell}$ at the present range and $u^{j+1}_{0,\ell}$ at the advanced range, as shown in figures 2 and 3, are used in the compu-

tation of $v^{j}_{0,\ell}$ and $v^{j+1}_{0,\ell}$, respectively. The subscript ℓ is varied from 0 through L+1. If ITYPES $\neq$ 1, then USCON3D is not called and may be a dummy subroutine.

6.6.10 Subroutine BCON3D

BCON3D is called by subroutine RHS to supply bottom conditions. If input parameter, ITYPEB = 0, the bottom is set to zero, i.e., arrays BOTY and BOTX are set to zero. If ITYPEB = 1, then BCON3D calls on user written subroutine UBCON3D to supply the bottom condition. If ITYPEB = 3, then BCON3D supplies an artificially absorbing flat bottom.

6.6.11 Subroutine UBCON3D

User-written subroutine UBCON3D is called upon to supply the bottom condition whenever input parameter ITYPEB is set to 1. UBCON3D must load arrays BOTY, shown in figures 8 and 9, and BOTX with bottom conditions $u^{j}_{4,\ell}$ and $u^{j+1}_{4,\ell}$, at the present and advanced ranges, respectively. Bottom conditions $u^{j}_{4,\ell}$ and $u^{j+1}_{4,\ell}$ are used in the computation of $v^{j}_{4,\ell}$ and $v^{j+1}_{4,\ell}$, respectively, as shown in figures 2 and 3. Note that, in figures 8 and 9, the subscript ℓ is varied from 0 through L+1. If ITYPEB $\neq$ 1, then UBCON3D is not executed and may be a dummy subroutine.

6.6.12 Subroutine PORT3D

Subroutine RHS calls on subroutine PORT3D to supply the boundary conditions at the port sidewall. If input parameter ITYPPW = 0, PORT3D sets the arrays PORTY, shown in figures 8 and 9, and PORTX equal to zero. If ITYPPW = 1, user-written subroutine UPORT3D is called by PORT3D. Future plans include additional boundary condition options.

6.6.13 Subroutine UPORT3D

User-supplied subroutine UPORT3D is called by PORT3D to supply port sidewall boundary conditions whenever input parameter ITYPPW = 1. UPORT3D must load arrays PORTY and PORTX with the wall conditions $u^{j}_{i,0}$ and $u^{j+1}_{i,0}$ at the present and advanced ranges, respectively. Wall conditions $u^{j}_{i,0}$ and $u^{j+1}_{i,0}$ are used in the computation of $v^{j}_{i,1}$ and $v^{j+1}_{i,1}$ as shown in figures 2 and 3. Note in figures 8 and 9 that the subscript i is varied from 1 through N-1. If ITYPPW $\neq$ 1, then UPORT3D is not called and may be a dummy subroutine.

6.6.14 Subroutine STBD3D

Subroutine RHS calls on subroutine STBD3D to supply the boundary conditions at the starboard sidewall. If input parameter ITYPSW = 0, STBD3D sets the arrays STBDY, shown in figures 8 and 9, and STBDX equal to zero. If ITYPSW = 1, user-written subroutine USTBD3D is called by STBD3D. Future plans include additional boundary condition options.

6.6.15 Subroutine USTBD3D

User-supplied subroutine USTBD3D is called by STBD3D to supply starboard sidewall boundary conditions whenever input parameter ITYPSW = 1. USTBD3D must load arrays STBDY and STBDX with the wall conditions $u^{j}_{i,L+1}$ and $u^{j+1}_{i,L+1}$ at the present and advanced ranges, respectively. Wall conditions $u^{j}_{i,L+1}$ and $u^{j+1}_{i,L+1}$ are used in the computation of $v^{j}_{i,L+1}$ and $v^{j+1}_{i,L+1}$ as shown in figures 2 and 3. Note in figures 8 and 9 that the subscript i is varied from 1 through N-1. If ITYPSW $\neq$ 1, then USTBD3D is not called and may be a dummy subroutine.

6.6.16 Subroutine AMIFD3

Subroutine AMIFD3 computes the upper, lower, and main diagonals of the A matrix. Computations performed are summarized in figure 2.

6.6.17 Subroutine BMIFD3

Subroutine BMIFD3 computes the upper, lower, and main diagonals of the B matrix. Computations performed are summarized in figure 4.

6.6.18 Subroutine TWOSTEP

Subroutine TWOSTEP marches the acoustic field one step forward by calling on the tri-diagonal solver, TRID3D, twice. Given the A matrix and the right-hand side of the system of equations summarized in figures 2 and 3, the purpose of the first call to TRID3D is to compute the vector v^{j+1}. Once the vector v^{j+1} has been obtained, TWOSTEP reorders the vector as shown on the right-hand side of the system of equations shown in figure 4. Given the B matrix and the reordered vector v^{j+1}, the purpose of the second call to TRID3D is to compute the solution field u^{j+1} at the next step in range. TWOSTEP reorders the solution field u^{j+1} returned by TRID3D as shown in figure 9.

6.6.19 Subroutine TRID3D

Subroutine TRID3D, a modified version of subroutine TRIDAG presented in reference 15, solves a system of N linear simultaneous equations having a tri-diagonal coefficient matrix.

6.6.20 Complex Function HNKL

HNKL computes the Hankel function. The algorithm for computing the Hankel function is given in reference 16.

6.6.21 Subroutine EXACT

Subroutine EXACT is used for test purposes only. If the exact solution is known, subroutine EXACT may be coded to compute and print items such as relative error between the computed and exact solutions. If the exact solution is unknown, subroutine EXACT should be a dummy subroutine.

6.6.22 Common Data File

In order to facilitate further development of the model, most variables and arrays have been declared in the common file, COMMON.FOR. COMMON.FOR is included in each subroutine as required.

SECTION 7. USER'S GUIDE

The next two sections describe input and output formats in detail. A test example showing a sample input runstream and user-written subroutines can be found later in this report.

7.1 INPUT FORMAT

Prior to executing the FOR3D model, the input runstream containing problem parameters must be stored in file FOR3D.IN. File FOR3D.IN is assigned to FORTRAN unit number NIU in the main program. If the user prefers to input problem parameters on cards, then parameter NIU should be equated to the card reader unit number, and the statement which assigns file FOR3D.IN should be removed from the main program. In either case, the input runstream is prepared in free format as follows.

