Geophys. J. Int. (1996) **124,** 869–887

Free-mode surface-wave computations

P. W. Buchen and R. Ben-Hador
School of Mathematics and Statistics, University of Sydney, NSW 2006, Australia

Accepted 1995 October 6. Received 1995 July 24; in original form 1995 April 12

SUMMARY

The theory of Love- and Rayleigh-wave dispersion for plane-layered earth models has undergone a number of developments since the initial work of Thomson and Haskell. Most of these were concerned with computational difficulties associated with numerical overflow and loss of precision at high frequencies in the original Thomson–Haskell formalism. Several seemingly distinct approaches have been followed, including the delta matrix, reduced delta matrix, Schwab–Knopoff, fast Schwab–Knopoff, Kennett's Reflection–Transmission Matrix and Abo-Zena methods. This paper analyses all these methods in detail and finds explicit transformations connecting them. It is shown that they are essentially equivalent and, contrary to some claims made, each solves the loss of precision problem equally well. This is demonstrated both theoretically and computationally. By extracting the best computational features of the various methods, we develop a new algorithm (see Appendix A5), called the *fast delta matrix algorithm.* To date, this is the simplest and most efficient algorithm for surface-wave dispersion computations (see Fig. 4).

The theory given in this paper provides a complete review of the principal methods developed for Love- and Rayleigh-wave dispersion of free modes in plane-layered perfectly elastic, isotropic earth models and puts to rest controversies that have arisen with regard to computational stability.

Key words: elastic-wave theory, guided waves, Love waves, Rayleigh waves, surface waves.

1 INTRODUCTION

Present interest in seismic surface-wave theory is largely concerned with departures from the standard model consisting of plane uniform layers under conditions of isotropy and perfect elasticity. Particular examples include anisotropy, visco-elasticity and lateral variations in material properties. In the case of lateral variations, some theoretical progress has been made as portrayed in the works of Woodhouse (1974), Malischewsky (1987) and Keilis-Borok (1989). Weak lateral variations are most conveniently analysed using perturbation theory. Solutions beyond the first order are rarely, if ever, carried out. The zeroth-order solution corresponds to the standard or unperturbed model. The first- and higher-order perturbations solutions depend strongly on the zeroth-order solution. Hence a thorough understanding of the zeroth-order solution, from both the theoretical and computational aspects, is required. This paper is essentially a review of the many different methods used to solve and compute the dispersion characteristics of the standard model described above.

The theory of seismic surface-wave dispersion in plane-layered earth models is well understood. We shall show that the theory, which has evolved from the work of Thomson (1950) and Haskell (1953), and the propagator theory of Gilbert & Backus (1966), leads to an elegant representation for the dispersion equation in the form

$$D(c, k) = \det|U' T V| = 0. \tag{1}$$

U and V are boundary matrices, respectively associated with the top and bottom of the model, and which also depend on the particular boundary conditions applied. T is the propagator matrix which transfers wave information from the top of the model to the bottom, and is the equal to the product $T_1 T_2 \ldots T_n$ (n = no. layers) of the individual layer propagators T_i. The above prescription holds for both L waves (Love), when U, V are 2×1 vectors, T is 2×2 and the determinant is of order 1; and R waves (Rayleigh), when U, V are 4×2 matrices, T is 4×4 and the determinant is of order 2. In addition, eq. (1) applies to models bounded by a free surface or a stack of layers sandwiched between two half-spaces.

In Section 2 we define the notation and state the basic equations. The Thomson–Haskell method, expressed in terms of propagators, is briefly reviewed in Section 3, and the representation above for $D(c, k)$ is derived.

It is also well known that, for R waves, a direct computation of the determinant for $D(c, k)$ may lead to a serious loss of numerical precision at high frequencies (see Fig. 1b). Several

methods have been successfully implemented to solve this instability problem. One popular method was the delta matrix method described in Pestel & Leckie (1963) and first applied to R-wave dispersion by Thrower (1965) and Dunkin (1965). Section 4 describes the basic ideas behind the method and how it avoids the problems of numerical loss of precision. Pestel and Leckie also describe an analytical trick which, in the case of R waves, reduces the order-6 delta matrices to order 5. Since seismic surface-wave applications can be very computationally intensive, significant savings in computer time can be realised using the reduced delta matrix method (see Fig. 4). The reduced delta matrix scheme was first applied to R-wave dispersion by Watson (1970).

A somewhat different approach to dispersion analysis was introduced by Knopoff (1964). It differed from the Haskell–Thomson method in two fundamental ways. First, the dispersion function was expressed in terms of interface matrices, rather than propagators, leading to a determinant of order $2n+1$ for L waves and $4n+2$ for R waves, as opposed to the order-1 and order-2 Thomson–Haskell determinants. Second, the determinant is evaluated using a recursive row decomposition of Laplace's method (see Appendix A8). We shall refer to this recursion as the Knopoff decomposition; it is explained in more detail in Section 5. It transpires that the Knopoff decomposition, which was first computed by Randall (1967), successfully avoids the instability inherent in the Thomson–Haskell method. Unfortunately, the Knopoff decomposition is very cumbersome to apply in practice, and the general complexity of the method substantially increases the chance of programming error.

Further enhancements and numerical studies of the Knopoff decomposition were performed in a series of papers by Schwab & Knopoff (1970, 1972). The common feature of these papers is the Knopoff decomposition for the computation of the dispersion function. The difficulties of this approach, mentioned above, have tended to detract from the real importance of these contributions. One of the purposes of this review is to make the Schwab–Knopoff method more transparent. This is achieved in Section 5.1 by finding explicit matrix transformations that connect the method to the more easily understood delta matrix representation.

Schwab (1970) employed an inspired set of row and column transformations which reduced certain of his layer matrices to almost block diagonal form, but left the dispersion function invariant. As a consequence of these transformations, the dispersion function could be expressed in particularly simple algebraic form. Schwab provided further computational savings by the use of clever factorizations, leading to the most efficient algorithm yet produced. We shall call this algorithm the fast Schwab–Knopoff method.

Section 3.3 provides the general theory of transformations of propagators, of which the fast Schwab–Knopoff method is a special case. In Section 6 we find the explicit matrix transformations that correspond to the fast Schwab–Knopoff method, but which are expressed within the more amenable framework of delta matrix theory. It is therefore logical to call this the *fast delta matrix algorithm*. It turns out to be even more efficient than the fast Schwab–Knopoff algorithm, but it is well to remember that it has its origins in the Schwab (1970) transformation scheme. We point out that the fast algorithms also incorporate the computational advantages of the reduced delta matrix method.

In Section 7 we investigate Abo-Zena's (1979) method to solving the instability problem. Here, the R-wave dispersion function is obtained by recursively computing a sequence of 4×4 antisymmetric matrices, which are closely related to the propagator delta matrices. Abo-Zena goes to considerable lengths to express the propagator as a linear combination of eigenfunction products, in the belief that this is the key to handling the high-frequency instability problem. In fact, this overstates the case, since it is only necessary to ensure that certain combinations of eigenfunctions which generate the instability are correctly computed. These combinations are in fact correctly handled in all the algorithms reviewed in this paper, with the exception, of course, of the Thomson–Haskell method.

In Section 7.1 we apply the Abo-Zena recursion to the 'fast' representation of the dispersion function derived in Section 6. Quite remarkably, this leads to exactly the same representation as the fast delta matrix algorithm, with the factorizations employed by Schwab (1970) automatically carried out in the process.

Section 8 reviews the Reflection–Transmission or RT-matrix method developed by Kennett (1974) and Kennett & Kerry (1979). Unlike the other algorithms, the RT-matrix method requires all computations to be performed using complex arithmetic. This results in the loss of computational efficiency, but there are some compensating features of the method. The principal one is that the RT method neatly side-steps the stability problem and thus avoids the difficulty at the outset. It achieves this through its ingenious representation in terms of reflection and transmission submatrices. Remarkably, this representation depends only on propagating and *decaying* evanescent waves. All dependence on growing evanescent waves, which is present in the other algorithms, is eliminated, and with it the source of the instability.

In order to make the text more readable and still provide a full self-contained theoretical description, most of the mathematical detail is relegated to a sequence of appendices, A1 to A8. Attention is given to unifying the notation as far as possible. Section 2.1 provides a list of the main symbol definitions and notation used throughout the paper.

2 BASIC EQUATIONS

Since the basic theory of surface waves in plane-layered media is well understood, we shall be content here to introduce the notation and state the fundamental equations.

The model contains a stack of n plane horizontal layers, indexed $1, 2, \ldots, n$, with constant elastic parameters. The top of the first layer of the stack will usually correspond to the Earth's surface, assumed to be stress-free. The stack is further assumed to overlie a uniform half-space indexed by layer number $\ell = n+1$. We shall also consider the case of the stack of layers sandwiched between two half-spaces corresponding to buried strata within the Earth, such as a sequence of coal seams. In this case the upper half-space will be indexed by layer number $i=0$ and we shall refer to the top of the first layer as a *buried* surface. A local coordinate system is adopted for each layer with z pointing vertically downwards and the origin at the top of the layer.[1] In this system, each finite layer

[1] In the case of an overlying half-space the origin is placed at the bottom of the half-space, which will then occupy the region $z < 0$.

occupies a local region $0 < z < d_i$, where d_i is the layer thickness. The underlying half-space occupies the local region $z > 0$.

Surface waves propagate horizontally in the x-direction with phase velocity $c = c(k)$ where k is the horizontal wavenumber. The frequency is given by $\omega = ck$. True surface waves satisfy the radiation condition that all wavefields vanish asymptotically as $z \to \pm\infty$ into any half-space. The relation $c = c(k)$ or $c = c(\omega)$ is the dispersion equation we seek. It is obviously dependent on the layer parameters, but is also critically determined by the assumed boundary conditions and continuity conditions at adjoining interfaces. Free mode dispersion corresponds to a configuration in which there are no physical sources in the model. The dispersion exhibits a spectrum which contains both continuous and discrete components. Specifically, the spectrum has a discrete modal structure and each mode is associated with a unique continuous dispersion curve $c = c_m(k)$, where m indexes the mode number and $m = 0$ corresponds to the fundamental mode; $m = 1$ to the first overtone etc. Each mode, except the fundamental, also displays its own cut-off frequency ω_m, below which surface-wave propagation is impossible. For true surface waves, the phase velocity is restricted to be less than the slowest half-space S velocity. Phase velocities violating this condition are pseudo-leaking modes and are not true surface waves. Such waves propagate a component of energy vertically into the half-space(s) and hence violate the radiation condition.

2.1 Notation and parameters

2.1.1 *Dispersion parameters*

k, ω	wave number and frequency
$c = \omega/k$	phase velocity
$D(c, k)$	dispersion function

2.1.2 *Model layer parameters*

$i = 1, 2, \ldots, n$	layer index numbers
$i = 0$	upper half-space index (if it exists)
$\ell = n + 1$	lower half-space index (always exists)
α_i, β_i	P and S velocities in layer-i
$\rho_i, \mu_i = \rho_i\beta_i^2$	density and rigidity in layer-i
d_i	thickness of layer-i
$\gamma_i = \beta_i^2/c^2$	
$t_i = (2 - c^2/\beta_i^2)$	

2.1.3 *Layer eigenfunctions*

	$c < \alpha_i (c < \beta_i)$	$c > \alpha_i (c > \beta_i)$
r_i	$(1 - c^2/\alpha_i^2)^{1/2}$	$i(c^2/\alpha_i^2 - 1)^{1/2} = i\bar{r}_i$
s_i	$(1 - c^2/\beta_i^2)^{1/2}$	$i(c^2/\beta_i^2 - 1)^{1/2} = i\bar{s}_i$
$C_{\alpha_i}(k)$	$\cosh(kr_i d_i)$	$\cos(k\bar{r}_i d_i)$
$S_{\alpha_i}(k)$	$\sinh(kr_i d_i)$	$i\sin(k\bar{r}_i d_i)$
$C_{\beta_i}(k)$	$\cosh(ks_i d_i)$	$\cos(k\bar{s}_i d_i)$
$S_{\beta_i}(k)$	$\sinh(ks_i d_i)$	$i\sin(k\bar{s}_i d_i)$

2.1.4 *Wave-field parameters*

x, z, t	horizontal coordinate (direction of surface-wave propagation); vertical coordinate (positive downwards and locally zero at top of each finite layer); and time
u, v, w	elastic displacement field (functions of x, z, t)
τ^x, τ^y, τ^z	vertical stress field $\sigma^{xz}, \sigma^{yz}, \sigma^{zz}$
A_i, A_i'	P wave up and down amplitudes (complex conjugates if $c > \alpha_i$)
B_i, B_i'	SV wave up and down amplitudes (complex conjugates if $c > \beta_i$)
C_i, C_i'	SH wave up and down amplitudes (complex conjugates if $c > \beta_i$)

2.1.5 *Layer vector and layer matrix definitions*

a_i	up–down amplitude vector for layer-i $[C_i, C_i']'$ for L waves; $[A_i, A_i', B_i, B_i']'$ for R waves
a_ℓ^*	$[C_\ell]$ for L waves; $[A_\ell, B_\ell]'$ for R waves
$y_i(z)$	displacement-stress state vector for layer-i
$\hat{y}_i = y_i(0)$	state vector at top of layer-i
$\check{y}_i = y_i(d_i)$	state vector at bottom of layer-i
M_i	rigidity matrix; diag$[1, \mu_i]$ for L waves and diag$[1, 1, \mu_i, \mu_i]$ for R waves
$E_i(z)$	eigenfunction matrix; diag$[e^{ks_iz}, e^{-ks_iz}]$ for L waves diag$[e^{kr_iz}, e^{-kr_iz}, e^{ks_iz}, e^{-ks_iz}]$ for R waves
P_i, Q_i	layer matrices with $Q_i = M_iP_i$ and $y_i(z) = Q_iE_i(z)a_i$
T_i	$Q_i^{-1}E_i(d_i)Q_i$ layer propagator or transfer matrix

2.1.6 *Other vector and matrix definitions*

T'	denotes the transpose of matrix T
$\bar{T}$	denotes the order-2 delta matrix associated with T
T^*	denotes the reduced delta matrix associated with T
e_j	a length-2 unit column vector with 1 in row j, 0 otherwise
e_{jk}	a 4×2 matrix with 1s in (row j, col 1); (row k, col 2) and 0s elsewhere.

