JOURNAL OF COMPUTATIONAL PHYSICS 108, 357–367 (1993)

Delta-Matrix Factorization for Fast Propagation through Solid Layers in a Fluid–Solid Medium

SVEN IVANSSON

Department of Hydroacoustics and Seismology, National Defence Research Establishment, S-172 90 Sundbyberg, Sweden

Received April 13, 1992; revised September 16, 1992

Propagator techniques for computation of wave propagation in layered fluid–solid media are well established in underwater acoustics and seismology. Delta matrices for solid layers provide a convenient way of computing the boundary values at a fluid–solid interface, with loss-of-precision control. A certain vector is propagated through a sequence of multiplications with delta matrices. We show that computations of this kind can be performed more efficiently if each delta matrix is decomposed as a product of *sparse* matrices which are applied in sequence. We propose two kinds of delta-matrix factorizations. In connection with dispersion computations, our first factorization turns out to give a method that is very much related to the powerful "fast form" of Knopoff's method. Our algorithm is slightly more efficient, however. Furthermore, the derivation gives insight into the structure of the delta-matrix propagator and other applications, such as propagation in connection with wavenumber integration for point-source studies, become immediate. Our second delta-matrix factorization gives a dispersion-function method that is significantly faster than the "fast form" of Knopoff's method. The most conspicuous gains in comparison to previous methods, however, appear for applications to wavenumber-integration techniques such as the reflectivity method. Very few arithmetic operations are needed. This factorization also provides a good basis for analyzing the numerical performance of delta-matrix propagation. It becomes quite obvious how to perform scaling so as to avoid overflow for arbitrarily thick homogeneous layers without layer splitting. In connection with multi-frequency computations distinct advantages are obtained.

1. INTRODUCTION

Laterally homogeneous fluid–solid media composed of homogeneous isotropic layers continue to provide important basic cases for studies of wave propagation in underwater acoustics and seismology. Over the years, several computational methods for such media have been proposed. Without any claim to be complete, we list a few of these methods below. They have all in common that each frequency is considered separately and that the wave-field is synthesized from its horizontal wavenumber components, each of which is the solution to a two-point boundary-value problem for an ordinary differential equation system with depth as the independent variable. We only deal with the P-SV case, omitting the much simpler SH case from the discussion.

Thomson [37] and Haskell [9] developed the propagator-matrix technique with 4×4 matrices for the solid layers. At high frequencies, however, there was a problem with numerical loss-of-precision. This problem was recognized and remedied in the sixties, e.g., by Dunkin [7] who suggested using 6×6 compound matrices or delta matrices as we will call them. Watson [38] noted that an effective reducion to 5×5 matrices could be made, this is the *reduced* delta-matrix formalism. Kind [16] developed an efficient algorithm for computing the delta-matrix propagator. Efficient multi-frequency computations were discussed by Kind and Odom [18].

A different procedure was proposed by Kennett [14, 15]. His method is based on a recursive construction scheme for 2×2 reflection and transmission matrices. The recursion can be performed in a numerically stable way.

Another line of investigation takes as its starting point the construction of a "global" equation system for the wave-field in all layers. The corresponding system matrix, the "global matrix," is assembled from 4×8 blocks, each block specifying the relations among the wave-field variables for two adjacent solid layers. The global matrix obtains a band structure and there are striking similarities to the way in which a global stiffness matrix is built up in conventional finite-element programs.

Disregardig the delta-matrix propagator technique, which can in fact be viewed as *one* particular recursive way of handling the global equation system, we may distinguish between two approaches for utilizing the global-matrix concept. The first approach aims at reconstructing the wave-field by summation of normal-mode contributions. Modal wavenumbers are calculated by finding the roots of a dispersion function which is taken as the determinant of the global matrix. Next, modal eigenfunctions are computed. This method has been developed in detail by Knopoff [19], Schwab [32], Schwab and Knopoff [33], Schwab *et al.* [34], and Panza [22]. In particular, a powerful recursive algorithm for numerically stable computation of the global-matrix determinant was given by Schwab [32]. It has been called the "fast form" of Knopoff's method and it was

0021-9991/93 $5.00

581/108/2-11

derived by elementary row and column operations on the global-matrix determinant. In comparison to a traditional delta-matrix method, a speed improvement of 38% was reported for the purely elastic case, where real arithmetic can be used.

The second global-matrix approach is concerned with actual solution of global-matrix equation systems for different horizontal wavenumbers with right-hand sides as given by the source conditions. Such an approach was suggested by Chin *et al.* [6] and independently developed by Schmidt and co-workers, e.g., [30, 29, 31, 28]. Numerical stability is achieved by using appropriately scaled variables and Gaussian elimination with partial pivoting, cf. the discussion by Keller [13] concerning numerical advantages of "parallel-shooting" methods.

Related to the global-matrix methods discussed are conventional finite-element and finite-difference techniques [3, 35, 21] and the recent "exact element" method by Karasalo [12].

In the original expressions for the delta-matrix propagator and for the recursive scheme of Knopoff's method, frequency and layer thickness appear in certain trigonometric (hyperbolic) factors. Modifications have been developed where exponential factors are used instead (cf. Euler's formulas). As a motivation it has been claimed that by separation of the different exponential factors, which may differ very much in magnitude for evanescent waves, the risk of numerical loss-of-precision may be reduced. An early paper of this kind is Randall [27], which starts from Knopoff's method. Abo-Zena [1] proposed a method, which, however, involved a large number of matrix multiplications. It was later modified and made more efficient by Menke [20] and Bhattacharya [5]. Franssens [8] suggested a modified propagator method, and Phinney *et al.* [24] discussed vectorization of reflectivity codes. Finally we may note that the global-matrix method by Schmidt was formulated in terms of potentials and exponentials already from the outset.

In the present paper we discuss delta-matrix factorization as a means of efficiency enhancement for propagation through solid layers, particularly when boundary values at a fluid–solid interface are needed. The basic idea is to write the delta-matrix propagator as a product of *sparse* matrices which can be applied in sequence to the vector being propagated. We propose two kinds of factorizations, the second being an extension of the first. Both our factorizations are initially derived in their 4×4 versions for propagating the displacement-stress vector. They can be immediately applied in any studies of hydroacoustic wave propagation, where propagation through a layered solid region (a solid bottom, an upper solid region such as ice, or a solid region interspaced between two fluid regions) is used to set up appropriate boundary conditions for the fluid (cf. [10, 11]). In seismology, our factorizations are useful for dispersion-function computations and in connection with the reflectivity method for a buried source (cf. [17]).

Our first factorization, presented in Section 3, turns out to give an algorithm very closely related to the "fast form" of Knopoff's method (Schwab [32], Schwab and Knopoff [33]). It is slightly more efficient, however, and, since it does not focus solely on dispersion-function evaluations, its range of applicability is wider. In Section 4 we introduce our second delta-matrix factorization. Like some of the methods already referred to, it has the feature of exponential-factor separation. Due to the additional sparse-matrix property, which is consequently developed, enhanced efficiency is achieved in comparison to previous methods. In particular, for dispersion-function computations it is significantly faster than the "fast form" of Knopoff's method.

In Section 5 we make some comments of a numerical character. For example, we explain how to obtain "optimal" reliability and efficiency by using an adaptive technique for propagation-method control. Relative computer times and test examples to support the statements on numerical stability can be found in Section 6. In Section 7 we discuss the prospects of our factorization methods for multi-frequency computations.