7.1.1 Input Runstream

```
Card   Parameters
 1     TITLE
 2     FRQ,ZS,CO,ISF,RA,ZA,N,IHNK,ITYPES,ITYPEB,ITYPPW,ITYPSW,FLDW,NSEC
 3     RMAX,DR,WDR,WDZ,PDR,PDZ,ISFLD,ISVP,IBOT
 4     U1,U2,U3,U4,U5,U6
 5     R1,Z1  **
 6     R2,Z2    *    BOTTOM PROFILE
         .      **   RANGE, WATER DEPTH (METERS)
         .      *
 N       .    **
 N+1   -1,-1
         ************************************************
 N+2   RSVP                                              *
 N+3   KSVP                                              *
         ************************************           *
 N+4   NLYR                                  *           *
         ****************************        *           **
 N+5   ZLYR(I,L),RHO(I,L),BETA(I,L)    *     **          *** NOTE
 N+6   ZSVP(1,L),CSVP(1,L)             **    *** NOTE    **    1
 N+7   ZSVP(2,L),CSVP(2,L)             ** NOTE   **   2   *
  .          .       .                 *   3   *          *
 N+M   ZSVP(J,L),CSVP(J,L) ************          *          *
                           **********************          *
                           ********************************
```

Input parameters and notes are defined in the next subsection.

7.1.2 Input Parameters

Units are in meters and meters/sec except as noted below.

Card — Parameter

1 — TITLE = Comment card. 80 characters maximum.

2 —

- FRQ = Frequency (HZ)
- ZS = Source depth
- C0 = Reference sound speed. If C0 = 0.0, C0 is set to average speed in first layer.
- ISF = Starting field flag. 0 = Gaussian. 1 = User field. If ISF = 0, RA is set to zero.
- RA = Horizontal range from source to starting field. RA is set to 0.0 if ISF = 0.
- ZA = Depth of starting field at range RA. If ZA = 0.0, ZA is set to max depth of bottom layer in first profile. If ITYPEB = 2 or 3 and ZA = 0.0, ZA is set to (4/3)*max depth of bottom layer. If ITYPEB = 2 or 3 and ZA not zero, the artificial bottom layer is extended to ZA meters provided that ZA is greater than or equal to max. depth of bottom layer in first profile.
- N = Number of equi-spaced receivers in starting field. If N = 0, N is set so that the receiver depth increment is less than or equal to 1/10 wavelength. If N is greater than MXN, N is set to MXN.
- IHNK = Hankel function flag. IHNK = 0, don't use Hankel function. IHNK = 1, divide starting field by Hankel function, then multiply the solution field by Hankel function before computing propagation loss. If starting field is Gaussian, IHNK should be set to 0. If starting field is elliptic, IHNK should be set to 1.
- ITYPES = Type of surface
  - 0 - Pressure release. SCON3D sets SURY and SURX = 0.0
  - 1 - User supplies surface condition. See subroutine USCON3D.
  - 2 - Not implemented yet.
- ITYPEB = Type of bottom
  - 0 - Pressure release. BCON3D sets BOTY and BOTX = 0.0
  - 1 - User supplies bottom condition. See subroutine UBCON3D.
  - 2 - Not implemented yet.
  - 3 - Absorbing layer introduced - flat bottom
  - 4 - Rigid bottom condition - Not implemented yet
- ITYPPW = Type of port sidewall boundary condition
  - 0 - Field along port sidewall is set to 0.0
  - 1 - User supplied. See subroutine UPORT3D.
  - 2 - Not implemented yet.
- ITYPSW = Type of starboard sidewall boundary condition
  - 0 - Field along starboard sidewall is set to 0.0
  - 1 - User supplied. See subroutine USTBD3D.
  - 2 - Not implemented yet.

Card	Parameter
	FLDW = Width of field in degrees.
	NSE = Number of sectors in field.
3	RMAX = Maximum range of solution
	DR = Range step. If DR = 0, DR is set to 1/10 wavelength. If bottom of problem is not flat, DR is recomputed so that max depth is either incremented or decremented by DZ. Solution is computed every DR meters.
	WDR = Range step at which solution is written on disk. If WDR not 0, an output disk file is assigned. WDR is rounded to nearest DR.
	WDZ = Depth increment at which solution is written on disk. Rounded to nearest DZ.
	PDR = Range step at which solution is printed. Rounded to nearest DR.
	PDZ = Depth increment at which solution is printed. Rounded to nearest DZ.
	ISFLD = 0 - don't print starting field. 1 - print starting field.
	ISVP = 0 - don't print sound velocity profile. 1 - print sound velocity profile.
	IBOT = 0 - don't print bottom depths. 1 - print bottom depths.
4	U1-U6 = User variables - real, single precision.
5	R1,Z1 = Bottom profile. First range and depth of water.
	R2,Z2 = etc.
	.
	.
N+1	-1,-1 = Marks the end of the bottom profile.
N+2	RSVP = Range of SVP. See Note 1.
N+3	KSVP = SVP flag. = 0 - SVP in input runstream. = Not zero. Profile (cards N+4 thru N+M) is supplied by user. User writes subroutine USVP3D. KSVP may be used in computed GOTO statement to transfer control in USVP3D.
N+4	NLYR = Number of layers. If ITYPEB = 2 or 3, program inserts an artificial layer and increments NLYR by 1. See Note 2.
N+5	ZLYR(I,L) = Max depth of layer I in profile.
	RHO(I,L) = Density in layer I (g/cm**3).
	BETA(I,L) = Attenuation in layer I (dB/wavelength). If BETA(I,L) is negative, attenuation is computed. See Note 3.

Card	Parameter
N+6	ZSVP(1,L) = Depth to top of layer I
	CSVP(1,L) = Speed of sound at top of layer I
	.
	.
N+M	ZSVP(J,L) = Depth to bottom of layer I
	CSVP(J,L) = Speed of sound at bottom of layer I. If only one SVP inputted, it is used through entire problem. If more than one SVP inputted, last SVP is used through remainder of problem.

Note 1 Repeat cards N+2 through N+M for each set of profiles at range RSVP. A set of profiles consists of NSOL (NSEC+1) profiles located on sector boundaries and equi-spaced from port to starboard in azimuth. NSEC is the number of sectors in the field of interest. NSEC adjacent sectors have NSOL unique boundaries.

Note 2 Repeat cards N+4 through N+M for each profile in the set. At range zero all profiles are the same. Beyond range zero, L profiles are entered from port to starboard as viewed from range R0. L is varied from 1 through NSOL. If all profiles are the same, enter -1,-1 after the last ZSVP and CSVP of the first profile. This will cause the first profile to be entered repeatedly until NSOL profiles have been entered.

Note 3 Repeat cards N+5 through N+M for each layer I in the profile. J is the number of points in layer I.

7.2 OUTPUT FORMAT

Output data generated by the FOR3D model are written on disk in file FOR3D.OUT. File FOR3D.OUT is assigned to Fortran unit number NOU in the main program. The data are written with Fortran unformatted WRITE statements as explained below.

The values of the variables and arrays listed in the following WRITE statement are written once at the beginning of the program as follows:

```
WRITE(NOU)FRQ,ZS,C0,ISF,RA,ZA,N,IHNK,ITYPES,ITYPEB,ITYPPW,ITYPSW,FLDW,
     NSEC,NSOL,RMAX,DR,WDR,WDZ,DZ,NLYR,ZLYR,RHO,BETA,U1,U2,U3,U4,U5,U6
```

These variables and arrays are defined in the previous section.