2.2 Love waves

In each layer for $0 < z < d_i$:

$$\frac{\partial^2 v_i}{\partial x^2} + \frac{\partial^2 v_i}{\partial z^2} = \frac{1}{\beta_i^2}\frac{\partial^2 v_i}{\partial t^2},$$

$$v_i(x, z, t) = V_i(z)\cos[k(x - ct)],$$

$$\tau_i^y(x, z, t) = -kY_i(z)\cos[k(x - ct)],$$

$$V_i(z) = C_i e^{-ks_iz} + C_i' e^{ks_iz},$$

$$-kY_i(z) = \mu_i\frac{\partial V_i}{\partial z} = -\mu_i s_i k(C_i e^{-s_ikz} - C_i' e^{s_ikz}).$$

State vector and amplitude vector:

$$y_i(z) = \begin{bmatrix} V_i(z) \\ Y_i(z) \end{bmatrix}; \qquad a_i = \begin{bmatrix} C_i \\ C_i' \end{bmatrix}; \qquad y_i(z) = M_iP_iE_i(-z)a_i, \tag{2}$$

where M_i, P_i, $E_i(z)$ are the 2×2 matrices

$$M_i = \begin{bmatrix} 1 & 0 \\ 0 & \mu_i \end{bmatrix}; \qquad P_i = \begin{bmatrix} 1 & 1 \\ s_i & -s_i \end{bmatrix};$$

$$E_i(z) = \begin{bmatrix} e^{ks_iz} & 0 \\ 0 & e^{-ks_iz} \end{bmatrix}. \tag{3}$$

Surface, interface and radiation conditions can be expressed entirely in terms of state vectors y_i or amplitude vectors a_i. The state-vector representation leads to the usual Thomson–Haskell formalism, while the amplitude representation is adopted in the Knopoff scheme.

2.2.1 *Boundary conditions*

(1) At a stress-free surface: $Y_1(0)=0$ or $e_2' y_1(0)=e_2'\hat{y}_1=0$.

(2) At an internal interface: continuity of displacement and stress means that state vectors at the bottom of layer-i and top of layer-$(i+1)$ are equal, i.e. $y_i(d_i)=y_{i+1}(0)$ or $\check{y}_i=\hat{y}_{i+1}$ for $i=1, 2, \ldots, n$.

(3) In any half-space, ($i=\ell$ and possibly $i=0$): all state vectors must vanish with distance measured away from the finite layers. Thus, for an upper half-space ($i=0$), $C_0=0$; while for a lower half-space ($i=\ell$): $C_\ell'=0$. These are equivalent to: $e_1' a_0=0$ and $e_2' a_\ell=0$ respectively.

It is convenient to define the reduced half-space vector (actually a scalar for L-waves) $a_\ell^*=C_\ell$. Then the lower half-space radiation condition is equivalent to the statement

$$a_\ell=e_1 a_\ell^*=\begin{bmatrix} C_\ell \\ 0 \end{bmatrix}.$$

2.2.2 *Remarks*

(1) The depth dependence of the state vector $y_i(z)$ is contained wholly in the eigenfunction matrix $E_i(-z)$. The choice of $-z$ in the definition of E_i is made for later algebraic convenience. In particular, its inverse is readily seen to be $E_i^{-1}(-z)=E_i(z)$.

(2) The elements of the matrix $E_i(-z)$ are real exponentials when $c<\beta_i$ and are complex exponentials equivalent to sines and cosines when $c>\beta_i$. In the former case, C_i, C_i' are real; in the latter case, they are complex conjugates in order that $V_i(z)$ and $Y_i(z)$ remain real-valued.

(3) When $c=\beta_i$, $s_i=0$ in layer-i, and the eigenfunctions are no longer exponential. The correct solutions are then of the form

$$V_i(z)=C_i+kC_i'z \quad \text{and} \quad Y_i(z)=-\mu_i C_i'.$$

Formally, the matrix P_i defined above becomes singular and the dispersion analysis blows up. This potential problem is solved, however, by replacing P_i and E_i by the matrices

$$P_i=\begin{bmatrix} 1 & kz \\ 0 & k \end{bmatrix}; \qquad E_i=\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

With this representation, it transpires that we need only consider the $c\neq\beta_i$ case. The solution for $c=\beta_i$ then results by taking the limit $s_i\to 0$ in the *final* expressions for the propagator matrices, even though the limit does not exist for matrices in the intermediate calculations.

2.3 Rayleigh waves

In each layer-i:

$$\frac{\partial^2\phi_i}{\partial x^2}+\frac{\partial^2\phi_i}{\partial z^2}=\frac{1}{\alpha_i^2}\frac{\partial^2\phi_i}{\partial t^2},$$

$$\frac{\partial^2\psi_i}{\partial x^2}+\frac{\partial^2\psi_i}{\partial x^2}=\frac{1}{\beta_i^2}\frac{\partial^2\psi_i}{\partial t^2},$$

$$\phi_i(x, z, t)=(A_i\,\mathrm{e}^{-kr_i z}+A_i'\,\mathrm{e}^{kr_i z})\cos[k(x-ct)],$$

$$\psi_i(x, z, t)=(B_i\,\mathrm{e}^{-ks_i z}+B_i'\,\mathrm{e}^{ks_i z})\sin[k(x-ct)],$$

$$u_i(x, z, t)=(\phi_x+\psi_z)_i=-kU_i(z)\sin[k(x-ct)],$$

$$w_i(x, z, t)=(\phi_z-\psi_x)_i=-kW_i(z)\cos[k(x-ct)],$$

$$\tau_i^x(x, z, t)=\mu_i(2\phi_{xz}+\psi_{zz}-\psi_{xx})_i=k^2X_i(z)\sin[k(x-ct)],$$

$$\tau_i^z(x, z, t)=\mu_i(p\phi_{zz}+q\phi_{xx}-2\psi_{xz})_i=k^2Z_i(z)\cos[k(x-ct)],$$

where $p_i=\alpha_i^2/\beta_i^2$ and $q_i=p_i-2$.

The state and amplitude vectors for Rayleigh waves are

$$y_i(z)=\begin{bmatrix} U_i(z) \\ W_i(z) \\ X_i(z) \\ Z_i(z) \end{bmatrix}; \qquad a_i=\begin{bmatrix} A_i \\ A_i' \\ B_i \\ B_i' \end{bmatrix}; \qquad y_i(z)=M_iP_iE_i(-z)a_i, \tag{4}$$

where M_i, P_i, $E_i(z)$ are the 4×4 matrices

$$M_i=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \mu_i & 0 \\ 0 & 0 & 0 & \mu_i \end{bmatrix}; \qquad E_i(z)=\begin{bmatrix} \mathrm{e}^{kr_iz} & 0 & 0 & 0 \\ 0 & \mathrm{e}^{-kr_iz} & 0 & 0 \\ 0 & 0 & \mathrm{e}^{ks_iz} & 0 \\ 0 & 0 & 0 & \mathrm{e}^{-ks_iz} \end{bmatrix};$$

$$P_i=\begin{bmatrix} 1 & 1 & s_i & -s_i \\ r_i & -r_i & 1 & 1 \\ 2r_i & -2r_i & t_i & t_i \\ t_i & t_i & 2s_i & -2s_i \end{bmatrix}. \tag{5}$$

2.3.1 *Boundary conditions*

(1) At a stress-free surface: $X_1(0)=Z_1(0)=0$ which are equivalent to the single vector equation: $e_{34}'\hat{y}_1=0$.

(2) As for L waves, the continuity of displacement and stress across internal interfaces is expressed by: $\check{y}_i=\hat{y}_{i+1}$ for $i=1, 2, \ldots, n$.

(3) Radiation conditions for half-spaces are given by $A_0=B_0=0$ for $i=0$ (upper half-space) and $A_\ell'=B_\ell'=0$ for $i=\ell$ (lower half-space). These are equivalent to $e_{13}'a_0=0$ and $e_{24}'a_\ell=0$ respectively.

Again, we shall find it convenient to define the reduced amplitude vector $a_\ell^*=[A_\ell, B_\ell]'$. Then the lower half-space radiation condition can be expressed as

$$a_\ell=e_{13}a_\ell^*=\begin{bmatrix} A_\ell \\ 0 \\ B_\ell \\ 0 \end{bmatrix}.$$

Remarks similar to those made for Love waves also apply to Rayleigh waves. In particular, the cases $c=\alpha_i$ and $c=\beta_i$ require special attention, since the eigenfunctions are then no longer exponential functions in layer-i. The correct solutions are

$$\phi_i(z)=(A_i+kA_i'z)\cos[k(x-ct)] \quad \text{if} \quad c=\alpha_i,$$

$$\psi_i(z)=(B_i+kB_i'z)\sin[k(x-ct)] \quad \text{if} \quad c=\beta_i.$$

The corresponding adjustments to matrices E_i and P_i are then as follows

For $c = \alpha_i$: take $E_i(-z) = \text{diag}[1, 1, e^{-s_i kz}, e^{s_i kz}]$ and replace the second column of P_i by the vector $[kz, -1, -2t_i, kz]'$.

When $c = \beta_i$: take $E_i(-z) = \text{diag}[e^{-r_i kz}, e^{r_i kz}, 1, 1]$ and replace the fourth column of P_i by the vector $[-1, kz, kz, -2]'$.

These adjustments then lead to the correct limiting form of the dispersion equation when $c \to \alpha_i$ or $c \to \beta_i$.

3 PROPAGATOR MATRICES AND THE THOMSON–HASKELL METHOD

We see from the previous section that both Love and Rayleigh waves are governed by matrix equations of the form

$$y_i(z) = Q_i E_i(-z) a_i, \tag{6}$$

where Q_i is a new layer matrix defined by $Q_i = M_i P_i$. Since M_i is a diagonal matrix, Q_i is obtained by multiplying each row of P_i by the corresponding diagonal element of M_i. The layer propagator (or transfer) matrix $T_i(z)$ is defined by

$$T_i(z) = Q_i E_i(z) Q_i^{-1}. \tag{7}$$

The components of T_i are listed in the Appendix for both Love (A1) and Rayleigh (A2) waves. All the elements of T_i are seen to be real, even though the elements of Q_i and $E_i(z)$ may be complex. Two elementary but important properties of $T_i(z)$ are the following. For any z_1 and z_2,

$$T_i(z_1) T_i(z_2) = T_i(z_1 + z_2) \quad \text{and} \quad T_i^{-1}(z) = T_i(-z).$$

There results are also useful for checking the computer code.