To compare the efficiency of different methods we state precisely the number of different arithmetic operations that are needed to propagate through each individual layer. (Execution times may differ significantly among different computers, cf. [25]). Layer velocities are taken as complex numbers to allow for anelastic attenuation.

2. THE BASIC DELTA-MATRIX PROPAGATOR

In this section we will fix notation, give background, and discuss our version of the well-known basic delta-matrix propagator.

For definiteness, we consider a laterally homogeneous medium composed of N homogeneous solid layers between the depths $z=z_0$ and $z=z_N$; $z_0, z_1, ..., z_N$ are successively increasing interface depths. Below $z=z_N$ we assume a homogeneous solid half-space where only outgoing waves are allowed. Above $z=z_0$ we may have fluid layers or vacuum. (The modifications for a rigid or a free boundary at $z=z_N$ will be obvious. For an upper solid region, simply take $z_0, z_1, ..., z_N$ as decreasing depths.) We will study monofrequent wave propagation with time dependence given by the factor $e^{-i\cdot\omega\cdot t}$, where $\omega>0$ is angular frequency.

Following Aki and Richards [2, Chap. 7], we decompose the wave-field in its horizontal-wavenumber components. Plane waves or cylindrical waves are assumed, both cases giving the same formalism. With k denoting a horizontal wavenumber we have $k=\omega\cdot p$, where p is horizontal slowness. Omitting the dependence on ω and p in our notation, we introduce the displacement-stress vector $\mathbf{r}(z)=(r_1(z),$

$r_2(z)$, $r_3(z)$, $r_4(z))^T$ as in [2]. The components of $\mathbf{r}$ are horizontal displacement, vertical displacement, tangential stress in horizontal cuts, and normal stress in horizontal cuts, respectively. However, we will find it more convenient to use the scaled vector

$$\begin{aligned}\mathbf{y}(z) &= (y_1(z), y_2(z), y_3(z), y_4(z))^T \\ &= (\omega \cdot r_1(z), \omega \cdot r_2(z), r_3(z), r_4(z))^T. \end{aligned} \tag{1}$$

We obtain

$$\mathbf{y}(z) = \mathbf{P}(z, \zeta) \cdot \mathbf{y}(\zeta), \tag{2}$$

where $\mathbf{P}(z, \zeta)$ is the appropriate 4×4 propagator matrix and z, ζ are depth coordinates.

Supposing the medium to be homogeneous between z and ζ (this region is part of one of our homogeneous solid layers), we use the following notation:

$$\begin{gathered}\rho \text{ is density} \\ \alpha \text{ is complex P-velocity } (\mathrm{Im}(\alpha) \leqslant 0) \\ \beta \text{ is complex S-velocity } (\mathrm{Im}(\beta) \leqslant 0) \\ \xi = \left(\frac{1}{\alpha^2} - p^2\right)^{1/2} \text{ is vertical P-slowness}, \qquad \chi = -i \cdot \xi \\ \eta = \left(\frac{1}{\beta^2} - p^2\right)^{1/2} \text{ is vertical S-slowness}, \qquad \nu = -i \cdot \eta \\ d = z - \zeta \text{ is thickness with sign} \\ CP = \cos(\omega \cdot \xi \cdot d), \qquad CQ = \cos(\omega \cdot \eta \cdot d) \\ STP = \xi \cdot \sin(\omega \cdot \xi \cdot d), \qquad STQ = \eta \cdot \sin(\omega \cdot \eta \cdot d) \\ SDP = \sin(\omega \cdot \xi \cdot d)/\xi, \qquad SDQ = \sin(\omega \cdot \eta \cdot d)/\eta \\ \gamma = 2\beta^2 \cdot p^2. \end{gathered} \tag{3}$$

When needed, we will insert a subscript j to indicate the layer j between z_{j-1} and z_j. The subscript $N+1$ is used for the homogeneous half-space.

The expressions for the components of $\mathbf{P} = \mathbf{P}(z, \zeta)$ become

$$\begin{aligned} P_{1,1} &= P_{3,3} = \gamma \cdot CP + (1-\gamma) \cdot CQ \\ P_{1,2} &= -P_{4,3} = p \cdot [2\beta^2 \cdot STQ - (1-\gamma) \cdot SDP] \\ P_{1,3} &= (p^2 \cdot SDP + STQ)/\rho \\ P_{1,4} &= -P_{2,3} = p \cdot (CP - CQ)/\rho \\ P_{2,1} &= P_{3,4} = p \cdot [2\beta^2 \cdot STP - (1-\gamma) \cdot SDQ] \\ P_{2,2} &= P_{4,4} = (1-\gamma) \cdot CP + \gamma \cdot CQ \\ P_{2,4} &= (STP + p^2 \cdot SDQ)/\rho \\ P_{3,1} &= -\rho \cdot [2\beta^2 \cdot \gamma \cdot STP + (1-\gamma)^2 \cdot SDQ] \\ P_{3,2} &= -P_{4,1} = -\rho \cdot 2\beta^2 \cdot p \cdot (1-\gamma) \cdot (CP - CQ) \\ P_{4,2} &= -\rho \cdot [(1-\gamma)^2 \cdot SDP + 2\beta^2 \cdot \gamma \cdot STQ]. \end{aligned} \tag{4}$$

Note that the components of $\mathbf{P}$ are entire analytic functions of p; the choice of branches and sheets for ξ and η causes no ambiguity. Furthermore, ω appears only in the trigonometric factors. In the purely elastic case (no attenuation), $\mathbf{P}$ is a real matrix for real p. With $\mathbf{r}(z)$ (cf. [2, (7.45)]) or Haskell's original variables, we would not have achieved all these attractive features simultaneously.

Now, let $\mathbf{G}$ be the 2×4 matrix which, for the homogeneous half-space, projects $\mathbf{y}(z_N)$ onto its in-going waves (cf. [2, (7.56)]). $\mathbf{G}$ can be taken as a matrix that is analytic in p except at two branch-cuts, one for each of ξ_{N+1} and η_{N+1}. For the case when there is no source below z_0 we obtain

$$\mathbf{G} \cdot \mathbf{P}(z_N, z_0) \cdot \begin{pmatrix} y_1(z_0) \\ y_2(z_0) \\ 0 \\ y_4(z_0) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{5}$$

since $y_3(z_0) = 0$. A solution to (5) is

$$\mathbf{y}(z_0) = (Y_5(z_0), -Y_3(z_0), 0, Y_1(z_0))^T, \tag{6}$$

where $\mathbf{Y}(z_0) = (Y_1(z_0), Y_2(z_0), Y_3(z_0), Y_4(z_0), Y_5(z_0), Y_6(z_0))$ is the 1×6 row-vector given by

$$\mathbf{Y}(z) = [\mathbf{G} \cdot \mathbf{P}(z_N, z)]^{\Delta} = \mathbf{G}^{\Delta} \cdot \mathbf{P}(z_N, z)^{\Delta}. \tag{7}$$

Here we have introduced the superscript Δ to denote the delta-matrix operator. The individual elements of our delta matrices are always 2×2 subdeterminants of the original matrices, the standard row- and column-index ordering $(1, 2, 3, 4, 5, 6) \leftrightarrow [(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)]$ is used. (Cf. Dunkin [7], a crucial property is that the delta matrix of a product is equal to the product of the individual delta matrices.) Hence, $\mathbf{G}^{\Delta}$ is a 1×6 row-vector while $\mathbf{P}^{\Delta}$ is a 6×6 matrix. Disregarding the trivial exceptional case when $Y_1(z_0) = Y_3(z_0) = Y_5(z_0) = 0$, it follows from Cramer's rule that (6) provides the general solution to (5) except for a multiplicative constant.