The values of the variables and arrays listed in the following WRITE statement are written at R0 and at each WDR (write range increment) thereafter.

```
WRITE(NOU)ANG,NZ,RA,WDZ,(U(M+I),I=IWZ,N,IWZ)
```

where M = (J-1)*N
J = Solution index. J varies from 1 to NSOL. Port to starboard.
N = Number of receivers in vertical column.
ANG = Relative Bearing of solution in degrees. Port is negative.
NZ = Number of receivers written on disk.
RA = Range in meters.
WDZ = Depth increment of receivers written on disk in meters.
WDZ = IWZ*DZ
DZ = Depth increment of receivers in vertical column.
U = Solution array. Complex.

Printed output is directed to printer unit number NPU.

7.3 PLOT PROGRAM

Since plotting facilities vary from one activity to another, it was decided that plotting capabilities should not be incorporated into the FOR3D model. A separate plot program has been written to plot the output of the model.

SECTION 8. TEST PROBLEM

In this section a test problem previously reported by Siegmann, Lee, and Botseas[17] is included to demonstrate the accuracy and capabilities of the model. The input runstream and the subroutines required to supply the starting field, sound speed profiles, bottom, surface, and sidewall boundary conditions are also included.

8.1 EXACT SOLUTION

The exact solution selected to demonstrate the accuracy of the model is

$$u(r,\theta,z) = \sum_{j=1}^{2} e^{i(M_j\theta - R_j(r) + R_j(r_0))} Z_j(z), \tag{8.1}$$

where

$$Z_j(z) = \alpha_j \sin(\mu_j k_0 z), \tag{8.2}$$

$$\mu_j = (N_j - 1/2)\,\pi/(k_0 H) \tag{8.3}$$

and

$$R_j(r) = \left[\frac{1}{2}\mu_j^2 k_0 r - \frac{M_j^2}{2k_0 r} \right] \left[1 - \frac{1}{4}\mu_j^2\right]^{-1} \tag{8.4}$$

In equations (8.1) through (8.4),

M_j = an integer azimuthal mode number,

α_j = a relative mode amplitude,

r = range,

z = depth,

θ = azimuthal angle,

$k_0 = 2*pi*f/C_0$,

r_0 = initial range,

N_j = integer vertical mode number,

H = depth of bottom,

C_0 = reference sound speed, and

f = source frequency.

Input parameters selected for the test case are as follows:

$M_1 = 24$, $M_2 = 36$, $N_1 = 3$, $N_2 = 5$, $\alpha_1 = 1$, $\alpha_2 = 1$,

Source frequency = 50 hz,

Source depth = 10 meters,

Initial range of starting field = 500 meters,

Depth of water = 100 meters,

Sound speed in water = 1500 m/s,

Reference sound speed = 1500 m/s,

Density in water = 1 gm/cm**3,

Attenuation in water = 0 db/wavelength,

Depth increment = .5 meters,

Range increment = 2 meters,

Maximum range = 1000 meters,

Width of field in azimuth = 10 degrees, and

Azimuthal increment = 0.1 degree.

Three-dimensional plots of exact propagation loss solutions at 20 and 60 meters in depth are shown in figures 12 and 13, respectively. Propagation loss is computed as follows:

$$PL = -20*LOG10(|u|) + 10*LOG10(r).$$

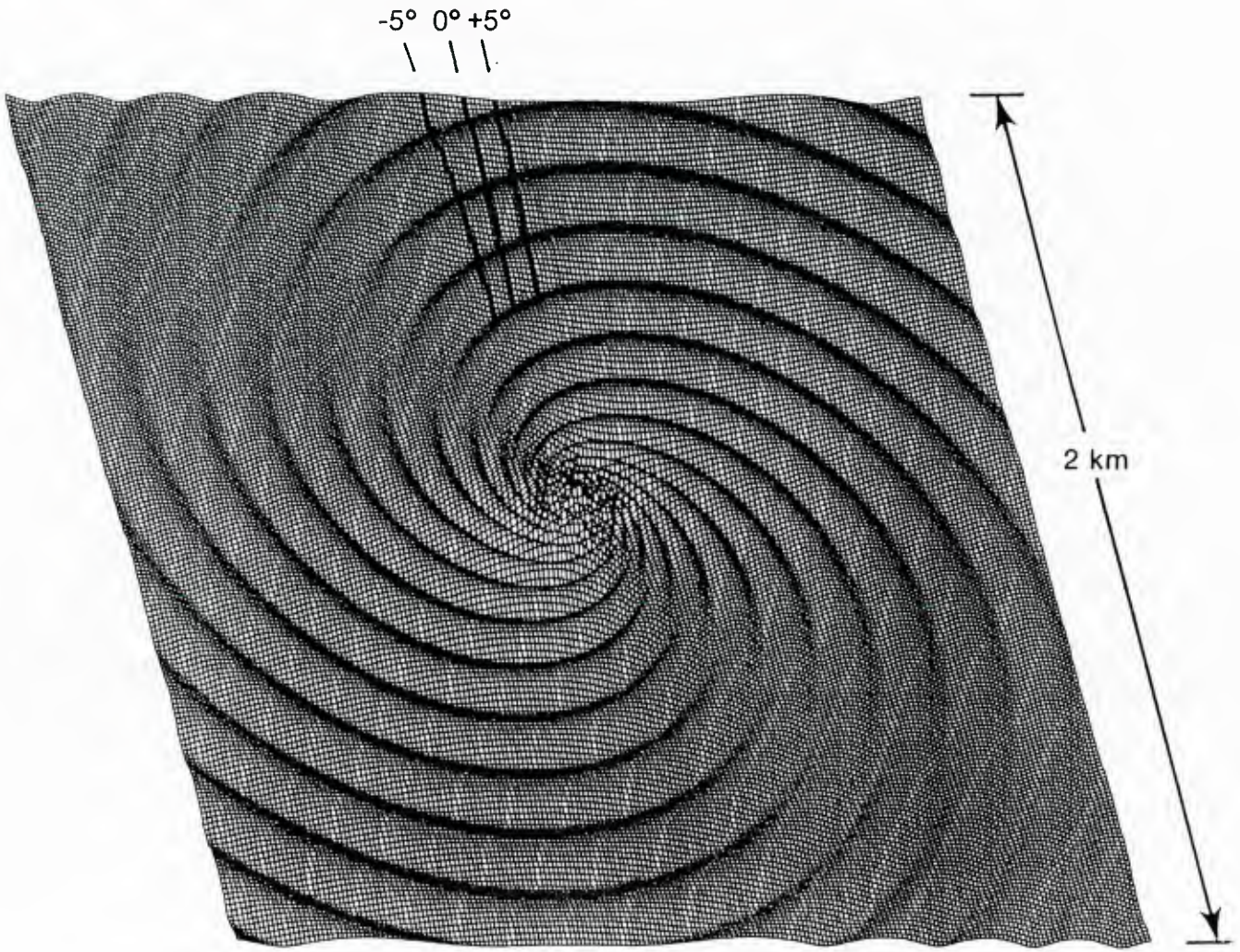


Figure 12. Exact Solution at 20 Meters in Depth

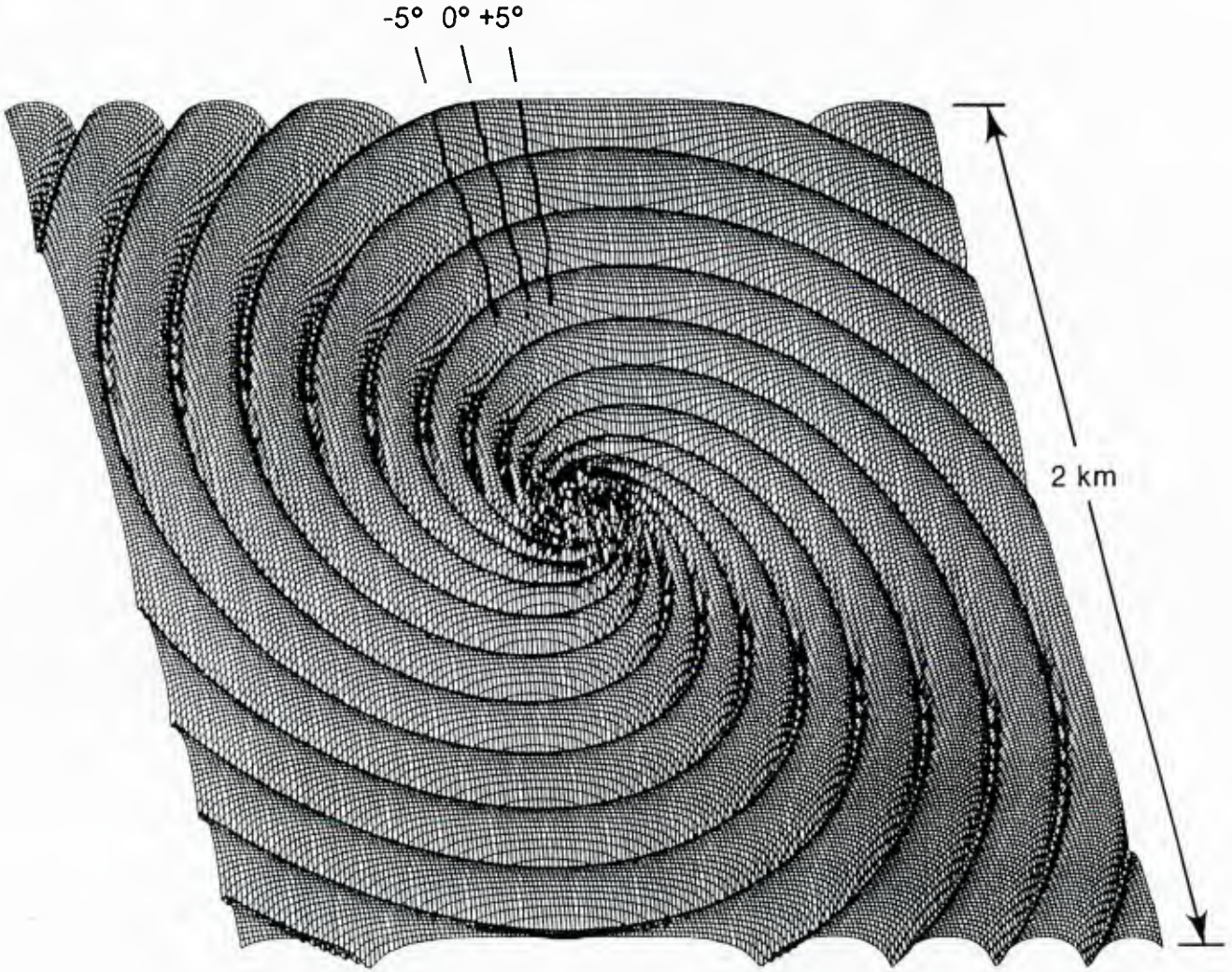


Figure 13. Exact Solution at 60 Meters in Depth

As can be seen in the figures, the exact solutions to this test case are spirals that radiate from the center and vary in all three dimensions.

8.2 COMPARISON OF RESULTS

Solutions obtained with FOR3D along relative bearing lines -5°, 0°, and +5° at receiver depths 20 and 60 meters and over the range interval 500-1000 meters as shown in figures 12 and 13 are compared with corresponding exact solutions in figure 14.

Solutions obtained with FOR3D are compared with each other in figure 15.

8.3 INPUT RUNSTREAM

The input runstream for the test case is listed below.

```
Test Case 3
50 10 1500 1 500 100 200 0 0 1 1 1 10 100
1000 2 2 10 5000 25 0 0 0
3 5 24 36 1 1
0 100
10000 100
-1,-1
0
1
```

8.4 USER-SUPPLIED SUBROUTINES

The subroutines which were supplied for this test case have been merged into one file, TEST3.FOR listed below. The listing of TEST3.FOR is followed by the command file TEST3.COM that compiled, linked, and executed FOR3D.