If z_1 and z_2 denote any two depths within the same layer, then

$$y_i(z_1) = Q_i E_i(-z_1) a_i \quad \text{and} \quad y_i(z_2) = Q_i E_i(-z_2) a_i.$$

Eliminating a_i from these two matrix equations then leads to

$$y_i(z_1) = T_i(z_2 - z_1) y_i(z_2). \tag{8}$$

The propagator $T_i(z_2 - z_1)$ therefore relates state vectors at any two depths within the same layer. To determine how state vectors from different layers are related, define two other vectors for each layer:

$$\hat{y}_i = y_i(0) = Q_i a_i \quad \text{and} \quad \check{y}_i = y_i(d_i) = Q_i E_i(-d_i) a_i. \tag{9}$$

Thus $\hat{y}_i$ and $\check{y}_i$ denote the state vector at the top and bottom of the current layer. It follows that $\hat{y}_i = T_i(d_i)\check{y}_i = T_i \check{y}_i$, where $T_i = T_i(d_i) = Q_i E_i Q_i^{-1}$ and $E_i = E_i(d_i)$. The continuity condition for state vectors implies that at the interface between layers i and $(i+1)$ we have $\check{y}_i = \hat{y}_{i+1}$. This immediately leads to the Thomson–Haskell recursion

$$\hat{y}_i = T_i \hat{y}_{i+1}, \quad i = 1, 2, \ldots, n, \tag{10}$$

with solution

$$\hat{y}_1 = (T_1 T_2 \ldots T_n) \hat{y}_\ell \quad (\ell = n + 1). \tag{11}$$

This means that the state vector at the top of the first layer is related to the state vector at the top of the lower half-space through the product $T = T_1 T_2 \ldots T_n$ of all layer propagator matrices.

To obtain the dispersion equation, it remains only to apply the surface condition and radiation condition to $\hat{y}_1$ and $\hat{y}_\ell$. In all cases, the boundary conditions lead to a matrix equation of the form $U' T V a_\ell^* = 0$. The dispersion equation therefore has the implicit form

$$D(c, k) = \det|U' T V| = 0, \tag{12}$$

where U, V are boundary matrices (listed below) which depend on the type of surface wave (Love or Rayleigh) and also the type of surface condition (free surface or buried surface). For L waves, U, V are 2×1 matrices and so the determinant is of order 1 and therefore redundant. For R waves, U, V are 4×2 matrices and the determinant is of order 2.

For Love waves

Free surface: $U' = e'_2 = [0 \quad 1]$.

Buried surface: $U' = e'_1 Q_0^{-1} = \frac{1}{2}[1 \quad \mu s]_0$.

Bottom half-space: $V = Q_\ell e_1 = \begin{bmatrix} 1 \\ \mu s \end{bmatrix}_\ell$.

For Rayleigh waves

Free surface: $U' = e'_{34} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$.

Buried surface: $U' = e'_{13} Q_0^{-1}$

$$= \tfrac{1}{2}\gamma_0 \begin{bmatrix} 2 & -t/r & 1/\mu r & -1/\mu \\ -t/s & 2 & -1/\mu & 1/\mu s \end{bmatrix}_0.$$

Lower half-space: $V = Q_\ell e_{13} = \begin{bmatrix} 1 & s \\ r & 1 \\ 2\mu r & \mu t \\ \mu t & 2\mu s \end{bmatrix}_\ell$.

Note that multiplication of a matrix Q on the left by e'_{ij} selects rows i and j of Q, while multiplication on the right by e_{ij} selects columns i and j of Q.

Within the theoretical framework developed above, it is a simple matter to find the dispersion equation for standard Rayleigh waves on a half-space and Stoneley waves at an interface separating two half-spaces. In both cases there are no intervening layers ($n = 0$), and the dispersion equation reduces to $D = \det|U' V| = 0$.

Specifically,

$$D_R(c) = \det|e'_{34} Q_1 e_{13}| = 0$$

for Rayleigh waves, and

$$D_S(c) = \det|e'_{13} Q_0^{-1} Q_1 e_{13}| = 0$$

for Stoneley waves. These are readily reduced to standard formulas which turn out to be non-dispersive (independent of k). Rayleigh waves, for example, have the dispersion equation $4 r_1 s_1 - t_1^2 = 0$, which expands to the classical formula

$$4(1 - c^2/\alpha_1^2)^{1/2}(1 - c^2/\beta_1^2)^{1/2} - (2 - c^2/\beta_1^2)^2 = 0.$$

3.1 Low- and high-frequency solutions

Low frequencies correspond to the limit $k \to 0$. In this case the layer eigenfunction matrix $E_i \to I$, the identity matrix, and hence $T_i \to Q_i I Q_i^{-1} = I$ also. It follows that the dispersion equation simplifies to $D(c, 0) = \det|U' V| = 0$. However, as seen above, this is the dispersion equation for the case when there are no intervening layers. Thus for free-surface Rayleigh waves,

the phase velocity in the low-frequency limit will tend to the lower half-space Rayleigh velocity, just as if the layers were not present. In effect the wavelengths are so large that the finite layers become 'invisible'.

In the high-frequency limit, we first consider the case of a very thick layer-j with $c < \beta_j$ so that the exponential eigenfunctions in E_j dominate all other terms in the dispersion equation. We may then approximate E_j by the diagonal matrix

$$E_j \approx \text{diag}[e^{kr_jd_j}, 0, e^{ks_jd_j}, 0] = \text{diag}[R, 0, S, 0],$$

where R, S are real and numerically large. In this case, the dispersion function can be factorized as

$$D(c, k) = RSD_1(c, k)D_2(c, k) = 0.$$

Here, $D_1 = \det|U'(T_1 \dots T_{j-1})V_j|$ is the dispersion function for layers 1 to j overlying a half-space with the properties of layer-j, and $D_2 = \det|U_j'(T_{j+1} \dots T_n)V_\ell|$ is the dispersion function for layers $(j+1)$ to n, sandwiched between two half-spaces with properties corresponding to layer-j and layer-ℓ respectively. Thus the dispersion function has decoupled into two terms corresponding to a free-surface Rayleigh wave for the first $(j-1)$ finite layers, and a buried surface Stoneley wave for layers $(j+1)$ to n. The thick layer-j effectively acts as a separating half-space when the frequency is sufficiently high.

We may follow Haskell (1953) and extend this argument to higher and higher frequencies, when all the finite layers behave as effective half-spaces for the short wavelengths inferred. Now the dispersion function factorizes into n terms. The first corresponds to a free-surface Rayleigh wave with the properties of the first layer. The remaining factors correspond to Stoneley waves (if they actually exist for the given model parameters) at each successively deeper interface.

3.2 Numerical considerations

We begin this section with a brief statement of a computational algorithm to implement the Thomson–Haskell recursion of the dispersion function $D(c, k)$, given in eq. (1).

$$X_1 = U'; \qquad X_{i+1} = X_i T_i \quad (i = 1, 2, \dots, n);$$

$$D = \det|X_\ell V| \quad (\ell = n+1). \tag{13}$$

This algorithm requires a subroutine to compute the propagator matrix T_i for each layer in the model for any selected values of the phase velocity c and wavenumber k. The intermediate matrix X_i is 1×2 (Love) or 2×4 (Rayleigh) and contains the successive products of the propagators. Note that, correspondingly, only 1×2 or 2×4 matrix multiplications are required, rather than 2×2 and 4×4 operations. For models with many layers, the computation of the propagator matrix is the most time-consuming. Considerable savings in computer time can be made by efficiently coding the propagator elements, using its many symmetry properties (see Appendix A2). Since the algorithm delivers only the dispersion function, the user must also provide code for extracting its zeros. In most practical applications, a simple bracketing method with linear interpolation will be sufficient. More accurate methods are available if desired. There are arguments for scanning the roots for either fixed c or fixed k (Kerry 1981), but we have avoided the issue in this paper by computing binary dispersion images, rather than specific dispersion curves.

The problem of numerical overflow due to the exponentials appearing in the E_i matrices is solved by properly scaling the elements of the propagator matrix. Underflow is avoided by zeroing all elements below a minimum threshold. Although this scaling will change the numerical value of the dispersion function, it will not effect the location of its zeros. Further details are provided in Section 9.

The problem of loss of numerical precision is, however, a far more serious one. The problem does not exist for Love waves, and appears only for Rayleigh waves, as illustrated in Fig. 1. The Thomson-Haskell algorithm generally works very well only up to a finite model-dependent $k_{\max}$ (or equivalent maximum frequency $\omega_{\max}$), beyond which severe precision loss is encountered. The reasons for this instability have been thoroughly discussed in the referenced works. It is due to the subtraction, in the computation of the determinant, of very large terms which are identically equal in theory, but may differ numerically. In particular, consider a layer with the Rayleigh-wave eigenfunction matrix

$$E_j = \text{diag}[R, R^{-1}, S, S^{-1}]; \qquad R = e^{kr_jd_j}, \qquad S = e^{ks_jd_j},$$

with R, S assumed real and numerically large. This matrix will generate a final Thomson–Haskell (2×2) dispersion matrix $U'TV$, whose elements are linear combinations of (R, R^{-1}, S, S^{-1}). Hence $D(c, k)$, the determinant of this matrix, will contain large terms of order R^2, S^2 and RS. However, as will be demonstrated in the next section, the terms in R^2 and S^2 should cancel identically. Owing to numerical round-off, exact cancellation fails, leading to the loss of precision alluded to, possibly with catastrophic results, as displayed in Fig. 1(b).

3.3 Transformations of propagators

Let S_i $(i = 1, 2, \dots, \ell)$ be any non-singular matrices with the same dimensions as the propagators T_i. The elements of S_i may depend on the layer-i properties. Let $\hat{z}_i = S_i\hat{y}_i$. Then the basic surface-wave recursion equation (10):

$$\hat{y}_i = T_i\hat{y}_{i+1} \quad (i = 1, 2, \dots, n);$$

$$U'\hat{y}_1 = 0, \qquad \hat{y}_\ell = Va_\ell^* \tag{14}$$

may be transformed to

$$\hat{z}_i = \tilde{T}_i\hat{z}_{i+1} \quad (i = 1, 2, \dots, n);$$

$$\tilde{U}'\hat{z}_1 = 0, \qquad \hat{z}_\ell = \tilde{V}a_\ell^*, \tag{15}$$

where

$$\tilde{T}_i = S_iT_iS_{i+1}^{-1}; \qquad \tilde{U}' = U'S_1^{-1}; \qquad \tilde{V} = S_\ell V. \tag{16}$$

With these definitions,

$$\begin{aligned}\tilde{U}'\tilde{T}\tilde{V} &= \tilde{U}'(\tilde{T}_1\tilde{T}_2 \dots \tilde{T}_n)\tilde{V} \\ &= (U'S_1^{-1})(S_1T_1S_2^{-1})(S_2T_2S_3^{-1}) \dots (S_nT_nS_\ell^{-1})(S_\ell V) \\ &= U'TV.\end{aligned}$$

Hence the dispersion function $D = \det|U'TV|$ remains invariant under the transformations S_i. Theoretically, the two recursions are equivalent.

We find in Section 5 such transformations connecting the delta matrix and Schwab–Knopoff algorithms, thereby demonstrating the equivalence of the two methods.

The question arises, however, of whether suitable transformations can be found which lead to significant computational improvement. We shall find, in Section 6, matrices S_i that transform the standard delta matrix recursion into a new *fast*

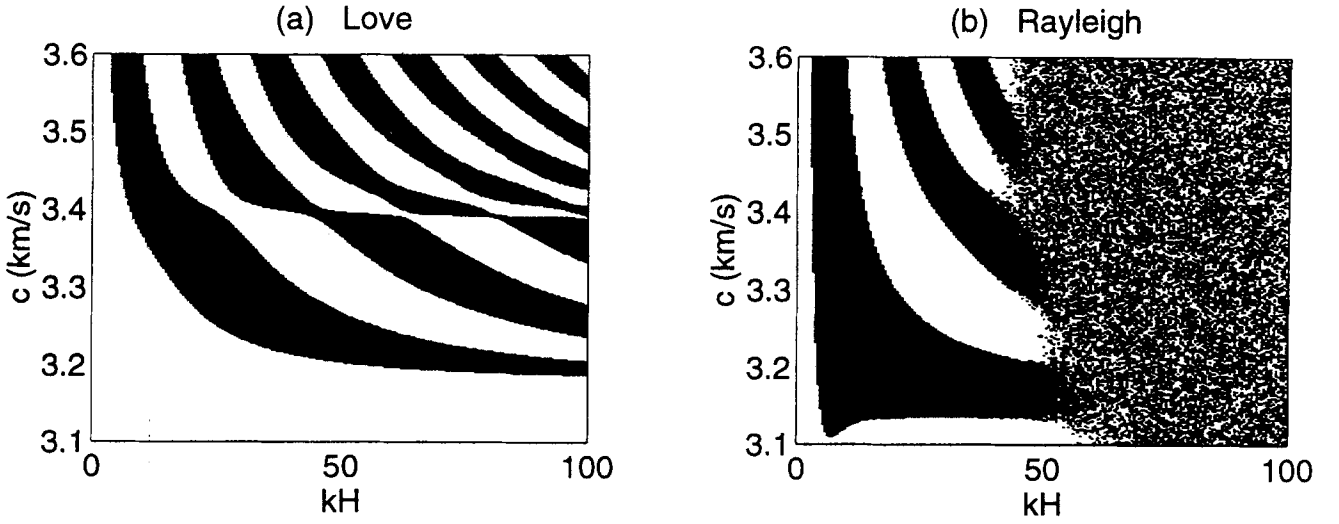

Figure 1. Love- and Rayleigh-wave dispersion functions $D(c, k)$ computed using the Thomson–Haskell method for Haskell's two-layer model-H1 (Haskell 1953; Case-1, p. 31).