In conclusion, our problem is to compute

$$\mathbf{Y}(z_0) = \mathbf{G}^{\Delta} \cdot (\mathbf{P}_N)^{\Delta} \cdot (\mathbf{P}_{N-1})^{\Delta} \cdot \cdots \cdot (\mathbf{P}_1)^{\Delta}, \tag{8}$$

where $\mathbf{P}_j = \mathbf{P}(z_j, z_{j-1})$, by propagation of 1×6 row-vectors from left to right. The boundary-value problem (5) has been replaced by the initial-value problem (8). When there are fluid layers above $z = z_0$, the second and fourth components of $\mathbf{y}(z_0)$ according to (6) provide appropriate boundary values for dealing with the fluid region (cf. [36]). When

there is a free surface at $z=z_0$, $Y_1(z_0)=0$ provides an appropriate dispersion equation.

We may note that the formalism leading to (8) turns out to be quite analogous for a solid region interspaced between two fluid regions. The 2×4 matrix $\mathbf{G}$ will be different, however. For cases with a source buried in a solid region, propagation of the type (8) is also useful [17].

It so happens that $(\mathbf{G}^{\Delta})_5 = -(\mathbf{G}^{\Delta})_2$. The antisymmetry

$$Y_5(z) = -Y_2(z) \tag{9}$$

is preserved during propagation according to (8). Recognizing this property, the computational work is reduced [38].

Working from (4), we may write $\mathbf{P}^{\Delta}$ explicitly. We do not give the expressions but note that the frequency-dependence will be isolated to factors $CP\cdot CQ$, $CP\cdot STQ$, $CP\cdot SDQ$, $STP\cdot CQ$, $STP\cdot STQ$, $STP\cdot SDQ$, $SDP\cdot CQ$. $SDP\cdot STQ$, and $SDP\cdot SDQ$. In the leftmost number column of Table I we present an account of the number of arithmetic operations of different types that are needed for propagation of our 1×6 row-vector $\mathbf{Y}$ through one layer, using the explicit form of $\mathbf{P}^{\Delta}$. The symbols "C" and "R" are used for complex and real numbers, respectively. For example, "C/C" designates division of two (different) complex numbers. A few operations that can be performed once for all layers, such as $p\cdot p$, are not included in the table. Furthermore, expressions like $2\beta^2$ are assumed to have been computed in advance and stored.

The efficient code underlying the leftmost number column of Table I is based on ideas by Kind [16]. The code in Fig. 3 of Schwab *et al.* [34], appropriately modified to allow for attenuation and complex velocities, is related but somewhat less efficient.

TABLE I

Number of Arithmetic Operations of Different Types That Are Needed to Propagate the 1×6 Vector $\mathbf{Y}$ through One Layer for a Specific Value of p

Operation type	Basic propagator method (Section 2)	First factorization method (Section 3)	Second factorization method (Section 4)
$e^{R}\cdot$	2	2	2
$[\cos(R), \sin(R)]$	2	2	2
$C^{1/2}$	2	2	2
C/C	2	2	3
$C\cdot C$	63	35	16
$C+C$	41	28	28
$R\cdot C$	16	14	12
$R+C$	3	1	1
$R\cdot R$	3	3	4
$R+R$	2	2	5

Note. The three number columns refer to the three delta-matrix methods discussed in Sections 2, 3, and 4, respectively. The symbols "C" and "R" are used for complex and real numbers, respectively. A few comparison operations for real numbers, needed for program-structure control, have not been included.

Computation of the elementary factors CP and CQ is efficiently done using

$$\cos(C) = e^{\mathrm{Im}(C)}\cdot 0.5\cdot[h\cdot U + \bar{U}], \tag{10}$$

where $h=e^{-2\cdot \mathrm{Im}(C)}$ and $U=\cos(\mathrm{Re}(C))+i\cdot\sin(\mathrm{Re}(C))$, for $\mathrm{Im}(C)\geqslant 0$. An advantage is that $\sin(C)$, needed for STP, SDP and STQ, SDQ, can be obtained with little extra work. The multiplications with "$e^{\mathrm{Im}(C)}$" are actually not performed; instead scaling exponents are stored (see also Section 5). We take care to store the scaling-exponent sums for possible later needs when the analyticity (in p) is crucial, as it is in the root-finding technique by Ivansson and Karasalo [11].

To obtain some idea of the relative computer times needed for the different operations in Table I, we can note that in double precision on a VAX 3200, a "C/C" operation is roughly twice as time-consuming as a "$C\cdot C$" operation. A "$C^{1/2}$" operation is almost four times as demanding; "e^{R}," twice; and "$[\cos(R), \sin(R)]$," five times as demanding as "$C\cdot C$." Concerning "$C+C$," it is almost five times less demanding than "$C\cdot C$." Ordinary operations involving "R" are, of course, even less expensive.

In the next two sections we will introduce delta-matrix factorizations which reduce the required number of "$C\cdot C$" operations drastically. From the rough estimates of relative computer times, this is indeed worthwhile to do.

3. FIRST DELTA-MATRIX FACTORIZATION

Isolating trigonometric factors, the propagator matrix $\mathbf{P}$ in (4) may be written

$$\mathbf{P} = \mathbf{C}^{-1}\cdot\mathbf{P}_T\cdot\mathbf{C}, \tag{11}$$

where

$$\mathbf{P}_T = \begin{pmatrix} CQ & 0 & STQ & 0 \\ 0 & CP & 0 & STP \\ -SDQ & 0 & CQ & 0 \\ 0 & -SDP & 0 & CP \end{pmatrix} \tag{12}$$

$$\mathbf{C} = \begin{pmatrix} 1-\gamma & 0 & 0 & -p \\ 0 & 1-\gamma & -p & 0 \\ 0 & 2\beta^2\cdot p & 1 & 0 \\ 2\beta^2\cdot p & 0 & 0 & 1 \end{pmatrix}\cdot \mathrm{diag}(1, 1, \rho^{-1}, \rho^{-1}) \tag{13}$$

$$\mathbf{C}^{-1} = \mathrm{diag}(1, 1, \rho, \rho) \cdot \begin{pmatrix} 1 & 0 & 0 & p \\ 0 & 1 & p & 0 \\ 0 & -2\beta^2\cdot p & 1-\gamma & 0 \\ -2\beta^2\cdot p & 0 & 0 & 1-\gamma \end{pmatrix}. \tag{14}$$