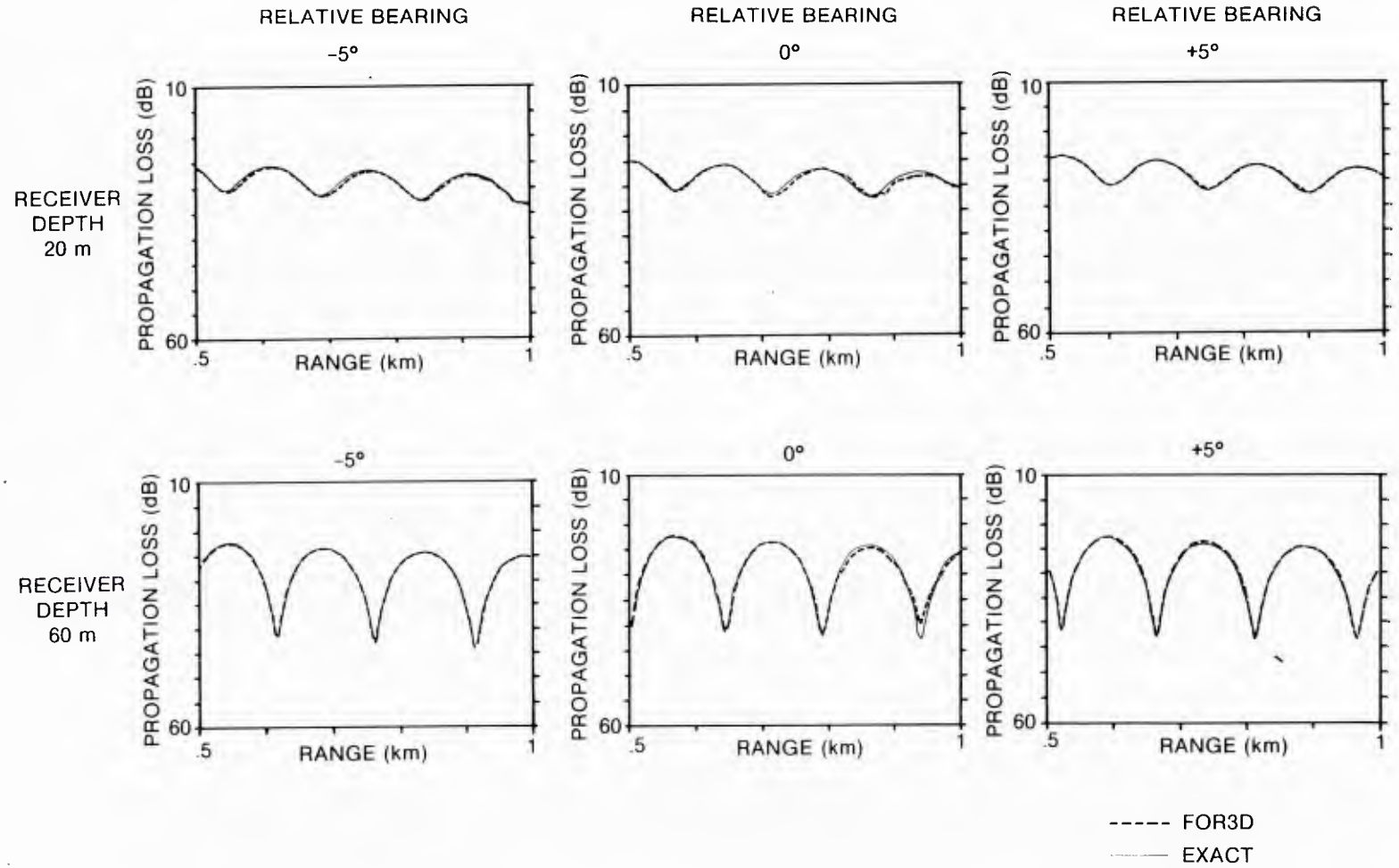


Figure 14. Comparison of FOR3D and Exact Solutions

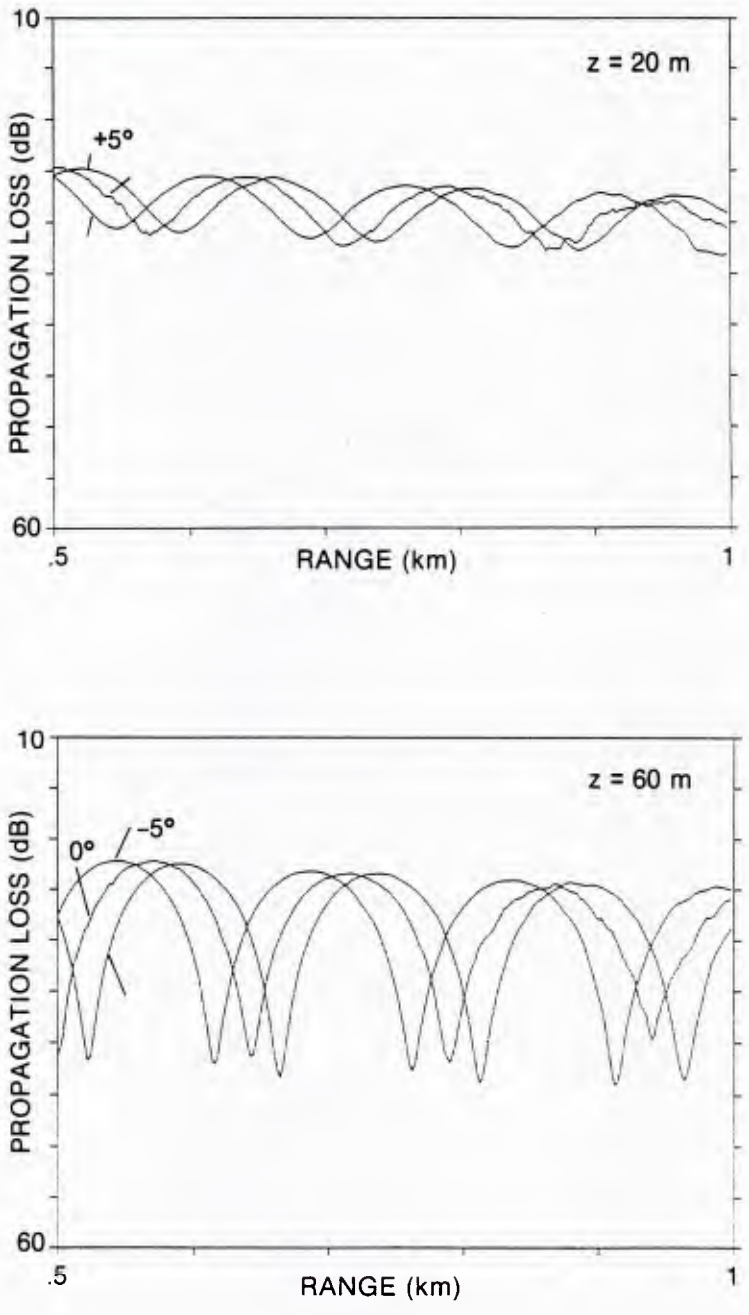


Figure 15. Comparison of FOR3D Solutions

```
      SUBROUTINE EXACT
C     ************************************************************************
C     *** EXACT SOLUTION FOR SIEGMANN TEST CASE
C     ************************************************************************
      INCLUDE 'COMMON.FOR'
      SMU1=(U1-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU1*SMU1)
      R1R0=(.5*SMU1*SMU1*XK0*R0-(U3*U3)/(2.0*XK0*R0))/TMP
      R1RA=(.5*SMU1*SMU1*XK0*RA-(U3*U3)/(2.0*XK0*RA))/TMP
      SMU2=(U2-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU2*SMU2)
      R2R0=(.5*SMU2*SMU2*XK0*R0-(U4*U4)/(2.0*XK0*R0))/TMP
      R2RA=(.5*SMU2*SMU2*XK0*RA-(U4*U4)/(2.0*XK0*RA))/TMP
      SZ1=U5*SIN(SMU1*XK0*ZI)
      SZ2=U6*SIN(SMU2*XK0*ZI)
      ANGR=ANG*PI/180.0
      UB=SZ1*CEXP(CMPLX(0.0,U3*ANGR-R1RA+R1R0))+
     C   SZ2*CEXP(CMPLX(0.0,U4*ANGR-R2RA+R2R0))
      PL=-20.0*ALOG10(CABS(UB))+10.0*ALOG10(RA)
      UC=UA-UB
      WRITE(6,120) PL,UB
      IF(ANGR.EQ.0)WRITE(6,*)RA,ZI,PL,UB
120   FORMAT(20X,3X,F10.3,3X,'(',E12.5,2X,E12.5,' )')
      RELERR=CABS(UC/UB)
      WRITE(6,150) UC,RELERR
150   FORMAT(36X,'(',E12.5,2X,E12.5,' )',2X,E12.5)
      RETURN
      END
C
C
      SUBROUTINE USVP3D
C     ************************************************************************
C     *** SOUND VELOCITY PROFILE SUBROUTINE
C     ************************************************************************
      INCLUDE 'COMMON.FOR'
      GO TO (100,200,300,400) ,KSVP
      NSVP=0
      RETURN
C
100   CONTINUE
C     *** SIEGMANN TEST CASE
      NSVP=2
      NLYR=1
      DO 75 L=1,NSOL
      ZLYR(1,L)=100.0
      RHO(1,L)=1.0
      BETA(1,L)=0.0
      ZSVP(1,L)=0.0
      CSVP(1,L)=1500.0
      ZSVP(2,L)=100.0
      CSVP(2,L)=1500.0
      IXSVP(1,L)=NSVP
75    CONTINUE
      RETURN
C
200   CONTINUE
C     *** USER INSERTS CODE HERE IF DESIRED
      RETURN
C
300   CONTINUE
```

```
C     *** USER INSERTS CODE HERE IF DESIRED
      RETURN
C
400   CONTINUE
C     *** USER INSERTS CODE HERE IF DESIRED
      RETURN
      END


      SUBROUTINE USFLD3D
C     ********************************************************************
C     *** EXACT STARTING FIELD
C     ********************************************************************
      INCLUDE 'COMMON.FOR'
      SMU1=(U1-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU1*SMU1)
      R1R0=(.5*SMU1*SMU1*XK0*R0-(U3*U3)/(2.0*XK0*R0))/TMP
      R1RA=(.5*SMU1*SMU1*XK0*RA-(U3*U3)/(2.0*XK0*RA))/TMP
      SMU2=(U2-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU2*SMU2)
      R2R0=(.5*SMU2*SMU2*XK0*R0-(U4*U4)/(2.0*XK0*R0))/TMP
      R2RA=(.5*SMU2*SMU2*XK0*RA-(U4*U4)/(2.0*XK0*RA))/TMP
      DO 100 J=1,NSOL
      M=(J-1)*N
      ANG=-FLDW/2.0*PI/180.0+((J-1)*PHI)
      DO 100 I=1,N
      ZI=I*DZ
      SZ1=U5*SIN(SMU1*XK0*ZI)
      SZ2=U6*SIN(SMU2*XK0*ZI)
      U(I+M)=SZ1*CEXP(CMPLX(0.0,U3*ANG-R1RA+R1R0))+
     C    U5*SZ2*CEXP(CMPLX(0.0,U4*ANG-R2RA+R2R0))
100   CONTINUE
      RETURN
      END
C
C
      SUBROUTINE UBCON3D
C     ********************************************************************
C     *** EXACT BOTTOM CONDITION SUBROUTINE
C     ********************************************************************
      INCLUDE 'COMMON.FOR'
      IF(THETA) 100,200,300
C
C     *** THETA LESS THAN 0.0. BOTTOM SLOPES UP.
100   CONTINUE
C      BOTY=U(N)
C      BOTX=.............
      RETURN
C
C     *** THETA = 0.0. BOTTOM IS FLAT.
200   CONTINUE
      SMU1=(U1-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU1*SMU1)