Layer	α (km/s)	β (km/s)	ρ (g/cc)	d (km)
1	6.14	3.39	2.70	13.60
2	5.50	3.18	2.70	11.85
$\frac{1}{2}$-space	8.26	4.65	3.00	∞

The binary images show $D > 0$ (dark) and $D < 0$ (light), so that actual dispersion curves $D = 0$ lie on the edges. Catastrophic precision loss occurs in the Thomson–Haskell R-wave scheme at a normalized wavenumber $kH \approx 50$, where H is the total thickness of the layers. No instability is evident for L waves.

recursion, similar to the fast version of the Schwab–Knopoff algorithm.

One apparent disadvantage of such transformed schemes is that the new 'propagator', $\tilde{T}_i = S_i T_i S_{i+1}^{-1}$, depends on the properties of *both* layer-i and layer-$(i+1)$. As such, $\tilde{T}_i$ is actually an interface matrix, connecting wave properties between two adjacent layers. We shall see that this disadvantage leads to unnecessary complications in the original Knopoff scheme. Fortunately, it does not affect the fast Schwab–Knopoff and fast delta matrix schemes.

4 DELTA MATRIX METHOD

Several methods to handle the precision problem mentioned in the last section have been proposed. Although not the first to appear in the seismological literature, the delta matrix method (see Pestel & Leckie 1963) is probably the simplest to understand. Let A be any matrix, not necessarily square. Its corresponding order-p delta matrix (also called compound matrix), denoted in this paper by $\bar{A}$, is defined to be the strictly ordered matrix of all order-p minors of A. That is, every element of $\bar{A}$ is a minor or determinant of order p obtained by selecting all possible sets of p rows and p columns of the original matrix A. The element $\bar{A}_{ij}$ is the minor constructed from the ith row set and jth column set. The strict ordering of elements is defined by row and column ranks relative to the original matrix A. Thus, if A is 4×4 and $p = 2$, the six rows of $\bar{A}$ correspond (in order) to the row sets

(12)(13)(14)(23)(24)(34)

of the original matrix A. The same ordering applies to the columns.

If A has dimension $m \times n$, then clearly $\bar{A}$ has dimension ${}^mC_p \times {}^nC_p$, where ${}^nC_p$ denotes the usual binomial coefficients and p must not exceed $\min(m, n)$. The delta matrix of order p for any given A is clearly unique.

We shall need a number of well-known properties of delta matrices, which we are content to state here without proof. They are:

(1) if A' denotes the transpose of A, then $\overline{A'} = \bar{A}'$;
(2) $\overline{A^{-1}} = \bar{A}^{-1}$;
(3) if $A = BC$, then $\bar{A} = \bar{B}\bar{C}$.

In the application of delta matrix theory to Rayleigh waves, we are concerned with boundary matrices U, V of dimension 4×2 and propagators T of dimension 4×4. We therefore select $p = 2$ and observe that $\bar{U}, \bar{V}$ have dimension 6×1, while $\bar{T}$ has dimension 6×6. As noted above, the index set (1, 2, 3, 4, 5, 6) for each row and/or column of an order-2 delta matrix corresponds to the set of index pairs (12, 13, 14, 23, 24, 34) of the original matrix. Thus, for example, the element

$$\bar{T}_{25} = \det \begin{bmatrix} T_{12} & T_{14} \\ T_{32} & T_{34} \end{bmatrix} = T_{12}T_{34} - T_{14}T_{32}.$$

When the Thomson–Haskell dispersion equation is expressed in terms of order-2 delta matrices, the result is (using the three

delta matrix properties):

$$D(c, k) = \bar{U}' \bar{T} \bar{V} = 0, \tag{17}$$

where

$$\bar{T} = \bar{T}_1 \bar{T}_2 \ldots \bar{T}_n; \qquad \bar{T}_i = \bar{Q}_i \bar{E}_i \bar{Q}_i^{-1}. \tag{18}$$

A complete list of the 36 elements of $\bar{T}_i$ is given in Appendix A3. Note that the matrix product above for $D(c, k)$ is symbolically $(1 \times 6)(6 \times 6) \ldots (6 \times 6)(6 \times 1)$, so that the final result is an order-1 determinant or simply a scalar.

We now investigate how the delta matrix method avoids the loss-of-precision problem inherent in the Thomson–Haskell method. Consider the offending layer defined in Section 3.1, with eigenfunction matrix $E_j = \text{diag}[R, R^{-1}, S, S^{-1}]$. Its corresponding delta matrix is readily calculated as the sixth-order diagonal matrix

$$\bar{E}_j = \text{diag}[1, RS, RS^{-1}, SR^{-1}, (RS)^{-1}, 1].$$

Note the important point that this does not depend on R^2 or S^2. Thus the delta matrix formulation of the Thomson–Haskell solution automatically avoids the troublesome high-order terms. One can now use the computational algorithm given for the Thomson–Haskell method, but with all matrices replaced by their corresponding delta matrices.

It may appear that the size of the dispersion calculation has been considerably escalated in the delta matrix approach, but this is an illusion. First, of the 36 elements of each layer propagator $\bar{T}_i$, only 15 are actually independent, due to symmetry (see Appendix A3). Second, the delta matrix algorithm performs only (1×6) by (6×6) matrix multiplications for each layer iteration. This requires only 36 scalar multiplications per iteration, compared to 32 in the original (2×4) by (4×4) Thomson–Haskell scheme.

Further symmetries in $\bar{T}_i$ make it is possible to reduce the order-6 matrices to order-5. The resulting algorithm then only requires 25 scalar multiplications per layer iteration. These improved algorithms are called *reduced* delta matrix methods and are considered in detail in the next section.

4.1 Reduced delta matrix method

The ideas behind this method can be found in Pestel & Leckie (1963), but were first applied to Rayleigh-wave dispersion by Watson (1970). The method exploits symmetries in the delta matrix propagator elements to reduce the sixth-order calculations to fifth-order. For models with many layers or many c, k pairs, the saving in computation time can be significant, as depicted in Fig. 4.

Let $\bar{T}_{ij}$ for $i, j = 1, 2, \ldots, 6$ denote the delta matrix elements of any layer propagator matrix (as listed in Appendix A3). Examination of the second and fifth rows and columns of $\bar{T}_{ij}$ identifies the following two symmetry properties:

$$S1: \quad \bar{T}_{i2} + \bar{T}_{i5} = [0, 1, 0, 0, 1, 0]',$$

$$S2: \quad \bar{T}_{2j} - \bar{T}_{5j} = [2\bar{T}_{21}, (2\bar{T}_{22} - 1), 2\bar{T}_{23}, 2\bar{T}_{24}, (1 - 2\bar{T}_{22}), 2\bar{T}_{26}].$$

These two properties are the key elements in the construction of the reduced delta matrix algorithm. To see how this comes about, consider the delta matrix recursion

$$\bar{X}_1 = \bar{U}'; \qquad \bar{X}_{i+1} = \bar{X}_i \bar{T}_i; \qquad D = \bar{X}_\ell \bar{V}. \tag{19}$$

The second step is repeated for as many layers as are present in the model. $\bar{U}'$, $\bar{X}_i$ and $\bar{V}'$ are 1×6 row vectors with components u_j, x_j and v_j (say) respectively. Then, regardless of the boundary conditions selected, we always have

$$u_2 + u_5 = 0, \qquad x_2 + x_5 = 0, \qquad v_2 + v_5 = 0.$$

Inspection of $\bar{U}'$ and $\bar{V}$ (whose elements are also listed in Appendix A3) confirms the first and third equations above. The first symmetry property S1 of $\bar{T}$ above, ensures that $x_2 + x_5 = 0$ for all layer iterations. To prove this, let

$$y_j = (\bar{X}\bar{T})_j = \sum_{i=1}^{6} x_i \bar{T}_{ij}$$

denote the j-element of the matrix product $\bar{X}\bar{T}$ where $x_2 + x_5 = 0$. Then, using property S1,

$$y_2 + y_5 = \sum_{i=1}^{6} x_i(\bar{T}_{i2} + \bar{T}_{i5}) = x_2 + x_5 = 0$$

completes the proof.

The reduced matrix elements are now defined by the following rules.

(1) For all row vectors $\bar{R} = [r_1, r_2, r_3, r_4, r_5, r_6]$ (including those that make up $\bar{T}$) define the reduced row vector $R^* = [r_1, r_2, r_3, r_4, r_6]$, obtained by dropping the fifth term.

(2) For all column vectors $\bar{C} = [c_1, c_2, c_3, c_4, c_5, c_6]'$ (including those that make up $\bar{T}$) define the reduced column vector $C^* = [c_1, (c_2 - c_5), c_3, c_4, c_6]'$, obtained by dropping the fifth term and replacing the second by the difference $(c_2 - c_5)$.

With these rules, we obtain the following reduced delta matrix elements:

$$U^{*\prime} = [u_1, u_2, u_3, u_4, u_6], \qquad X_i^* = [x_1, x_2, x_3, x_4, x_6],$$

$$V^* = [v_1, 2v_2, v_3, v_4, v_6]'$$

and

$$T^* = \begin{bmatrix} \bar{T}_{11} & \bar{T}_{12} & \bar{T}_{13} & \bar{T}_{14} & \bar{T}_{16} \\ 2\bar{T}_{21} & 2\bar{T}_{22} - 1 & 2\bar{T}_{23} & 2\bar{T}_{24} & 2\bar{T}_{26} \\ \bar{T}_{31} & \bar{T}_{32} & \bar{T}_{33} & \bar{T}_{34} & \bar{T}_{36} \\ \bar{T}_{41} & \bar{T}_{42} & \bar{T}_{43} & \bar{T}_{44} & \bar{T}_{46} \\ \bar{T}_{61} & \bar{T}_{62} & \bar{T}_{63} & \bar{T}_{64} & \bar{T}_{66} \end{bmatrix}. \tag{20}$$

In this construction, we have used the fact that $v_5 = -v_2$, and the second row of T^* above is obtained from symmetry property S2 above. This agrees with Watson's result, once allowance for a change in row and column indexing is made.

It remains to show that, when the Thomson–Haskell recursion is applied to the reduced delta matrices, the original dispersion function is obtained. This is demonstrated as follows. First,

$$\begin{aligned} (\bar{X}\bar{T})_j &= \sum_{i=1}^{6} x_i \bar{T}_{ij} \quad (j \neq 5) \\ &= x_1 \bar{T}_{1j} + x_2(\bar{T}_{2j} - \bar{T}_{5j}) + x_3 \bar{T}_{3j} + x_4 \bar{T}_{4j} + x_6 \bar{T}_{6j} \\ &= \sum_{i=1}^{5} x_i^* T_{ik}^* \quad (k = 1, 2, \ldots, 5) \\ &= (X^* T^*)_k, \end{aligned}$$

and so the (1×6) by (6×6) layer transfer products are preserved in the reduced (1×5) by (5×5) products.

Second, the dispersion function has the representation

$$D = X_\ell \tilde{V} = \sum_{i=1}^{6} x_i v_i$$

$$= x_1 v_1 + 2x_2 v_2 + x_3 v_3 + x_4 v_4 + x_6 v_6 \quad \text{since } x_5 v_5 = x_2 v_2$$

$$= \sum_{i=1}^{5} x_i^* v_i^* = X_\ell^* V^* = D^*.$$

and this completes the proof.

5 SCHWAB-KNOPOFF METHOD

This method is based on Knopoff's (1964) representation of surface-wave dispersion. The method was first programmed for numerical computation by Randall (1967), and later developed and improved by Schwab (1970) and Schwab & Knopoff (1972). The original Knopoff (1964) approach differs from the Thomson–Haskell method in two significant ways.