For the corresponding delta matrices, we obtain

$$\mathbf{P}^{\Delta} = (\mathbf{C}^{-1})^{\Delta} \cdot (\mathbf{P}_T)^{\Delta} \cdot \mathbf{C}^{\Delta}, \tag{15}$$

where

$$(\mathbf{P}_T)^{\Delta} = \begin{pmatrix} CP\cdot CQ & 0 & STP\cdot CQ & -CP\cdot STQ & 0 & STP\cdot STQ \\ 0 & 1 & 0 & 0 & 0 & 0 \\ -SDP\cdot CQ & 0 & CP\cdot CQ & SDP\cdot STQ & 0 & CP\cdot STQ \\ CP\cdot SDQ & 0 & STP\cdot SDQ & CP\cdot CQ & 0 & -STP\cdot CQ \\ 0 & 0 & 0 & 0 & 1 & 0 \\ SDP\cdot SDQ & 0 & -CP\cdot SDQ & SDP\cdot CQ & 0 & CP\cdot CQ \end{pmatrix} \tag{16}$$

$$\mathbf{C}^{\Delta} = \begin{pmatrix} (1-\gamma)^2 & -(1-\gamma)\cdot p & 0 & 0 & (1-\gamma)\cdot p & -p^2 \\ 2\beta^2\cdot p\cdot(1-\gamma) & 1-\gamma & 0 & 0 & \gamma & p \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ -2\beta^2\cdot p\cdot(1-\gamma) & \gamma & 0 & 0 & 1-\gamma & -p \\ -(2\beta^2\cdot p)^2 & -2\beta^2\cdot p & 0 & 0 & 2\beta^2\cdot p & 1 \end{pmatrix} \times \frac{1}{\rho}\cdot \mathrm{diag}(\rho, 1, 1, 1, 1, \rho^{-1}) \tag{17}$$

$$(\mathbf{C}^{-1})^{\Delta} = \rho\cdot\mathrm{diag}(\rho^{-1}, 1, 1, 1, 1, \rho)\times \begin{pmatrix} 1 & p & 0 & 0 & -p & -p^2 \\ -2\beta^2\cdot p & 1-\gamma & 0 & 0 & \gamma & -(1-\gamma)\cdot p \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 2\beta^2\cdot p & \gamma & 0 & 0 & 1-\gamma & (1-\gamma)\cdot p \\ -(2\beta^2\cdot p)^2 & 2\beta^2\cdot p\cdot(1-\gamma) & 0 & 0 & -2\beta^2\cdot p\cdot(1-\gamma) & (1-\gamma)^2 \end{pmatrix}. \tag{18}$$

The important point is that the factor matrices in (15) are relatively *sparse* matrices of simple form. Propagating a 1×6 row-vector $\mathbf{Y}$ with the antisymmetry property (9) by multiplication with $\mathbf{P}^{\Delta}$ as in (8), significantly enhanced efficiency is achieved if the factor matrices of (15), and (16)–(18), are applied in sequence. In fact, by carefully exploiting common factors etc., multiplication with $(\mathbf{C}^{-1})^{\Delta}$ can be handled with only five "$C\cdot C$", eight "$C+C$", and two "$R\cdot C$" operations. The same is true for multiplication with $\mathbf{C}^{\Delta}$. Here we have not counted a required computation of $2\beta^2\cdot p$, which is common to both these multiplications, and scaling with ρ or ρ^{-1} has been performed.

The very propagation with $(\mathbf{P}_T)^{\Delta}$ according to (16) requires 16 "$C\cdot C$", 12 "$C+C$", one "$R\cdot C$", and two "$R+R$" operations, if scaling is made and stored appropriately. In addition, the elementary trigonometric factors and their products must first be computed. This is done in the same way as indicated in Section 2.

We have emphasized the usefulness of splitting the matrix $\mathbf{P}$ into factors which are applied in sequence to the 1×6 row-vector $\mathbf{Y}$ being propagated. However, it turns out to be beneficial to lump together adjacent $(\mathbf{C}_{j+1})^{\Delta}$ and $(\mathbf{C}_j^{-1})^{\Delta}$ that appear in (8) when individual factors are decomposed according to (15). Starting with the original matrices, we obtain

$$\mathbf{C}_{j+1}\cdot\mathbf{C}_j^{-1} = \begin{pmatrix} 1+\delta_j\cdot p^2 & 0 & 0 & [(1-q_j)+\delta_j\cdot p^2]\cdot p \\ 0 & 1+\delta_j\cdot p^2 & [(1-q_j)+\delta_j\cdot p^2]\cdot p & 0 \\ 0 & -\delta_j\cdot p & q_j-\delta_j\cdot p^2 & 0 \\ -\delta_j\cdot p & 0 & 0 & q_j-\delta_j\cdot p^2 \end{pmatrix} \tag{19}$$

where $q_j = \rho_j/\rho_{j+1}$ and $\delta_j = q_j\cdot 2\beta_j^2 - 2\beta_{j+1}^2$. Note that the product matrix in (19) is as sparse as each of its two factors (there is no "fill-in").

For the delta matrix we obtain the attractive representation after some tedious algebra,

$$(\mathbf{C}_{j+1}\cdot\mathbf{C}_j^{-1})^{\Delta} = \mathrm{diag}(1, q_j, q_j, q_j, q_j, 1) + \mathbf{H}_j\cdot\mathrm{diag}(\delta_j\cdot p, p, 0, 0, p, \Delta_j), \tag{20}$$

where $\Delta_j = (1-q_j)+\delta_j\cdot p^2$ and

$$\mathbf{H}_j = \begin{pmatrix} 2\cdot p & \Delta_j & 0 & 0 & -\Delta_j & 0 \\ -1 & 0 & 0 & 0 & 0 & -p \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & p \\ 0 & \delta_j & 0 & 0 & -\delta_j & -2 \end{pmatrix} + \begin{pmatrix} -p^2 \\ p \\ 0 \\ 0 \\ -p \\ 1 \end{pmatrix}\cdot(-\delta_j\cdot p, -\delta_j\cdot\Delta_j, 0, 0, \delta_j\cdot\Delta_j, \Delta_j). \tag{21}$$

The matrix (20) handles reflection and transmission at the interface at depth z_j in one step. Since $\mathbf{G}$ in (5) can in fact be factorized with $\mathbf{C}_{N+1}$ as its right factor, (20) is preferably applied already at the interface of the homogeneous half-space ($j=N$).

Taking advantage of the simple structure, computation of Δ_j and propagation of our 1×6 row-vector $\mathbf{Y}$ with $(\mathbf{C}_{j+1}\cdot\mathbf{C}_j^{-1})^{\Delta}$ turns out to be possible with only eight "$C\cdot C$", ten "$C+C$", five "$R\cdot C$", and one "$R+C$" operations. The total requirements, including computation of and propagation with $(\mathbf{P}_T)^{\Delta}$, are specified in the middle number column of Table I. Comparing to the leftmost number column, our sparse-matrix factorization according to (15) has been very successful. Only 35 "$C\cdot C$" operations are needed for each layer.

We may compare our present method, involving alternating propagations with $(\mathbf{C}_{j+1}\cdot\mathbf{C}_j^{-1})^{\Delta}$ and $(\mathbf{P}_T)_j^{\Delta}$, to the "fast form" of Knopoff's method [32; 33; 34, Fig. 1]. The algorithms that finally emerge have striking similarities, although they are derived in different ways. Our algorithm

is applicable not only for computation of dispersion functions, since the derivation focuses on the basic delta-matrix propagator rather than on the global-matrix determinant. Furthermore, our method is the more efficient of the two, since fewer arithmetic operations are needed.