      R1R0=(.5*SMU1*SMU1*XK0*R0-(U3*U3)/(2.0*XK0*R0))/TMP
      R1RAX=(.5*SMU1*SMU1*XK0*RA-(U3*U3)/(2.0*XK0*RA))/TMP
      R1RAY=(.5*SMU1*SMU1*XK0*(RA-DR)-(U3*U3)/(2.0*XK0*(RA-DR)))/TMP
      SMU2=(U2-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU2*SMU2)
      R2R0=(.5*SMU2*SMU2*XK0*R0-(U4*U4)/(2.0*XK0*R0))/TMP
      R2RAX=(.5*SMU2*SMU2*XK0*RA-(U4*U4)/(2.0*XK0*RA))/TMP
```

```
      R2RAY=(.5*SMU2*SMU2*XK0*(RA-DR)-(U4*U4)/(2.0*XK0*(RA-DR)))/TMP
      ZI=(N+1)*DZ
      DO 250 J=1,NSOL+2
      ANG=-FLDW/2.0*PI/180.0+((J-1)*PHI)
      SZ1=U5*SIN(SMU1*XK0*ZI)
      SZ2=U6*SIN(SMU2*XK0*ZI)
      BOTX(J)=SZ1*CEXP(CMPLX(0.0,U3*ANG-R1RAX+R1R0))+
     C     U5*SZ2*CEXP(CMPLX(0.0,U4*ANG-R2RAX+R2R0))
      BOTY(J)=SZ1*CEXP(CMPLX(0.0,U3*ANG-R1RAY+R1R0))+
     C     U5*SZ2*CEXP(CMPLX(0.0,U4*ANG-R2RAY+R2R0))
250   CONTINUE
      RETURN
C
C     *** THETA GREATER THAN 0.0, BOTTOM SLOPES DOWN.
300   CONTINUE
C     BOTY=.............
C     BOTX=.............
      RETURN
      END
C
C
      SUBROUTINE UPORT3D
C     ******************************************************************
C     *** EXACT PORT BOUNDARY CONDITION
C     ******************************************************************
      INCLUDE 'COMMON.FOR'
      SMU1=(U1-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU1*SMU1)
      R1R0=(.5*SMU1*SMU1*XK0*R0-(U3*U3)/(2.0*XK0*R0))/TMP
      R1RAX=(.5*SMU1*SMU1*XK0*RA-(U3*U3)/(2.0*XK0*RA))/TMP
      R1RAY=(.5*SMU1*SMU1*XK0*(RA-DR)-(U3*U3)/(2.0*XK0*(RA-DR)))/TMP
      SMU2=(U2-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU2*SMU2)
      R2R0=(.5*SMU2*SMU2*XK0*R0-(U4*U4)/(2.0*XK0*R0))/TMP
      R2RAX=(.5*SMU2*SMU2*XK0*RA-(U4*U4)/(2.0*XK0*RA))/TMP
      R2RAY=(.5*SMU2*SMU2*XK0*(RA-DR)-(U4*U4)/(2.0*XK0*(RA-DR)))/TMP
      ANG=-FLDW/2.0*PI/180.0-PHI
      DO 250 I=1,N
      ZI=I*DZ
      SZ1=U5*SIN(SMU1*XK0*ZI)
      SZ2=U6*SIN(SMU2*XK0*ZI)
      PORTX(I)=SZ1*CEXP(CMPLX(0.0,U3*ANG-R1RAX+R1R0))+
     C     U5*SZ2*CEXP(CMPLX(0.0,U4*ANG-R2RAX+R2R0))
      PORTY(I)=SZ1*CEXP(CMPLX(0.0,U3*ANG-R1RAY+R1R0))+
     C     U5*SZ2*CEXP(CMPLX(0.0,U4*ANG-R2RAY+R2R0))
250   CONTINUE
      RETURN
      END
C
C
      SUBROUTINE USTBD3D
C     ******************************************************************
C     *** EXACT STARBOARD BOUNDARY CONDITION
C     ******************************************************************
      INCLUDE 'COMMON.FOR'
      SMU1=(U1-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU1*SMU1)
      R1R0=(.5*SMU1*SMU1*XK0*R0-(U3*U3)/(2.0*XK0*R0))/TMP
      R1RAX=(.5*SMU1*SMU1*XK0*RA-(U3*U3)/(2.0*XK0*RA))/TMP
      R1RAY=(.5*SMU1*SMU1*XK0*(RA-DR)-(U3*U3)/(2.0*XK0*(RA-DR)))/TMP
```

```
      SMU2=(U2-.5)*PI/(ZA*XK0)
      TMP=(1.0-.25*SMU2*SMU2)
      R2R0=(.5*SMU2*SMU2*XK0*R0-(U4*U4)/(2.0*XK0*R0))/TMP
      R2RAX=(.5*SMU2*SMU2*XK0*RA-(U4*U4)/(2.0*XK0*RA))/TMP
      R2RAY=(.5*SMU2*SMU2*XK0*(RA-DR)-(U4*U4)/(2.0*XK0*(RA-DR)))/TMP
      ANG=FLDW/2.0*PI/180.0+PHI
      DO 250 I=1,N
      ZI=I*DZ
      SZ1=U5*SIN(SMU1*XK0*ZI)
      SZ2=U6*SIN(SMU2*XK0*ZI)
      STBDX(I)=SZ1*CEXP(CMPLX(0.0,U3*ANG-R1RAX+R1R0))+
     C     U5*SZ2*CEXP(CMPLX(0.0,U4*ANG-R2RAX+R2R0))
      STBDY(I)=SZ1*CEXP(CMPLX(0.0,U3*ANG-R1RAY+R1R0))+
     C     U5*SZ2*CEXP(CMPLX(0.0,U4*ANG-R2RAY+R2R0))
250   CONTINUE
      RETURN
      END
```

```
! COMMAND FILE TO COMPILE, LINK AND EXECUTE TEST 3
$SET VERIFY
$ASSIGN NL SYS$PRINT
$SET DEF DRB0:[BOTSEAS.TR3D]
$FOR FOR3D,AMIFD3,BMIFD3,HNKL,PRINTP,-
TWOSTEP,RHS,TRID3D,INDX3D,-
SVP3D,SFLD3D,SCON3D,BCON3D,PORT3D,STBD3D,-
USCON3D,TEST3
$LIN FOR3D,AMIFD3,BMIFD3,HNKL,PRINTP,-
TWOSTEP,RHS,TRID3D,INDX3D,-
SVP3D,SFLD3D,SCON3D,BCON3D,PORT3D,STBD3D,-
USCON3D,TEST3
$COPY TEST3.IN FOR3D.IN
$RUN FOR3D
```