The first departure occurs in the choice of layer transfers. Knopoff expressed these in terms of the amplitude vectors a_i, in place of the state vectors $\hat{y}_i$. Since $\hat{y}_i = Q_i a_i$, however, the basic layer recursion equation (10) can be expressed as

$$Q_i a_i = R_i a_{i+1} \quad (i = 1, 2, \ldots, n);$$

$$\tilde{U}' a_1 = 0; \qquad a_\ell = \tilde{V} a_\ell^*, \tag{21}$$

where $\tilde{U}' = U' Q_1^{-1}$, $\tilde{V} = Q_\ell^{-1} V$ and $R_i = T_i Q_{i+1}$ is an interface matrix, depending on layer properties on both sides of the interface-i. Knopoff writes the recursion as a single matrix equation of the form

$$K \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \\ a_\ell \end{bmatrix} = \begin{bmatrix} \tilde{U}' & & & & \\ Q_1 & -R_1 & & & \\ & Q_2 & -R_2 & & \\ & & \ddots & & \\ & & & Q_n & -R_n^* \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \\ a_\ell \end{bmatrix} = 0, \tag{22}$$

with $R_n^* = R_n \tilde{V}$. For L waves, the non-zero blocks for each row of K have dimensions: (1×2) for $\tilde{U}'$; (2×4) for each $[Q_i, -R_i]$ and (2×3) for the last block $[Q_n, -R_n^*]$. The R-wave dimensions are exactly twice those of the L waves, i.e. (2×4); (4×8) and (4×6) respectively.

The dispersion equation is therefore

$$D = \det|K| = 0. \tag{23}$$

It is at this point that a second significant departure is made in the original Knopoff scheme and all subsequent Schwab–Knopoff schemes as well. The determinant is evaluated using a row decomposition[2] of Laplace's Method (see Appendix A8). The procedure begins by developing minors of K in the rows of the first block $\tilde{U}'$; each corresponding complementary minor is then developed in the rows of the next block $[Q_1, -R_1]$; then $[Q_2, -R_2]$ etc. until the final block $[Q_n, -R_n^*]$ is reached. We shall call this procedure the Knopoff decomposition. It results in the following matrix product for the determinant (e.g. Schwab & Knopoff 1972):

$$\det|K| = K_0 \mathring{K}_1 K_2 \mathring{K}_3 \ldots K_n^*, \tag{24}$$

[2] Actually, a column decomposition leads to a simpler representation.

where K_n^* denotes K_n if n is even, or $\mathring{K}_n$ if n is odd. The product contains two types of matrices, K_i and $\mathring{K}_i$, whose elements are obtained as certain minors of the blocks $\Gamma_i = [Q_i, -R_i]$ which appear in K.

Love waves

Let Γ_{ab}^i denote the order-2 minor of $\Gamma_i = [Q_i, -R_i]$ constructed from column-a of Q_i and column-b of $(-R_i)$. The structure of K implies that no other column combinations need be considered. Then

$$K_i = \begin{bmatrix} \Gamma_{11} & \Gamma_{12} \\ \Gamma_{21} & \Gamma_{22} \end{bmatrix}_i; \qquad \mathring{K}_i = \begin{bmatrix} -\Gamma_{22} & \Gamma_{21} \\ \Gamma_{12} & -\Gamma_{11} \end{bmatrix}_i. \tag{25}$$

The boundary matrices are

$$K_0 = \tilde{U}'; \qquad K_n^* = \begin{bmatrix} \Gamma_{11} \\ \Gamma_{21} \end{bmatrix}_{n(\text{even})} \text{ or } \begin{bmatrix} \Gamma_{21} \\ -\Gamma_{11} \end{bmatrix}_{n(\text{odd})}. \tag{26}$$

The Knopoff decomposition of $\det|K|$ for L-waves is therefore the symbolic matrix multiplication of order $(1 \times 2)(2 \times 2) \ldots (2 \times 2)(2 \times 1)$. The final result is, of course, a scalar.

Rayleigh waves

Let Γ_{abcd}^i denote the order-4 minor of $\Gamma_i = [Q_i, -R_i]$ constructed from columns-(a, b) of Q_i and columns-(c, d) of $(-R_i)$. Again, the structure of K requires no other column combinations. Then

$$K_i = \begin{bmatrix} \Gamma_{1212} & \Gamma_{1213} & \Gamma_{1214} & \Gamma_{1223} & \Gamma_{1224} & \Gamma_{1234} \\ \Gamma_{1312} & \Gamma_{1313} & \Gamma_{1314} & \Gamma_{1323} & \Gamma_{1324} & \Gamma_{1334} \\ \Gamma_{1412} & \Gamma_{1413} & \Gamma_{1414} & \Gamma_{1423} & \Gamma_{1424} & \Gamma_{1434} \\ \Gamma_{2312} & \Gamma_{2313} & \Gamma_{2314} & \Gamma_{2323} & \Gamma_{2324} & \Gamma_{2334} \\ \Gamma_{2412} & \Gamma_{2413} & \Gamma_{2414} & \Gamma_{2423} & \Gamma_{2424} & \Gamma_{2434} \\ \Gamma_{3412} & \Gamma_{3413} & \Gamma_{3414} & \Gamma_{3423} & \Gamma_{3424} & \Gamma_{3434} \end{bmatrix}_i, \tag{27}$$

$$\mathring{K}_i = \begin{bmatrix} \Gamma_{3434} & -\Gamma_{3424} & \Gamma_{3423} & \Gamma_{3414} & -\Gamma_{3413} & \Gamma_{3412} \\ -\Gamma_{2434} & \Gamma_{2424} & -\Gamma_{2423} & -\Gamma_{2414} & \Gamma_{2413} & -\Gamma_{2412} \\ \Gamma_{2334} & -\Gamma_{2324} & \Gamma_{2323} & \Gamma_{2314} & -\Gamma_{2313} & \Gamma_{2312} \\ \Gamma_{1434} & -\Gamma_{1424} & \Gamma_{1423} & \Gamma_{1414} & -\Gamma_{1413} & \Gamma_{1412} \\ -\Gamma_{1334} & \Gamma_{1324} & -\Gamma_{1323} & -\Gamma_{1314} & \Gamma_{1313} & -\Gamma_{1312} \\ \Gamma_{1234} & -\Gamma_{1224} & \Gamma_{1223} & \Gamma_{1214} & -\Gamma_{1213} & \Gamma_{1212} \end{bmatrix}_i \tag{28}$$

Denoting the boundary matrices by $\Gamma^0 = \tilde{U}'$ (2×4) and $\Gamma_n^* = [Q_n, -R_n^*]$ (4×6) we obtain

$$K_0 = [\Gamma_{12}^0, \Gamma_{13}^0, \Gamma_{14}^0, \Gamma_{23}^0, \Gamma_{24}^0, \Gamma_{34}^0], \tag{29}$$

consisting of order-2 minors only; and

$$K_n^* = \begin{bmatrix} \Gamma_{1212}^* \\ \Gamma_{1312}^* \\ \Gamma_{1412}^* \\ \Gamma_{2312}^* \\ \Gamma_{2412}^* \\ \Gamma_{3412}^* \end{bmatrix}_{n(\text{even})}, \text{ or } \begin{bmatrix} \Gamma_{3412}^* \\ -\Gamma_{2412}^* \\ \Gamma_{2312}^* \\ \Gamma_{1412}^* \\ -\Gamma_{1312}^* \\ \Gamma_{1212}^* \end{bmatrix}_{n(\text{odd})}. \tag{30}$$

The Knopoff decomposition of $\det|K|$ for R waves is therefore the symbolic matrix multiplication of order $(1\times 6)(6\times 6)\ldots(6\times 6)(6\times 1)$, and the final result is again a scalar.

The computation of the order-4 minors Γ^i_{abcd}, which constitute the elements of K_i and $\mathring{K}_i$, represents a formidable task when the original Knopoff representation $\Gamma_i=[Q_i, -R_i]$ is used. However, considerable simplification is achieved by rewriting the Knopoff recursion $Q_i a_i - R_i a_{i+1}=0$ in the equivalent form $a_i - S_i a_{i+1}=0$ with $S_i = Q_i^{-1}R_i = Q_i^{-1}T_i Q_{i+1}$. The interface matrix S_i now corresponds to the transformed propagator $\tilde{T}_i$ considered in Section 3.3, with transformation matrix Q_i. With this modification, the Γ_i blocks of matrix K reduce to the simpler form

$$\Gamma_i=[I, -S_i], \tag{31}$$

where I is a 2×2 identity matrix for L waves, and a 4×4 identity matrix for R waves. In this representation, the L-wave order-2 minors Γ^i_{ab} reduce to simple scalars; the R-wave order-4 minors Γ^i_{abcd} reduce to more manageable order-2 minors. For example, the minor Γ_{1212} reduces as follows:

$$\Gamma_{1212}=\det\begin{bmatrix}1 & 0 & -S_{11} & -S_{12}\\ 0 & 1 & -S_{21} & -S_{22}\\ 0 & 0 & -S_{31} & -S_{32}\\ 0 & 0 & -S_{41} & -S_{42}\end{bmatrix}=\det\begin{bmatrix}S_{31} & S_{32}\\ S_{41} & S_{42}\end{bmatrix}.$$

Similar expressions exist for all the other order-4 minors.

In fact, there is no reason why we cannot simplify the Knopoff method even further by starting from the Thomson–Haskell recursion, $\hat{y}_i = T_i\hat{y}_{i+1}$, and use the Knopoff decomposition of K to obtain the dispersion function $D=\det|K|$. In this representation, K has the same block structure as before, but with Γ_i replaced by

$$\Gamma_i=[I, -T_i]. \tag{32}$$

Since, the propagators T_i depend only on the current layer-i properties, the analysis is indeed simplified.

In the case of L waves, we find a simple connection between the matrices K_i, $\mathring{K}_i$ and the propagators T_i:

$$K_i = J'T_i; \qquad \mathring{K}_i = T_i J; \qquad J=\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}. \tag{33}$$

The boundary matrices are $K_0=U'$ and $K^*_n = J^*V$, where $J^*=I$ the 2×2 identity matrix if n is even, and $J^*=J'$ if n is odd.

The Knopoff decomposition for the dispersion function now becomes

$$\begin{aligned}\det|K| &= U'(T_1J)(J'T_2)(T_3J')\ldots T_nJ^*V\\ &= U'(T_1T_2\ldots T_n)V.\end{aligned}$$

This is seen to agree exactly with the standard Thomson–Haskell result, when the condition $JJ'=I$ is applied.

5.1 Relation to delta matrices

We have seen that the Knopoff decomposition for R waves leads to the symbolic representation $\det|K|=(1\times 6)(6\times 6)\ldots(6\times 6)(6\times 1)$, which is precisely the same as the delta matrix representation. We may therefore expect a close relationship between the two methods, and this is now discussed in detail. Let $\bar{T}_i$ denote the 6×6 delta matrix associated with T_i and discussed in Section 4. Then we find, similar to the L-wave case,

$$K_i = J\bar{T}_i; \qquad \mathring{K}_i = \bar{T}_i J;$$

$$J=\begin{bmatrix}&&&&&1\\ &&&&-1&\\ &&&1&&\\ &&1&&&\\ &-1&&&&\\ 1&&&&&\end{bmatrix}; \qquad JJ=I. \tag{34}$$

In addition, the boundary matrices are $K_0=\bar{U}'$ and $K^*_n = J^*\bar{V}$, with $J^*=I$ (the sixth-order identity) if n is even and $J^*=J$ if n is odd.

In this case, the Knopoff decomposition for $\det|K|$ reduces to the standard delta matrix representation eq. (17):

$$\begin{aligned}\det|K| &= \bar{U}'(\bar{T}_1J)(J\bar{T}_2)(\bar{T}_3J)\ldots(J^*\bar{V})\\ &= \bar{U}'(\bar{T}_1\bar{T}_2\ldots\bar{T}_n)\bar{V}.\end{aligned}$$

This, of course, explains why Knopoff's decomposition also handles the loss-of-precision problem inherent in the R-wave Thomson–Haskell scheme.

If we follow through all the transformations from the original Knopoff scheme, where $\Gamma_i=[Q_i, -R_i]$, leading up to the delta matrix scheme, we find:

$$K_i = |Q_i|\bar{Q}_i^{-1}\bar{T}_i\bar{Q}_{i+1}J; \qquad \mathring{K}_i = |Q_i|J\bar{Q}_i^{-1}\bar{T}_i\bar{Q}_{i+1}, \tag{35}$$

where $|Q_i|=\det|Q_i|$.