4. SECOND DELTA-MATRIX FACTORIZATION

Simply by using Euler's formulas we may decompose $\mathbf{P}_T$ from (11) and write

$$\mathbf{P} = \mathbf{C}^{-1} \cdot \mathbf{P}_T \cdot \mathbf{C} = \mathbf{C}^{-1} \cdot \mathbf{D}^{-1} \cdot \mathbf{P}_E \cdot \mathbf{D} \cdot \mathbf{C}, \tag{22}$$

where

$$\mathbf{P}_E = \operatorname{diag}(e^{-\omega \cdot \nu \cdot d}, e^{-\omega \cdot \chi \cdot d}, e^{\omega \cdot \nu \cdot d}, e^{\omega \cdot \chi \cdot d}) \tag{23}$$

$$\mathbf{D} = \frac{1}{2} \cdot \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 1 \end{pmatrix} \cdot \operatorname{diag}\left(\frac{1}{\nu}, \frac{1}{\chi}, 1, 1\right) \tag{24}$$

$$\mathbf{D}^{-1} = \operatorname{diag}(\nu, \chi, 1, 1) \cdot \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}. \tag{25}$$

Here d, χ, and ν are defined in (3). We may note that the operation (change of variables) given by $\mathbf{D}_j \cdot \mathbf{C}_j \cdot \mathbf{y}(z)$ furnishes a projection onto down-going and up-going P- and S-waves within layer j.

For the corresponding delta matrices, we obtain

$$\begin{aligned} \mathbf{P}^{\Delta} &= (\mathbf{C}^{-1})^{\Delta} \cdot (\mathbf{P}_T)^{\Delta} \cdot \mathbf{C}^{\Delta} \\ &= (\mathbf{C}^{-1})^{\Delta} \cdot [(\mathbf{D}^{-1})^{\Delta}]_{\text{mod}} \cdot (\mathbf{P}_E)^{\Delta} \cdot (\mathbf{D}^{\Delta})_{\text{mod}} \cdot \mathbf{C}^{\Delta}, \end{aligned} \tag{26}$$

where

$$\begin{aligned} (\mathbf{P}_E)^{\Delta} = \operatorname{diag}(&e^{-\omega \cdot (\chi+\nu) \cdot d}, 1, e^{\omega \cdot (\chi-\nu) \cdot d}, \\ &e^{-\omega \cdot (\chi-\nu) \cdot d}, 1, e^{\omega \cdot (\chi+\nu) \cdot d}) \end{aligned} \tag{27}$$

$$\begin{aligned} (\mathbf{D}^{\Delta})_{\text{mod}} = &\begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \end{pmatrix} \\ &\cdot \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 \end{pmatrix} \\ &\times \frac{1}{4} \cdot \operatorname{diag}\left(\frac{1}{\chi \cdot \nu}, 4, \frac{1}{\nu}, \frac{1}{\chi}, 4, 1\right) \end{aligned} \tag{28}$$

$$\begin{aligned} [(\mathbf{D}^{-1})^{\Delta}]_{\text{mod}} = &\operatorname{diag}(\chi \cdot \nu, 1, \nu, \chi, 1, 1) \\ &\times \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \end{pmatrix} \\ &\cdot \begin{pmatrix} 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & -1 & 0 & 0 \end{pmatrix}. \end{aligned} \tag{29}$$

The matrices $\mathbf{D}^{\Delta}$ and $(\mathbf{D}^{-1})^{\Delta}$ have been slightly modified to enhance computational efficiency, hence the subscripts mod.

The matrices in (27)–(29) are indeed very sparse. Propagation of our 1×6 vector $\mathbf{Y}$, with the antisymmetry property (9), can be done with a very small number of arithmetic operations if the factor matrices of (26)–(29) are applied in sequence, from left to right. Counting all operations, including those to compute the elements of the factor matrices, we obtain the rightmost column of Table I. Here we have lumped $(\mathbf{C}_{j+1})^{\Delta}$ and $(\mathbf{C}_j^{-1})^{\Delta}$ together as in Section 3, and the elements of $(\mathbf{P}_E)^{\Delta}$ have been computed efficiently with storage of an appropriate scaling exponent (see Section 5). Note in particular that we only need 16 "$C \cdot C$" operations in total!

The idea to isolate the exponentials $e^{\omega(\pm\chi\pm\nu)\cdot d}$ is not new. Its potential for reducing the required number of arithmetic operations does not seem to have been fully recognized, however. Instead, loss-of-precision control has often been a major concern (e.g., [1]). Our matrix $(\mathbf{P}_E)^{\Delta}$ appears in [8] as well as in [24], but sparse-matrix factorization for the remaining parts of $\mathbf{P}^{\Delta}$ is not treated and the latter paper concentrates on multi-frequency computations (cf. Section 7). Menke [20] and Bhattacharya [5], as well as Franssens [8], discuss propagation in terms of a matrix

$$\begin{aligned} &\mathbf{M}_1 + e^{-\omega \cdot (\chi+\nu) \cdot d} \cdot \mathbf{M}_2 + e^{-\omega \cdot (\chi-\nu) \cdot d} \cdot \mathbf{M}_3 \\ &\quad + e^{\omega \cdot (\chi-\nu) \cdot d} \cdot \mathbf{M}_4 + e^{\omega \cdot (\chi+\nu) \cdot d} \cdot \mathbf{M}_5, \end{aligned} \tag{30}$$

where the matrices $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, $\mathbf{M}_4$, $\mathbf{M}_5$ are not sparse. The formulation by Bhattacharya [5], probably the most efficient among these and actually derived using some matrices of the type we have considered, was claimed by the author to have the same efficiency as a traditional (reduced) delta-matrix propagator method for dispersion-function computations and hence be 38% slower than Knopoff's method.

5. NUMERICAL CONSIDERATIONS

In this section we discuss some numerical aspects on implementation of the methods proposed in the preceding sections. First we may note that the factorization in Section 4 breaks down for layer j if p is such that $\chi = 0$ or $\nu = 0$, i.e., if $p = \pm 1/\alpha_j$ or $p = \pm 1/\beta_j$. In the immediate vicinity of these points we may also expect this factorization to be unsuitable, considering error propagation during the computations. The appropriate course of action is obvious: the propagation method is chosen separately for each layer in an *adaptive* way, taking the current value of p into account. Since the propagation technique of Section 4 is the fastest, this method is chosen except in the exceptional cases when p is very close to the P- or S-slowness of the layer in question. For the exceptional cases, the propagation technique of Section 3 is used.

In fact, one could easily develop hybrid methods. For p close to $\pm 1/\alpha$, for example, one could use a propagator factorization with the technique of Section 4 for the "S part" but with the "P part" relying on *CP*, *STP*, *SDP* as in Section 3.

For computation of *CP*, *STP*, *SDP*, etc., we have pointed out the suitability of using Euler's formulas as in (10). However, for certain arguments numerical loss-of-precision may occur when the sum (or difference) is computed. This appears to be harmless except for *SDP* and *SDQ* in certain cases when ξ or η is small. In these cases, the error may be magnified by division with a small quantity. The remedy is again obvious: when ξ (η) is very small, we compute *CP*, *STP*, *SDP* (*CQ*, *STQ*, *SDQ*) in part by using real-arithmetic series development for certain hyperbolic-function factors rather than by directly using expressions like (10).