SECTION 9. CONCLUSION

A computer model, FOR3D, was developed for implementing the Lee-Saad-Schultz method for solving the LSS three-dimensional, wide angle wave equation. The model is designed to predict acoustic propagation loss in range-, depth-, and azimuthal-dependent ocean environments. The computational speed of the model is favorable since the Lee-Saad-Schultz method requires only solving two tri-diagonal systems of equations for each step marched forward in range. Theory shows that the Lee-Saad-Schultz method is unconditionally stable.

Although the model is still under development, it is presently capable of accepting arbitrary initial, surface, bottom, and sidewall boundary conditions. Another important feature of the model is that it has wide angle capability in the vertical plane. Angles of propagation as wide as 31 degrees have been accurately handled by the model.

The model is designed, in some instances at the sacrifice of speed, such that future modifications and capabilities may be incorporated easily. After the model has been sufficiently developed, it is anticipated that special versions of the model will be generated that maximize computational speed.

Plans for further development of the model include methods for treating irregular interfaces, a variety of optional starting fields, a variety of optional boundary conditions, and a rigorous treatment of a three-dimensional array of sound speed profiles. User contributions that will enhance the model are invited.

SECTION 10. REFERENCES

1. F.D. Tappert, "The Parabolic Approximation Method," J.B. Keller and J. Papadakis, ed., Wave Propagation and Underwater Acoustics, Springer Verlag, New York, 1977.

2. R.N. Baer, "Propagation through a Three-Dimensional Eddy Including Effects on an Array," Journal of the Acoustical Society of America, 69(1981), 70-75.

3. J.S. Perkins and R.N. Baer, "An Approximation to the Three-Dimensional Parabolic Equation Method for Acoustic Propagation," Journal of the Acoustical Society of America, 72(1982), 515-522.

4. W.L. Siegmann, G.A. Kriegsmann, and D. Lee, A Wide Angle Three-Dimensional Parabolic Wave Equation, P.D. Scully-Power and D. Lee, ed., NUSC Technical Document 7145, Naval Underwater Systems Center, New London, CT, 7 May 1984.

5. M.H. Schultz, D. Lee, and K.R. Jackson, Application of the Yale Sparse Technique to Solve the Three-Dimensional Parabolic Wave Equation, P.D. Scully-Power and D. Lee, ed., NUSC Technical Document 7145, Naval Underwater Systems Center, New London, CT, 7 May 1984.

6. W.L. Siegmann, G.A. Kriegsmann, and D. Lee, "A Wide Angle Three-Dimensional Parabolic Wave Equation," Journal of the Acoustical Society of America, 78(1985), 659-664.

7. D. Lee and W.L. Siegmann, "A Mathematical Model for the 3-Dimensional Ocean Sound Propagation," Mathematical Modelling, 7(1986), 143-162.

8. D. Lee, Y. Saad, and M.H. Schultz, An Efficient Method for Solving the Three-Dimensional Wide Angle Wave Equation, Research Report YALEU/DCS/RR-643, Yale University, New Haven, Ct, Oct 1986.

9. D. Lee, G. Botseas, and J.S. Papadakis, "Finite-Difference Solutions to the Parabolic Wave Equation," Journal of the Acoustical Society of America, vol. 70, no. 3, 1981, pp. 795-800.