We are left with the conclusion that there is no real advantage of the Knopoff decomposition over the standard delta matrix method. In fact, the opposite is probably true. Both handle the loss-of-precision problem equally well. However, the Knopoff decomposition is considerably more difficult to implement than the delta matrix method. Even when we employ the transformation that reduces order-4 minor calculations to order-2, we are still left with the need to compute the two matrix types K_i and $\mathring{K}_i$, which depend on layer-i and layer-$(i+1)$ properties. On the other hand, the delta matrix method requires only one type of layer matrix, $\bar{T}_i$, which depends only on current layer parameters.

6 FAST DISPERSION FUNCTION ALGORITHMS

In Schwab (1970), a considerable effort is made to develop an efficient algorithm in terms of computational speed. Since many surface-wave applications are computationally intensive, significant savings can be made. In particular, Schwab was able to perform row and column transformations on the R-wave propagator which considerably simplified its algebraic form. The resulting algorithm exploits this in two ways. First, computational speed is enhanced by the need to evaluate simpler matrix elements; and second, the non-zero elements can be factorized for optimal speed of computation, through a significant reduction in the number of multiplications per layer iteration. Since the Knopoff decomposition of the dispersion function is still employed in this fast version of the algorithm, we refer to the scheme as the *fast* Schwab–Knopoff method.

In this section, we find the explicit matrix transformations that reduce the propagators T_i to the simple form discovered by Schwab. We shall use this representation to find the corresponding delta matrix expression for the dispersion function. In this way, the main elements of the fast Schwab–Knopoff method become more transparent, without the difficulties of the Knopoff decomposition mentioned above.

Our starting point is the observation that the matrix

$$Y_i = \gamma_i \begin{bmatrix} 2 & 0 & 0 & -1 \\ 0 & t_i & -1 & 0 \\ 0 & 2 & -1 & 0 \\ -t_i & 0 & 0 & 1 \end{bmatrix}; \qquad \gamma_i^{-1} = 2 - t_i = c^2/\beta_i^2 \tag{36}$$

transforms the layer matrix P_i defined by eq. (5) to block diagonal form. Specifically,

$$Z_i = Y_i P_i = \begin{bmatrix} 1 & 1 & 0 & 0 \\ -r_i & r_i & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & s_i & -s_i \end{bmatrix}. \tag{37}$$

Since r_i and s_i respectively depend only on the layer velocities α_i and β_i, the transformation Y_i effectively decouples the P and S waves in each layer.

The propagator $T_i = M_i P_i E_i P_i^{-1} M_i^{-1}$ can therefore be written as $T_i = M_i Y_i^{-1} F_i Y_i M_i^{-1}$, where F_i is the block diagonal matrix

$$F_i = Z_i E_i Z_i^{-1} = \begin{bmatrix} C_{\alpha_i} & -\frac{1}{r_i} S_{\alpha_i} & 0 & 0 \\ -r_i S_{\alpha_i} & C_{\alpha_i} & 0 & 0 \\ 0 & 0 & C_{\beta_i} & \frac{1}{s_i} S_{\beta_i} \\ 0 & 0 & s_i S_{\beta_i} & C_{\beta_i} \end{bmatrix}. \tag{38}$$

Following the notation of Section 3.3, we now choose

$$S_i = Y_i M_i^{-1} \tag{39}$$

to transform the standard Thomson–Haskell scheme. The propagator T_i is then transformed into the interface matrix

$$\tilde{T}_i = S_i T_i S_{i+1}^{-1} = F_i L_i, \tag{40}$$

with

$$L_i = (YM^{-1})_i (MY^{-1})_{i+1} = \begin{bmatrix} a_i & 0 & 0 & -b_i' \\ 0 & b_i & a_i' & 0 \\ 0 & b_i' & a_i & 0 \\ -a_i' & 0 & 0 & b_i \end{bmatrix}. \tag{41}$$

The layer parameters (a_i, a_i', b_i, b_i'), defined below, depend on the properties of layer-i and layer-$(i+1)$ and the phase velocity c. For $i = 1, 2 \ldots, n$, define

$$\varepsilon_i = \rho_{i+1}/\rho_i; \qquad \eta_i = 2(\gamma_i - \varepsilon_i \gamma_{i+1}) = 2(\beta_i^2 - \varepsilon_i \beta_{i+1}^2)/c^2. \tag{42}$$

Then, after some elementary algebra, we obtain

$$a_i = \varepsilon_i + \eta_i; \qquad a_i' = a_i - 1; \qquad b_i = 1 - \eta_i; \qquad b_i' = b_i - 1. \tag{43}$$

L_i may be regarded as the matrix that couples the P and S waves across interface-i.

With this representation, we find

$$\tilde{T}_i = (FL)_i = \begin{bmatrix} aC_\alpha & -bS_\alpha/r & -a'S_\alpha/r & -b'C_\alpha \\ -arS_\alpha & bC_\alpha & a'C_\alpha & b'rS_\alpha \\ -a'S_\beta/s & b'C_\beta & aC_\beta & bS_\beta/s \\ -a'C_\beta & b'sS_\beta & asS_\beta & bC_\beta \end{bmatrix}_i, \tag{44}$$

which is indeed much simpler than the original untransformed propagator T_i, listed in Appendix A2, despite the fact that $\tilde{T}_i$ depends on layer-i and layer-$(i+1)$ properties.

To complete the description, we must also evaluate the transformed boundary matrices. We obtain

$$\tilde{U}' = U' S_1^{-1} = U' M_1 Y_1^{-1}; \qquad \tilde{V} = S_\ell V = Y_\ell M_\ell^{-1} V, \tag{45}$$

which for R-waves on a free surface reduce to

$$\tilde{U}' = \mu_1 \begin{bmatrix} 0 & -2 & t_1 & 0 \\ t_1 & 0 & 0 & 2 \end{bmatrix}; \qquad \tilde{V} = Z_\ell e_{13} = \begin{bmatrix} 1 & 0 \\ -r_\ell & 0 \\ 0 & 1 \\ 0 & s_\ell \end{bmatrix}. \tag{46}$$

In order to control the instability problem, Schwab (1970) employed the Knopoff decomposition to obtain the dispersion function D. However, the delta matrix method achieves the same goal, and, for reasons stated previously, is computationally preferable. We therefore adopt this approach here, which leads to a new algorithm which may be referred to as the *fast delta matrix algorithm.*

The resulting representation for $D(c, k)$ has many similarities with the fast Schwab–Knopoff solution. Details of the fast delta matrix elements are given in Appendix A4. In Appendix A5, we present the results of the corresponding fast delta matrix recursion:

$$\bar{X}_1 = \bar{U}'; \qquad \bar{X}_{i+1} = \bar{X}_i \bar{T}_i; \qquad D = \bar{X}_\ell \bar{V}, \tag{47}$$

taking care, as did Schwab (1970), to optimize the algorithm for computational speed by exploiting common factors wherever they occur. We also identify in A5 how this scheme is modified to accommodate the reduced matrix elements.

Despite the obvious similarities with the fast Schwab–Knopoff scheme, the present approach has a number of advantages. First it is algebraically simpler and more transparent, making it easier to see precisely how it is related to the original Thomson–Haskell scheme in terms of propagator matrices. Second, although both handle the stability problem equally well (see Figs 2 and 3), the new fast delta matrix algorithm is even more efficient than the fast Schwab–Knopoff scheme (see Fig. 4).

7 ABO-ZENA METHOD

Abo-Zena (1979) has suggested another approach to surface-wave dispersion. Although the validity of some of the claims made in this paper must be considered doubtful, the method adopted gives further insight into R-wave dispersion analysis. The basic idea of the method is to write the dispersion function

$D = \det|U'TV|$ in the equivalent 2×2 form

$$D = \det\begin{bmatrix} u_1'Tv_1 & u_1'Tv_2 \\ u_2'Tv_1 & u_2'Tv_2 \end{bmatrix}, \tag{48}$$

where u_1, u_2 and v_1, v_2 are 4×1 vectors which constitute the columns of U and V respectively. That is, $U = [u_1, u_2]$ and $V = [v_1, v_2]$. Note that each term $u_i'Tv_j$ in the above determinant is a scalar. Hence

$$\begin{aligned} D &= (u_1'Tv_1)(u_2'Tv_2) - (u_1'Tv_2)(u_2'Tv_1) \\ &= v_1'(T'X_1T)v_2, \end{aligned} \tag{49}$$

where $X_1 = (u_1u_2' - u_2u_1')$ is a 4×4 antisymmetric matrix. For notational convenience, define

$$\text{asym}[x_1, x_2, x_3, x_4, x_5, x_6] = \begin{bmatrix} 0 & x_1 & x_2 & x_3 \\ -x_1 & 0 & x_4 & x_5 \\ -x_2 & -x_4 & 0 & x_6 \\ -x_3 & -x_5 & -x_6 & 0 \end{bmatrix}. \tag{50}$$

With this notation, we find for free-surface R waves:

$$X_1 = \text{asym}[0, 0, 0, 0, 0, 1] = \text{asym}(\bar{U}'), \tag{51}$$

and, since $T = T_1T_2 \ldots T_n$, this leads to the recursion

$$X_1 = \text{asym}(\bar{U}'); \qquad X_{i+1} = T_i'X_iT_i; \qquad D = v_1'X_fv_2. \tag{52}$$

This looks like an inefficient algorithm because of the triple matrix products. In fact, from a numerical viewpoint it is only more efficient than the RT-algorithm (see Fig. 4). Nevertheless, some simplifications can be made. First, since $T'XT$ is antisymmetric whenever X is, we may conclude that X_i is antisymmetric for all $i = 1, 2, \ldots, n$, and therefore has only six independent components.

The algorithm cannot be computed as it stands because of the instability problem. Abo-Zena solves the problem as follows. Recall that $T_i = (QEQ^{-1})_i$; so, if we define p_a and q_a' as the ath column of Q_i and ath row of Q_i^{-1} $(a = 1, 2, 3, 4)$ so p_a, q_a are 4×1 vectors, and also define

$$E_i = \text{diag}[e^{krd}, e^{-krd}, e^{ksd}, e^{-ksd}]_i = \text{diag}[f_1, f_2, f_3, f_4] \tag{53}$$

(f_a are scalars), then T_i has the representation

$$T_i = \sum_{a=1}^{4} f_a p_a q_a'. \tag{54}$$

Hence

$$X_{i+1} = T_i'X_iT_i = \sum_{a,b=1}^{4} f_af_bq_a(p_a'X_ip_b)q_b'. \tag{55}$$

This sum has 16 terms, but since X_i is antisymmetric, the products $p_a'X_ip_b$ (scalars) are identically zero whenever $a = b$, and equal to $-p_b'X_ip_a$ when $a \neq b$. We thus obtain the simpler representation with only six terms:

$$X_{i+1} = \sum_{a<b} f_af_b(p_a'X_ip_b)[q_aq_b' - q_bq_a']. \tag{56}$$

One important observation, noted by Abo-Zena, is that this representation is independent of the products $f_a^2 = (e^{\pm 2r_ikd_i}, e^{\pm 2s_ikd_i})$, which give rise to the numerical instability encountered in the Thomson–Haskell scheme. Since they are obviated in the Abo-Zena scheme, instability will not arise, just as in the delta matrix and Schwab–Knopoff schemes.

Indeed, it is not difficult to prove that the representation eq. (56) above, for X_{i+1}, is equivalent to the delta matrix representation

$$\bar{X}_{i+1} = \bar{X}_i\bar{T}_i; \qquad \bar{T}_i = (\bar{Q}\bar{E}\bar{Q}^{-1})_i, \tag{57}$$

with $X_i = \text{asym}(\bar{X}_i)$, and this establishes the connection between the two methods.

Abo-Zena makes much of the fact that this algorithm computes X_{i+1} by factoring out all the frequency-dependent eigenfunctions (f_a, f_b above). In particular, it is claimed that this is the key feature in solving the numerical instability problem. A close examination, however, reveals that this is not the main reason. The instability problem is solved by ensuring that the eigenfunction products f_a^2 occurring in eq. (52) are correctly computed. All the methods discussed in this paper, other than the Thomson–Haskell, correctly handle the troublesome products.

It is therefore possible to apply the Abo-Zena recursion in a somewhat simpler manner. From Appendix A2 we see that each propagator T_i can be written in the form $T_i = A + B$, where A is a 4×4 matrix with elements proportional to either C_{α_i} or S_{α_i}; and B is a similar matrix with terms in C_{β_i} or S_{β_i}. The Abo-Zena recursion can therefore be expanded in that form

$$X_{i+1} = (A' + B')X_i(A + B). \tag{58}$$

The reason for the instability lies in the fact that the matrix products in this recursion must be independent of terms in $C_{\alpha_i}^2$, $S_{\alpha_i}^2$, $(C_{\alpha_i}S_{\alpha_i})$ and $C_{\beta_i}^2$, $S_{\beta_i}^2$, $(C_{\beta_i}S_{\beta_i})$. It is important to realise that this does not imply that $A'X_iA$ and $B'X_iB$ are zero, since many non-zero terms arise from the factors $C_{\alpha_i}^2 - S_{\alpha_i}^2 = 1$ and $C_{\beta_i}^2 - S_{\beta_i}^2 = 1$. Analytic expressions for $A'XA$ and $B'XB$ can be found in Appendix A6.