For p giving rise to evanescent waves, scaling must be performed to avoid overflow in $(\mathbf{P}_E)^d$ according to (27). It is quite obvious how to scale properly. Suppose for definiteness that $\mathrm{Re}(\chi)$ and $\mathrm{Re}(\nu)$ are nonnegative. Then we may simply add $\omega \cdot \mathrm{Re}(\chi + \nu) \cdot d$ to a stored scaling-exponent sum and apply $(\mathbf{P}_E)^d$ in the form (scaled to maximum-norm 1)

$$\begin{aligned}\mathrm{diag}((h_P \cdot h_Q)^2 \cdot e^{-i \cdot \omega \cdot \mathrm{Im}(\chi+\nu)\cdot d}, h_P \cdot h_Q,\\ (h_Q \cdot h_Q) \cdot e^{i \cdot \omega \cdot \mathrm{Im}(\chi-\nu)\cdot d},\\ (h_P \cdot h_P) \cdot e^{-i \cdot \omega \cdot \mathrm{Im}(\chi-\nu)\cdot d}, h_P \cdot h_Q, e^{i \cdot \omega \cdot \mathrm{Im}(\chi+\nu)\cdot d}).\end{aligned} \quad (31)$$

where $h_P = e^{-\omega \cdot \mathrm{Re}(\chi)\cdot d}$ and $h_Q = e^{-\omega \cdot \mathrm{Re}(\nu)\cdot d}$. This is in fact the same type of scaling that was discussed in connection with (10), and it was included for the compilation of Table I. As pointed out by Schwab *et al.* [34], the risk of overflow is eliminated in this way. Thick layers can be handled without layer splitting even in the high-frequency regime. As an additional precaution against overflow, it is advisable to scale the propagated row-vector itself if it becomes very large or very small.

There are certain 1×6 row-vectors for which propagation with the matrix in (31) is not always numerically reliable. These are the row-vectors whose last component vanishes. If the angular frequency ω or the layer thickness d is large enough, the propagated vector may become the zero vector because of underflow, or small numerical noise in the last component may dominate the result. Using the observation after (25), and the definition of a delta matrix, it is easily realized that a row-vector of the type discussed will enter "$(\mathbf{P}_E)^d$ of layer j" precisely when p gives a mode for the modified medium where layer j is continued to a homogeneous half-space where only out-going waves are allowed. Such p form a discrete set.

In fact, the problem just discussed appears to be harmful only when the shape of certain modes (typically interface modes) as functions of depth is desired. In such cases, special techniques like "parallel shooting" in combination with "inverse iteration" are needed at a final stage (cf. [13, 34, 26]). Applications to wavenumber integration and solution of the dispersion equation (cf. [10, 11]) are quite successful. We may note that a "problematic vector," connected to a certain p, correctly becomes very small in magnitude upon propagation as compared to what is obtained at adjacent p.

For the purely elastic case (no attenuation) with real or imaginary p, only real arithmetic is needed for the methods of Sections 2 and 3, as is also the case for dispersion-function computations with Knopoff's method. (To obtain a real-valued start vector $\mathbf{G}^d$ for all such p, a version of (8) taken in the opposite direction should be used to avoid a "homogeneous half-space" $\mathbf{G}$.) In this case, the complex operations of Table I can be replaced with the corresponding operations with reals. Furthermore, only one of "e^R and $[\cos(R), \sin(R)]$" will be needed. For our second factorization method as developed in Section 4, real arithmetic for a certain layer will require in addition that p is greater than both wave slownesses for this layer. Hence, to achieve real-arithmetic computations with maximum efficiency, an adaptive approach should be taken, where the method of Section 4 is used for all layers except for the "low-velocity" layers where the method of Section 3 is invoked.

6. PERFORMANCE ON THE COMPUTER

Relative computer times for propagation through one homogeneous solid layer (anelastic case) with our different methods have been obtained by computations on a VAX 3200 computer. For our second factorization method (Section 4) we needed 1.239 ms, which will be referred to as one time unit. The times needed for our first factorization method (Section 3) and our basic delta-matrix propagator

method (Section 2) were 1.27 and 1.72 time units, respectively. These results show clearly the gains that are obtained. Moreover, they are consistent with the contents of Table I, confirming the usefulness of counting and minimizing the number of required arithmetic operations.

For completeness, the computer codes given by Schwab *et al.*, [34, Figs. 1 and 3], were also tested. To allow for anelasticity, these codes were changed in the obvious way [33, p. 148] by introducing complex arithmetic. One may note that there is no longer a need for a separate nonevanescent option. As for the methods of Table I, efficiency was enhanced by using "e^R and $[\cos(R), \sin(R)]$" rather than "e^C" and by noting that scaling is not necessary at each layer. The results that emerged were 1.35 time units for the "fast form" Knopoff-method code [34, Fig. 1] and 1.94 time units for the reduced delta-matrix code [34, Fig. 3]. Again, the time consumption directly reflects the number of arithmetic operations. In particular, there are 45 "$C \cdot C$" operations in the Knopoff-method code, while 77 such operations appear in this delta-matrix code.

The relative computer times have been collected in Table II. Our previous statements that our basic delta-matrix code is more efficient than the corresponding code in [34] and that our first factorization method is more efficient than Knopoff's method have now been justified by computer tests. Finally, one should keep in mind that Knopoff's method is a dispersion-function method only, whereas all the other methods have a wider range of applicability.

To give a more concrete appreciation of the performance of our methods, we also include an application to a particular reasonably realistic problem. Consider a medium that consists of an isovelocity water column with depth 40 m overlying a 25 m thick solid sediment structure modeled by 100 solid layers (to mimic a gradient) and a bedrock modeled as a homogeneous solid half-space with P-velocity 4.6 km/s, S-velocity 2.5 km/s, P-attenuation 0.08 dB/λ, S-attenuation 0.2 dB/λ, and density 2.6 kg/dm^3. In the sediment, the P- and S-velocities increase with depth from 1.6 to 3.6 and from 0.4 to 1.0 km/s, respectively. The P- and S-attenuations increase from 0.165 to 0.265 and from 0.5 to 0.8 dB/λ, respectively, whereas the density increases from 1.5 to 2.5 kg/dm^3. We wish to determine all modal wavenumbers in the first quadrant with real and imaginary slowness parts less than 5.0 and 1.0 s/km, respectively. Vertical Pekeris branch-cuts are used, cf. [4].

The dispersion equation for this example was solved with the adaptive winding-number integration method of Ivansson and Karasalo [11]. This provides a useful test for our methods, since computations for a vast number of different complex p are needed. In Table III we provide some performance details. By comparing the numbers within parentheses in Table III to the computer times per layer propagation in Table II, we realize that about 90% or more of the computation time in this example comes from the solid-layer propagations. The remainder, which is common to the three methods, arises from handling of stacks, adaptive control, etc. [11]. As expected, for the same method the time needed for propagation through one layer is fairly independent of frequency. Our second factorization method clearly exhibits the best efficiency. Furthermore, the agreement between the modal wavenumbers as determined by the three different methods was excellent.

Further computational examples, for which our factorization methods have been tested, can be found in [10] (applications to wavenumber integration) and [11]. In Fig. 3 of [11], vast numbers of modal wavenumbers were found close to the P- and S-slowness values of a thick layer. No problems were encountered for any of the p values treated, and the adaptive approach described at the begin-

TABLE II

CPU Times, with Complex Double Precision on a VAX 3200, to Handle One Solid Layer with Different Methods

Basic propagator method (Section 2)	2.131 ms	1.72
First factorization method (Section 3)	1.576 ms	1.27
Second factorization method (Section 4)	1.239 ms	1.00
Propagator code of [34]	2.406 ms	1.94
Knopoff-method code of [34]	1.679 ms	1.35

Note. The numbers in the rightmost column are relative times, taking the result for the second factorization method as one time unit.