10. D. Lee and G. Botseas, IFD: An Implicit Finite-Difference Computer Model for Solving the Parabolic Equation, NUSC Technical Report 6659, Naval Underwater Systems Center, New London, CT, 27 May 1982.

11. G. Botseas, D. Lee, and K.E. Gilbert, IFD: Wide Angle Capability, NUSC Technical Report 6905, Naval Underwater Systems Center, New London, CT, 28 Oct 1983.

12. J.F. Claerbout, Fundamentals of Geophysical Data Processing, McGraw-Hill Book Co., Inc., NY, 1976, pp. 206-207.

13. H.K. Brock, The AESD Parabolic Equation Model, NORDA Technical Note 12, Naval Ocean Research and Development Activity, NSTL Station, Mississippi, Jan 1978.

14. F. Jensen and H. Krol, The Use of the Parabolic Equation Method in Sound Propagation Modelling, SACLANTCEN Memorandum SM-72, North Atlantic Treaty Organization, SACLANT ASW Research Center, La Spezia, Italy, 15 Aug 1975.

15. B. Carnahan, H.A. Luther, and J.O. Wikes, Applied Numerical Methods, John Wiley & Sons, Inc., 1969.

16. M. Abramowitz and I. Stegun, eds., Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, Applied Mathematics, Series 55, U.S. Govt. Printing Office, Washington, D.C., 1964

17. W.L. Siegmann, D. Lee, and G. Botseas, "Finite Difference Computations of Three-Dimensional Sound Speed Propagation," D. Lee, R.L. Sternberg, and M.H. Schultz, ed., Proceedings of the First IMACS Symposium on Computational Acoustics, North Holland, 1987 (in press).

DISTRIBUTION LIST

Addressee	No. of Copies
SACLANT ASW Research Center (Dr. F.C. Jensen)	1
NRL (Dr. Ralph Baer, Code 5160; Orest Diachok; Dr. John S. Perkins, Code 5160; Dr. W.A. Kuperman; Dr. Michael Porter)	5
NOSC (Code 8302, Dr. Homer Bucker, Code 5311B; David Gordon; M.A. Pedersen,)	4
NORDA (LCDR M.A. McCallister, Dewayne White, Dr. Stanley Chin-Bing; Dr. M.F. Werby)	4
NCSC (Dr. Larry Green, Code 4120)	1
CNO (OP-952)	1
NAVSEASYSCOM (SEA-003, -06R, -63D, -63R, -63R-1, -63R-13, -53RA)	7
NAVAIRDEVCEN	1
DTIC	12
Defense Research Establishment, Pacific (Dr. Gary Brooke, Dr. N. Ross Chapman)	2
Defense Research Establishment, Atlantic (Dr. Dale D. Ellis, Dr. D. Chapman)	2
NOAA/ERL/WDL (Michael Jones)	1
NOAA/AOML/OAL (Dr. David Palmer, Code 5122; Dr. John Tsai)	2
NOO, NSTL Station (Nils Paz, Code 7332; Dr. Lloyd R. Breslau, Code 9000; Dr. William Jobst)	3
ONR, Arlington (Robert Obrochta, Dr. Richard Lau, Dr. Marshall Orr, Dr. Raymond Fitzgerald)	4
ONR (Codes 100, 200, 220, 222, 400, 422, 425AC, 480, 486)	9
OCEANAV	1
NAVOCEANO	2
Weapons System Research Laboratory (Dr. D. Kewley)	1
NAVPGSCOL (Prof. Calvin Dunlap, Code 68DU; Prof D. Odero, Code 610D; LT J.E. Jaeger)	3
Technical University, Poland (Dr. Adam Branski)	1
Georgia Institute of Technology (Prof. William F. Ames, Prof. Allan D. Pierce)	2
University of Miami (Prof. Harry DeFarrari, Prof. Frederick D. Tappert)	2
Dahlhousie University, Canada (Dr. Jules deG Gribble)	1
Colorado School of Mine (Dr. John A. DeSanto)	1
Science Applications, Inc. (Dr. Lewis Dozier, Dr. Robert Greene, Dr. John Hanna, Dr. C.W. Spofford)	4
University of Toronto, Canada (Prof. Kenneth R. Jackson)	4
DREP (Dr. N.R. Chapman)	1
ODSI Defense System Inc. (Paul Etter)	1
TRW (Dr. Frank E. Fendell)	1
APL/Johns Hopkins (Dr. Robert F. Henrick)	1
APL/University of Texas (Prof. Robert A. Koch, Susan Payne)	2
APL/Penn State (Dr. S.T. McDaniel)	12
Northwestern University (Prof. G. Kriegsmann)	4
Columbia University (Prof. John T. Kuo, Dr. Yu-Chiung Teng)	2

DISTRIBUTION LIST (Cont'd)

Addressee	No. of Copies
Admiralty Underwater Weapons Establishment (Dr. Roy G. Levers)	1
University of Rhode Island (Dr. Larry Mayer, Prof. G.R. Verma)	2
University of New Orleans (Prof. J.E. Murphy)	1
Daubin Systems Corp. (Dr. Lan Nghiem-Phu)	1
Bell Telephone Laboratory (Richard S. Patton, Denni Seals)	2
Polytechnic in Westchester (Prof. S. Preiser)	1
Prof. Morris Schulkin, Potomac, MD	1
Yale University (Prof. Martin H. Schultz)	4
Rensselaer Polytechnic Institute (Prof. W.L. Siegmann, Prof. M.J. Jacobson)	6
Polytechnic Institute of New York (Prof. Jacob Yaniv)	1
University of Masachusetts (Prof. Donald F. St. Mary)	6
University of Tennessee (Prof. V. Doogalis)	1
Rutgers University (Prof. Robert Vichnevetsky)	1
Iowa State University (Dr. James Corones)	1
FLOPETROL, SCHLUMBERGER, France	1
Forschungsanstalt der Bundeswehr fuer Wasserschall un Geophysik (Prof. G. Ziehm)	1
University of Crete, Greece (Prof. John S. Papadakis)	1
NORDA (R.W. McGirr)	24
NORDA (G. Morris)	12
NORDA (Dr. David King)	10
Harvard University (Prof. Allan R. Robinson, Dr. Wayne Leslie)	2
Massachusetts Institute of Technology (Prof. A.B. Baggeroer, Prof. M. Buckingham)	2
INO (Dr. Christopher Mooers, Dr. David Adamec, Dr. Michele Rienecker, Robert Willems)	4
University of Illinois at Urbana/Champaign (Dr. Youcef Saad)	1
The Catholic University of America (Prof. J.J. McCoy, Dr. L. Fishman)	2
The Florida State University (Prof. E.C. Young, Prof. C. Tam)	2
Michigan State University (Prof. T.Y. Li)	1