Since X_i is antisymmetric, the recursion can now be written as

$$X_{i+1} = A'X_iA + B'X_iB + A'X_iB - (A'X_iB)'. \tag{59}$$

This recursion will not suffer from the instability problem if the exact analytic expressions for $A'X_iA$ and $B'X_iB$ are used, instead of computing them numerically. Details of this modified recursion can also be found in Appendix A6.

7.1 Fast extension of the Abo-Zena method

Here we apply the Abo-Zena recursion to the fast transformation of the dispersion function developed in Section 6. Under this transformation,

$$D = \tilde{U}'(\tilde{T}_1\tilde{T}_2 \ldots \tilde{T}_n)\tilde{V},$$

with $\tilde{T}_i = F_iL_i$ (see eq. 40). The Abo-Zena recursion then becomes

$$X_{i+1} = \tilde{T}_i'X_i\tilde{T}_i = [L'(F'XF)L]_i. \tag{60}$$

The matrix products in this representation are easily evaluated analytically. In fact, in the notation of Appendix A5, we obtain

$$\begin{aligned} X_i &= \text{asym}[x_1, x_2, x_3, x_4, x_5, x_6], \\ \tilde{X}_i &= F_i'X_iF_i = \text{asym}[x_1, q_1, q_2, q_3, q_4, x_6] \quad (x_1 = x_6), \\ X_{i+1} &= L_i'\tilde{X}_iL_i \\ &= \text{asym}[b'y_1 + by_2, ay_1 + a'y_2, \varepsilon q_3, \varepsilon q_4, b'z_1 + bz_2, az_1 + a'z_2] \\ &= \text{asym}[\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4, \hat{x}_5, \hat{x}_6]. \end{aligned}$$

Remarkably, this is exactly the same algorithm as given in Appendix A5, but is generated here in a far simpler and more direct manner. The instability problem has been eliminated. Furthermore, the efficient Schwab factorization, which previously required considerable effort, emerges automatically. The decomposition in eq. (60) effectively performs the factorization as part of the process.

8 RT MATRIX ALGORITHM

The reflection–transmission (RT) matrix method (Kennett 1974, Kennett & Kerry 1979) solves the stability problem for the dispersion function $D(c, k)$ in a completely different manner. In order to see how this is achieved, we begin with the Thomson–Haskell scheme for $D(c, k) = \det(U'TV)$. When fully written out for the case of a free-surface problem, we obtain

$$D(c, k) = \det[e'_{12}(Q_1 E_1 Q_1^{-1})(Q_2 E_2 Q_2^{-1}) \ldots (Q_n E_n Q_n^{-1}) Q_\ell e_{34}]. \tag{61}$$

A regrouping of terms allows the equivalent representation

$$D(c, k) = \det[e'_{12} Q_1 (E_1 Q_1^{-1} Q_2)(E_2 Q_2^{-1} Q_3) \ldots (E_n Q_n^{-1} Q_\ell) e_{34}]. \tag{62}$$

This has the form $D(c, k) = \det(U'SV)$, where $U' = e'_{12}Q_1$, $V = e_{34}$, $S = S_1 S_2 \ldots S_n$ and

$$S_i = E_i Q_i^{-1} Q_{i+1} = E_i \bar{S}_i; \qquad \bar{S}_i = Q_i^{-1} Q_{i+1}. \tag{63}$$

The propagator matrix $T_i = Q_i E_i Q_i^{-1}$ relates stress–displacement vectors through $\hat{y}_i = T_i \hat{y}_{i+1}$. The interface matrix S_i, on the other hand, relates amplitude vectors through $a_i = S_i a_{i+1}$. The diagonal matrix $E_i = E(d_i)$ is the phase operator for waves in layer-i. We shall therefore refer to $\bar{S}_i$ as the *unphased* interface matrix and to S_i as the corresponding *phased* interface matrix.

In the RT matrix method, the amplitude vector a_i is partitioned into order-2 up and down components:[3]

$$a_i = \begin{bmatrix} a_u \\ a_d \end{bmatrix}_i; \qquad a_u = \begin{bmatrix} A'_i \\ B'_i \end{bmatrix}; \qquad a_d = \begin{bmatrix} A_i \\ B_i \end{bmatrix}.$$

The corresponding 2×2 partitions of $S_i = E_i \bar{S}_i$ are

$$\begin{bmatrix} S_{uu} & S_{ud} \\ S_{du} & S_{dd} \end{bmatrix}_i = \begin{bmatrix} E_u & 0 \\ 0 & E_d \end{bmatrix}_i \begin{bmatrix} \bar{S}_{uu} & \bar{S}_{ud} \\ \bar{S}_{du} & \bar{S}_{dd} \end{bmatrix}_i = \begin{bmatrix} E_u \bar{S}_{uu} & E_u \bar{S}_{ud} \\ E_d \bar{S}_{du} & E_d \bar{S}_{dd} \end{bmatrix}_i. \tag{64}$$

The elements of these up and down partitions are listed in Appendix A7. In this representation,

$$E_u = \begin{bmatrix} e^{-kr_i d_i} & 0 \\ 0 & e^{-ks_i d_i} \end{bmatrix} \quad \text{and} \quad E_d = \begin{bmatrix} e^{kr_i d_i} & 0 \\ 0 & e^{ks_i d_i} \end{bmatrix} = E_u^{-1}.$$

Thus E_u, E_d are the up and down phase operators for waves in layer-i. For phase velocities $c < \beta_i$, the P and S waves are both propagating and the phase operators are complex-valued. For phase velocities $c > \alpha_i$, the P and S waves are both evanescent and the phase operators are real-valued. It is clear that the numerical stability problem has its origin in the $E_d = E_u^{-1}$ sub-matrix when the waves are evanescent. In this case, the exponential terms are real and may numerically swamp the ensuing recursion.

The partitions of the unphased interface matrix $\bar{S}_i$ have two symmetry properties which are used to improve computational efficiency. Let A^* denote the matrix obtained from any 2×2 matrix A by reversing the sign of its non-diagonal elements. Then, in the present representation, it transpires that

$$\bar{S}_{uu} = \bar{S}^*_{dd} \quad \text{and} \quad \bar{S}_{ud} = \bar{S}^*_{du}. \tag{65}$$

The order-2 unphased reflection and transmission matrices associated with $\bar{S}_i$ are defined by (Kennett & Kerry 1979):

$$\left.\begin{aligned} \bar{T}_d &= \bar{S}_{dd}^{-1} & \bar{S}_{uu} &= \bar{T}_u - \bar{R}_d \bar{T}_d^{-1} \bar{R}_u \\ \bar{R}_d &= \bar{S}_{ud} \bar{S}_{dd}^{-1} & \bar{S}_{ud} &= \bar{R}_d \bar{T}_d^{-1} \\ \bar{T}_u &= \bar{S}_{uu} - \bar{S}_{ud} \bar{S}_{dd}^{-1} \bar{S}_{du} & \bar{S}_{du} &= -\bar{T}_d^{-1} \bar{R}_u \\ \bar{R}_u &= -S_{dd}^{-1} \bar{S}_{du} & \bar{S}_{dd} &= \bar{T}_d^{-1} \end{aligned}\right\}. \tag{66}$$

There is an analogous set of relations for the phased interface sub-matrices ($S_{uu}, S_{ud}, S_{du}, S_{dd}$) in terms of corresponding phased RT matrices (T_d, R_d, T_u, R_u). The connection between the phased and unphased RT matrices, obtained from eq. (64) above, is:

$$T_d = \bar{T}_d E; \qquad R_d = E \bar{R}_d E; \qquad T_u = E \bar{T}_u; \qquad R_u = \bar{R}_u, \tag{67}$$

where $E = E_u$.

Remarkably, as noted by Kennett & Kerry (1979), the phased RT matrices depend only on E_u and not on $E_d = E_u^{-1}$. Hence, in this description, all the evanscent waves are exponentially decaying. The source of the numerical stability problem is therefore circumvented.

The recursion for the RT matrices has been shown by Kennett & Kerry (1979) to be expressible in the form

$$\left.\begin{aligned} R_d^{ac} &= R_d^{ab} + T_u^{ab} R_d^{bc} (I - R_u^{ab} R_d^{bc})^{-1} T_d^{ab} \\ T_d^{ac} &= T_d^{bc} (I - R_u^{ab} R_d^{bc})^{-1} T_d^{ab} \\ R_u^{ac} &= R_u^{bc} + T_d^{bc} R_u^{ab} (I - R_d^{bc} R_u^{ab})^{-1} T_u^{bc} \\ T_u^{ac} &= T_u^{ab} (I - R_d^{bc} R_u^{ab})^{-1} T_u^{bc} \end{aligned}\right\}, \tag{68}$$

where I denotes the 2×2 identity matrix. The superscripts indicate that RT matrices for layers between $z_a < z < z_c$ can be constructed from the RT matrices for $z_a < z < z_b$ and $z_b < z < z_c$. The recursion can be performed either upwards from the half-space towards the free-surface, or downwards in the opposite direction. It transpires that there is considerable computational advantage in performing the recursion upwards, starting from the last finite layer and adding one layer at a time until the surface layer is reached. In practice, it is most efficient to compute the above recursion using the unphased RT matrices for the layer being added (those with the *ab* superscripts). The updated RT matrices can then be correctly phased by applying the phase operators of eq. (67) corresponding to the layer being added, *after* each recursion step. One reason for doing this lies in the observation that the only place that the wavenumber explicitly appears is in the phase operators. Hence the unphased RT matrices for a given phase velocity c can be

[3] This partitioning requires a trivial re-ordering of our matrix elements as described in Appendix A7.

used without the need to recompute them for each new value of the wavenumber k.

The symmetry relations eq. (65) above have their counterpart for the RT matrices, which can be expressed as

$$R_u T_d^* + T_d R_d^* = 0, \quad \text{and} \quad T_u T_d^* + R_d R_d^* = I. \tag{69}$$

These are somewhat different to the relations derived by Kennett, Kerry & Woodhouse (1978) due to different normalizations adopted for the matrix representations.

In the RT representation, the dispersion function takes the form (Kennett & Kerry 1979)

$$D(c, k) = \det[(N_d + N_u R_d^{0n})(T_d^{0n})^{-1}], \tag{70}$$

where in our notation,

$$N_d = e'_{34} Q_1 e_{13} = \mu_1 \begin{bmatrix} 2r_1 & t_1 \\ t_1 & 2s_1 \end{bmatrix}$$

$$N_u = e'_{34} Q_1 e_{24} = \mu_1 \begin{bmatrix} -2r_1 & t_1 \\ t_1 & -2s_1 \end{bmatrix}. \tag{71}$$

The $0n$ superscripts indicate that the RT matrices are computed for the entire stack of layers between the free-surface $z = 0$ and the half-space $z = z_n$. These are, of course, the results of the full recursion described above.

We make a number of points regarding the implementation of the RT algorithm.

(1) Even though the above representation for $D(c, k)$ is real-valued, all computations necessarily require complex arithmetic. This is in direct contrast to all the other algorithms considered in this paper, for which the computations can be organized in such a way that only real arithmetic is required. This feature has the biggest impact on the efficiency of the RT method for surface-wave computations in perfectly elastic media.

(2) If one is interested only in the zeros of $D(c, k)$, it is possible to ignore the factor $(T_d^{0n})^{-1}$, particularly since its determinant can lead to underflow problems when the frequency is high. However, it is useful to keep the factor to ensure that $D(c, k)$ remains real-valued. Indeed, it is only necessary to keep its phase in the form

$$\arg(\det[(T_d^{0n})^{-1}]).$$

The underflow problem is then eliminated. We have adopted this strategy in our implementation of the RT algorithm, by developing a separate recursion for the phase of $\det(T_d)$. This recursion is easily derived from equations given above.

(3) The RT algorithm gives a dispersion image (Fig. 2) that is identical to those we have computed for all the other algorithms. However, it is seen to be the least efficient (Fig. 4).