TABLE III

Performance Measures for Determination of Modal Wavenumbers for a Certain Structure Involving 100 Solid Bottom Layers and a Homogeneous Half-Space below One Fluid Layer

			CPU time		
Freq (Hz)	Number of modes located	Number of treated p values	Basic propagator method (Section 2)	First factorization method (Section 3)	Second factorization method (Section 4)
25	8	1589	6 min 8 s (2.31 ms)	4 min 35 s (1.73 ms)	3 min 43 s (1.40 ms)
100	27	7130	27 min 5 s (2.28 ms)	20 min 12 s (1.69 ms)	16 min 24 s (1.38 ms)
500	131	48939	3 h 5 min (2.27 ms)	2 h 18 min (1.69 ms)	1 h 52 min (1.38 ms)
1000	261	115350	7 h 16 min (2.27 ms)	5 h 26 min (1.70 ms)	4 h 25 min (1.38 ms)

Note. The root-finding technique of Ivansson and Karasalo [11] has been used. The computations were made in complex double precision on a VAX 3200 computer. The numbers within parentheses refer to the CPU time divided by the product of the number of solid layers (100) and the number of treated p values.

ning of Section 5 proved quite successful. In addition, the computations in [11] were made not only with delta-matrix propagator methods but also with an independent computer code by Karasalo based on "exact finite elements" [12]. The agreement was excellent.

In conclusion, the efficiency of our factorization methods has been clearly demonstrated in terms of computer times. No stability problems have been encountered and our different methods have given numerically equivalent results in all cases.

7. MULTI-FREQUENCY COMPUTATIONS

As noted by Kind and Odom [18], we have

$$\mathbf{P}(\omega_1+\omega_2)=\mathbf{P}(\omega_1)\cdot\mathbf{P}(\omega_2) \tag{32}$$

and, hence,

$$[\mathbf{P}(\omega_1+\omega_2)]^{\Delta}=[\mathbf{P}(\omega_1)]^{\Delta}\cdot[\mathbf{P}(\omega_2)]^{\Delta}. \tag{33}$$

(In other words, analogous addition rules hold for frequencies and for depths.) Here $\mathbf{P}$ is our layer propagator from (4); we have explicitly indicated the angular frequencies which will be varied. In the same way, we obtain for our propagator matrices from Sections 3 and 4 the relations

$$\mathbf{P}_T(\omega_1+\omega_2)=\mathbf{P}_T(\omega_1)\cdot\mathbf{P}_T(\omega_2) \tag{34}$$

$$[\mathbf{P}_T(\omega_1+\omega_2)]^{\Delta}=[\mathbf{P}_T(\omega_1)]^{\Delta}\cdot[\mathbf{P}_T(\omega_2)]^{\Delta} \tag{35}$$

and

$$\mathbf{P}_E(\omega_1+\omega_2)=\mathbf{P}_E(\omega_1)\cdot\mathbf{P}_E(\omega_2) \tag{36}$$

$$[\mathbf{P}_E(\omega_1+\omega_2)]^{\Delta}=[\mathbf{P}_E(\omega_1)]^{\Delta}\cdot[\mathbf{P}_E(\omega_2)]^{\Delta}. \tag{37}$$

When computations for a number of equidistant frequencies $\Delta\omega$, $2\cdot\Delta\omega$, $3\cdot\Delta\omega$, ... are required, these relations can be used for recursive computation of $[\mathbf{P}(n\cdot\Delta\omega)]^{\Delta}$, $[\mathbf{P}_T(n\cdot\Delta\omega)]^{\Delta}$, and $[\mathbf{P}_E(n\cdot\Delta\omega)]^{\Delta}$, respectively, for $n=1, 2, 3, \ldots$. We only need to compute in advance and store a factor matrix $[\mathbf{P}(\Delta\omega)]^{\Delta}$, $[\mathbf{P}_T(\Delta\omega)]^{\Delta}$, or $[\mathbf{P}_E(\Delta\omega)]^{\Delta}$, respectively [18, 24]. To speed up the matrix multiplications, one should note that $\mathbf{P}^{\Delta}$, $(\mathbf{P}_T)^{\Delta}$, and $(\mathbf{P}_E)^{\Delta}$ have only 15, 9, and 4 independent elements, respectively. As for $(\mathbf{P}_E)^{\Delta}$, it is readily realized that the recursively generated scaled versions according to (31) will all have maximum-norm one.

With this approach to computation of the delta-matrix propagator for a new frequency, we can compile a table like Table I to specify the number of arithmetic operations needed. This is done in Table IV. Repeated evaluation of mathematical functions is not needed, but concerning $\mathbf{P}^{\Delta}$ a large number of operations are needed to update its 15 independent components according to (33). As a consequence, the present multi-frequency technique does not enhance efficiency in connection with the basic delta-matrix propagator method of Section 2.

TABLE IV

Number of Arithmetic Operations of Different Types That Are Needed to Propagate the 1×6 Vector $\mathbf{Y}$ through One Layer for a Specific Value of p

Operation type	Basic propagator method	First factorization method	Second factorization method
e^R	—	—	—
$[\cos(R), \sin(R)]$	—	—	—
$C^{1/2}$	—	—	—
C/C	—	—	—
$C\cdot C$	100	59	21
$C+C$	86	49	26
$R\cdot C$	—	6	7
$R+C$	—	—	—
$R\cdot R$	—	1	1
$R+R$	2	2	3

Note. In contrast to Table I, we consider an "additional" frequency for which the multi-frequency techniques of Section 7 are applied to eliminate the need for repeated evaluation of mathematical functions.

Concerning our second factorization method (Section 4), however, significant gains in efficiency can be expected by comparing Tables IV and I. Since $(\mathbf{P}_E)^{\Delta}$ is a simple diagonal matrix, only four "$C\cdot C$", one "$R\cdot R$", and one "$R+R$" operations are needed to update it to a new frequency (see (31) for the scaling). The gains, more than a doubling of the speed, have been confirmed by computer tests. Due to our consequent sparse-matrix factorization we obtain a more efficient multi-frequency algorithm than Phinney *et al.* [24], who seem to need 33 "$C\cdot C$" operations in comparison to our 21. For the same reasons as in Section 5, an adaptive approach involving our first factorization method is needed to be able to handle *all* values of p.

Multi-frequency computations of the type discussed occur naturally in computation of transient pulse responses by wavenumber-integration methods. A useful strategy might be to consider different groups of frequencies, with the frequency increasing between groups and each group being treated by the multi-frequency approach. In this way, one could take advantage of adaptive techniques, such as those proposed by Ivansson and Karasalo [10, 11], and concentrate the computational work to the high-frequency groups, where "many p" must be considered to achieve accurate results. Furthermore, numerical accuracy would be degraded by too many recursive multiplications. Computation of dispersion curves is often done by "mode following" techniques (e.g., [23]). Taking into account the possibility to make multi-frequency computations rapidly for the same p might prove useful in this context as well (cf. [15, p. 307]).

8. CONCLUSIONS

Propagation of a row-vector by multiplication with (reduced) delta matrices for solid layers is a useful tool in connection with wave-propagation studies for layered fluid–solid media. By factorizing the delta matrices as products of *sparse* matrices which are applied in sequence to the row-vector being propagated, computational efficiency can be enhanced significantly. With appropriate scaling, even high-frequency cases can be treated without subdividing thick layers.

We have proposed two kinds of delta-matrix factorizations. The first one turned out to give an algorithm which, for dispersion-function computations, can be considered as a slightly improved version of the "fast form" of Knopoff's method. Our second factorization provided a significantly faster algorithm which involves very few arithmetic operations. Since it is not valid for certain exceptional wavenumbers, however, an adaptive technique involving both factorizations is needed to achieve complete reliability.

The required number of arithmetic operations of different kinds for propagation through one solid layer is detailed in Table I in the text. In particular, concerning the complex multiplications we only need 16 with our second factorization. With complex double-precision calculations on a VAX 3200, we achieve about 470, 635, and 810 solid-layer propagations per second for the three methods considered in Table I, respectively. For comparison, we achieved handling 415 solid layers per second with the anelastic version of the delta-matrix code in [34] and 595 solid layers with Knopoff's method ("fast form").

Knopoff's method is a dispersion-function method only. Due to the direct connection to the traditional delta-matrix propagator, our factorization methods are not only applicable to dispersion-function evaluations but also in connection with wavenumber-integration methods for studies of propagation from a point source. It is here that the gains in comparison to previous methods become most conspicuous. For the basic situation in underwater acoustics, with a fluid region containing a point source and hydrophones on top of a solid layered bottom, the appropriate boundary values at the fluid–solid interface can be computed very rapidly. The formalism turns out to be quite analogous for an upper solid region or a solid region interspaced between two fluid regions.

Delta-matrix factorization is also highly beneficial in connection with multi-frequency computations, as indicated by Table IV in the text. With our second factorization method we achieve more than 1600 solid-layer propagations per second on our VAX 3200 for each additional frequency.

Full wave methods based on a "global matrix" have been claimed to be useful for cases with point sources and receivers at many different depths within the solid regions. With a consequent propagator approach for such cases, the ordinary 4×4 propagator matrices will be needed for some solid layers in addition to the delta matrices [17]. Our factorizations give enhancement of efficiency for propagation by ordinary propagator matrices too (11–14, 22–25), in conjunction with delta-matrix propagation as well as separately.

ACKNOWLEDGMENTS

Some years ago I got a good introduction to wave-propagation problems from using a reflectivity program developed by Rainer Kind. During the last two years I have had the opportunity to work in a research group headed by Ilkka Karasalo, which has been very stimulating.

Note added in proof. In a recent paper (Geophys. J. Int., October 1992), Bhattacharya derived a decomposed representation of the same type as our second factorization. He used it for proving layer-reduction results, but approaches to using it for the actual computations were not discussed.

REFERENCES

1. A. Abo-Zena, *Geophys. J. R. Astron. Soc.* **58**, 91 (1979).
2. K. Aki and P. G. Richards, *Quantitative Seismology* (Freeman, San Francisco, 1980).
3. A. S. Alekseev and B. G. Mikhailenko, *J. Geophys.* **48**, 161 (1980).
4. C. L. Bartberger, *J. Acoust. Soc. Am.* **61**, 1643 (1977).
5. S. N. Bhattacharya, *Geophys. J. R. Astron. Soc.* **88**, 97 (1987).
6. R. C. Y. Chin, G. W. Hedstrom, and L. Thigpen, *Geophys. J. R. Astron. Soc.* **77**, 483 (1984).
7. J. Dunkin, *Bull. Seismol. Soc. Am.* **55**, 335 (1965).
8. G. R. Franssens, *Geophys. J. R. Astron. Soc.* **75**, 669 (1983).
9. N. A. Haskell, *Bull. Seismol. Soc. Am.* **43**, 17 (1953).
10. S. Ivansson and I. Karasalo, *J. Acoust. Soc. Am.* **92**, 1569 (1992).
11. S. Ivansson and I. Karasalo, *J. Sound Vib.* **161**, 173 (1993).
12. I. Karasalo, *J. Sound Vib.*, to appear.
13. H. B. Keller, *Numerical Solution of Two Point Boundary Value Problems*, SIAM Regional Conference Series in Applied Mathematics, (SIAM, Philadelphia, 1976).
14. B. L. N. Kennett, *Bull. Seismol. Soc. Am.* **64**, 1685 (1974).
15. B. L. N. Kennett, *Seismic Wave Propagation in Stratified Media* (Cambridge Univ. Press, Cambridge, 1983).
16. R. Kind, *J. Geophys.* **42**, 191 (1976).
17. R. Kind, *J. Geophys.* **44**, 603 (1978).
18. R. Kind and R. I. Odom, *J. Geophys.* **53**, 127 (1983).
19. L. Knopoff, *Bull. Seismol. Soc. Am.* **54**, 431 (1964).
20. W. Menke, *Geophys. J. R. Astron. Soc.* **59**, 315 (1979).
21. A. H. Olson, J. A. Orcutt, and G. A. Frazier, *Geophys. J. R. Astron. Soc.* **77**, 421 (1984).
22. G. F. Panza, *J. Geophys.* **58**, 125 (1985).
23. G. F. Panza and P. Suhadolc, in *Seismic Strong Motion Synthetics*, edited by B. A. Bolt (Academic Press, 1987), p. 153.
24. R. A. Phinney, R. I. Odom, and G. J. Fryer, *Bull. Seismol. Soc. Am.* **77**, 2218 (1987).
25. L. D. Porter, F. Schwab, K. K. Nakanishi, I. F. Weeks, G. F. Panza, E. Mantovani, D. McMenamin, W. D. Smythe, A. H. Liao, J. A. Landoni, N. N. Biswas, F.-S. Chang, S. S. Bor, E. G. Kausel, P. Gasperini, and J. L. Luthey, *Bull. Seismol. Soc. Am.* **70**, 1415 (1980).

26. M. B. Porter and E. L. Reiss, *J. Acoust. Soc. Am.* **77**, 1760 (1985).
27. M. J. Randall, *Bull. Seismol. Soc. Am.* **57**, 1299 (1967).
28. H. Schmidt, SACLANTCEN Report SR-113, La Spezia, 1988 (unpublished).
29. H. Schmidt and J. Glattetre, *J. Acoust. Soc. Am.* **78**, 2105 (1985).
30. H. Schmidt and F. Jensen, *J. Acoust. Soc. Am.* **77**, 813 (1985).
31. H. Schmidt and G. Tango, *Geophys. J. R. Astron. Soc.* **84**, 331 (1986).
32. F. Schwab, *Bull. Seismol. Soc. Am.* **60**, 1491 (1970).
33. F. Schwab and L. Knopoff, in *Seismology: Surface Waves and Earth Oscillations*, Methods in Computational Physics, Vol. 11, edited by B. A. Bolt (Academic Press, New York, 1972), p. 87.
34. F. Schwab, K. Nakanishi, M. Cuscito, G. F. Panza, G. Liang, and J. Frez, *Bull. Seismol. Soc. Am.* **74**, 1555 (1984).
35. P. Spudich and U. Ascher, *Geophys. J. R. Astron. Soc.* **75**, 101 (1983).
36. H. Takeuchi and M. Saito, in *Seismology: Surface Waves and Earth Oscillations*, Methods in Computational Physics, Vol. 11, edited by B. A. Bolt (Academic Press, New York, 1972), p. 217.
37. W. T. Thomson, *J. Appl. Phys.* **21**, 89 (1950).
38. T. H. Watson, *Bull. Seismol. Soc. Am.* **60**, 161 (1970).