(4) It may be remarked that, even though the RT algorithm

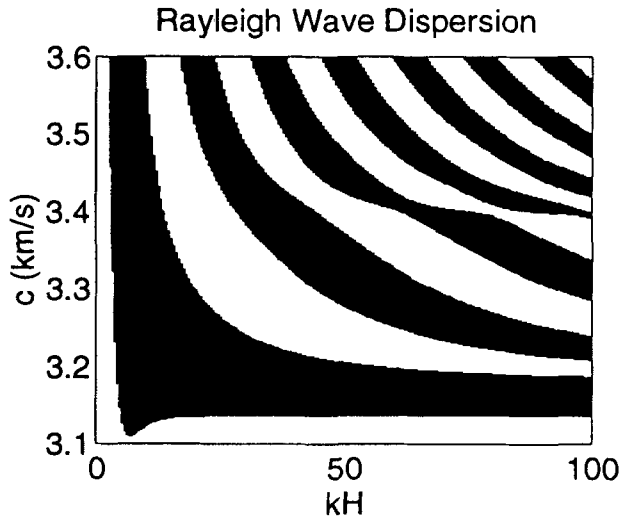

Figure 2. R-wave $D(c, k)$ images for the same model as in Fig. 1, obtained with the reduced delta matrix, fast Schwab–Knopoff, Abo-Zena, fast delta matrix and Kennett's RT matrix algorithms. The images are identical to the naked eye and there is, of course, no evidence of numerical instability.

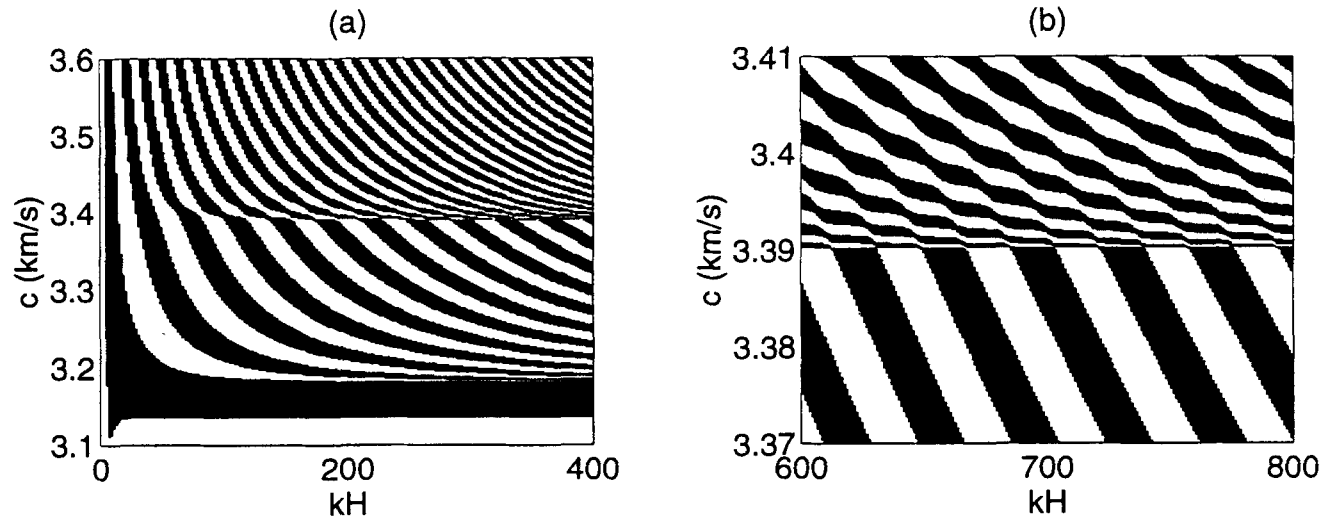

Figure 3. R-wave $D(c, k)$ images for the same model as in Fig. 1 over two high-frequency ranges: (a) shows the separation of crustal and channel modes; (b) shows details of the separation near the phase velocity $c = 3.39$ (km s^{-1}). Both images were computed with the new fast delta matrix algorithm, but the same images are also produced by the other stable algorithms.

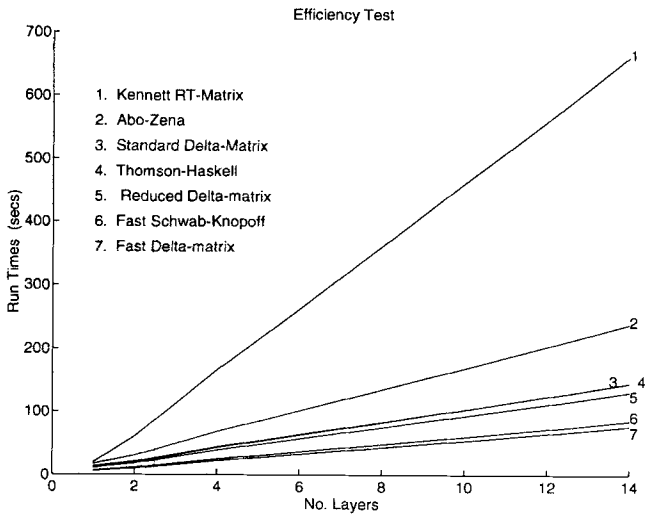

Figure 4. Comparison of run times for computing $D(c, k)$ dispersion images of size 300 × 200 pixels on a PC486 DX2-66. Run times increase approximately linearly with the number of layers. The most efficient is the new fast delta matrix algorithm, being about 12 per cent faster than the fast Schwab–Knopoff algorithm as described in Schwab Knopoff (1972, p. 126). The reduced delta matrix algorithm has a similar improvement over the standard delta matrix method. There is little noticeable difference between Thomson–Haskell and the standard delta matrix algorithm up to 14 layers, although we expect the Thomson–Haskell to be relatively faster when the number of layers is increased further.

is the least efficient of the algorithms considered here, this need not be the case for viscoelastic and anisotropic media. For viscoelastic media, all algorithms will require complex arithmetic. In the case of anisotropic media, Love and Rayleigh waves do not decouple, so the propagator and interface matrices expand to dimension 6. The corresponding order-2 delta matrices would then have dimension 15, a considerable increase over the dimension-6 delta matrices required for isotropic media. Since the RT algorithm circumvents the need to implement special stability methods, it is likely to be the preferred method for anisotropic media.

9 SCALING FOR OVERFLOW AND UNDERFLOW

The layer eigenfunctions C_α, S_α and C_β, S_β can lead to numerical overflow when $c < \alpha < \beta$ and the frequency is high enough. The problem is treated in detail in Schwab *et al.* (1984) and Panza (1985), but is easily understood as follows. In the range $c < \alpha$, the eigenfunctions are hyperbolic, so we have

$$C_\alpha = \tfrac{1}{2}(e^{\zeta_\alpha} + e^{-\zeta_\alpha}) \quad \text{and} \quad S_\alpha = \tfrac{1}{2}(e^{\zeta_\alpha} - e^{-\zeta_\alpha}),$$

where $\zeta_\alpha = kr_i d_i$. Similar expressions hold for C_β and S_β. There exists a machine-dependent $\zeta_{\max}$ such that, if $\zeta_\alpha > \zeta_{\max}$, the exponentials suffer numerical overflow and underflow.[4] Numerically, C_α and S_α become indistinguishable from $e^{\zeta_\alpha}/2$.

One approach to the problem is to reduce ζ_α by dividing thick layers into a number of thinner layers. Schwab *et al.* (1984) have explored this idea and found that it had only limited success. There exists, however, a simpler procedure which works for all practical cases. The propagator elements for a layer in which the overflow problem occurs can be scaled by the large and positive factor $C_\alpha \approx S_\alpha$. Although this scaling changes the value of the dispersion function $D(c, k)$, it will not alter its zeros. It has no effect at all on the binary dispersion function images displayed in this paper. Nevertheless, care must be exercised to ensure that the removal of the overflow problem in this manner does not result in a corresponding underflow problem. This will occur for propagator elements with components independent of C_α and S_α. The underflow is avoided by setting these components equal to zero. Thus a propagator element of the form $aC_\alpha + bS_\alpha + c$ is replaced by $a + b$ after scaling.

There are several such propagator elements in the delta matrix, fast delta matrix and Abo-Zena algorithms. They are readily identified for the delta matrix elements in Appendix A3. In the fast delta matrix algorithm (Appendix A5), scaling will remove all terms in x_1 (and x_6). In the Abo-Zena recursion (Appendix A6), all the g_i factors, corresponding to the term $A'X_iA + B'X_iB$, will be removed when scaling is applied. All figures displayed in this paper have been produced with scaling of the above type.

[4] We have found $\zeta_{\max} \approx 80$ to be reasonable and have used this value in all images displayed in this paper.

10 CONCLUSION

We have reviewed in this paper several methods for computing seismic surface wave dispersion functions. These are:

(1) Thomson–Haskell
(2) Standard delta matrix
(3) Reduced delta matrix
(4) Schwab–Knopoff
(5) Fast Schwab–Knopoff
(6) Fast delta matrix
(7) Abo-Zena
(8) Kennett RT matrix.

All the methods, bar the first and last, were designed (to some extent) to handle the numerical instability problem inherent in the original Thomson–Haskell method. We have found analytical relations connecting these methods and discussed how each solves the instability problem. We have programmed the different algorithms and compared their outputs for a range of models (see Figs 2 and 3). We conclude from this study that although there are significant differences in computational efficiency, as illustrated in Fig. 4, each solves the problem of numerical precision loss completely. Scaling for overflow/underflow must also be applied, as described in Section 9, if dispersion for unlimited frequencies is required.

We have also presented details of a new fast delta matrix algorithm in Appendix A5. This algorithm is derived from a transformation of the delta matrix representation, suggested by the fast version of the Schwab–Knopoff algorithm. Factorizations, similar to those employed in the fast Schwab–Knopoff method, partly give the fast delta matrix method its computational efficiency. It transpires that the fast delta matrix algorithm is slightly more efficient than the fast Schwab–Knopoff scheme, and also that it has a much simpler algebraic representation. We therefore propose that the fast delta matrix algorithm is the best one to use in the computation of surface-wave dispersion for perfectly elastic, isotropic, plane-layered models. In addition, it incorporates all the computational

advantages of the reduced delta matrix scheme and is readily scaled for overflow and underflow problems.

We have explained a misunderstanding given in the paper by Abo-Zena (1979) and developed his recursion in a simpler manner. When the recursion is applied to the 'fast' transformation of the propagator matrix, we obtain, rather remarkably, the same algorithm as the fast delta matrix algorithm with the factorizations *automatically* performed. In this regard, the Abo-Zena recursion applied to the fast transformed propagator is the most convenient theoretical tool for deriving the fast delta matrix algorithm.

We have not considered here dispersion computations for viscoelastic and anisotropic media. These present different problems of efficiency and stability. However, in these cases, it seems likely that Kennett's RT matrix method, although inefficient for perfectly elastic and isotropic media, will be a very appropriate algorithm.

REFERENCES

Abo-Zena, A., 1979. Dispersion function computations for unlimited frequency values, *Geophys. J. R. astr. Soc.*, **58,** 91–105.

Dunkin, J.W., 1965. Computation of modal solutions in layered, elastic media at high frequencies, *Bull. seism. Soc. Am.*, **55,** 335–358.

Gilbert, F. & Backus, G.E., 1966. Propagator matrices in elastic wave and vibration theory, *Geophysics,* **31,** 326–332.

Haskell, N.A., 1953. The dispersion of surface waves on multilayered media, *Bull. seism. Soc. Am.,* **43,** 17–34.

Keilis-Brook, V.I. (ed.), 1989. Seismic surface waves in a laterally inhomogeneous Earth, in *Modern Approaches in Geophysics,* **9,** Kluwer, Dordrecht.

Kennett, B.L.N., 1974. Reflection, rays and reverberations, *Bull. seism. Soc. Am.,* **64,** 1685–1696.

Kennett, B.L.N. & Kerry, N.J., 1979. Seismic waves in a stratified halfspace, *Geophys. J. R. astr. Soc.,* **57,** 557–583.

Kennett, B.L.N., Kerry, N.J. & Woodhouse, J.H., 1978. Symmetries in reflection and transmission of elastic waves, *Geophys. J. R. astr. Soc.,* **52,** 215–229.

Kerry, N.J., 1981. Synthesis of seismic surface waves, *Geophys. J. R. astr. Soc.,* **64,** 425–446.

Knopoff, L., 1964. A matrix method for elastic wave problems, *Bull. seism. Soc. Am.,* **54,** 431–438.

Malischewsky, P., 1987. Surface waves and discontinuities, in *Developments in Solid Earth Geophysics,* **16,** Elsevier, Amsterdam.

Panza, G.F., 1985. Synthetic seismograms: the Rayleigh waves modal summation, *J. geophys. Res.,* **58,** 125–145.

Pestel, E. & Leckie, F.A., 1963. *Matrix Methods in Elasto-Mechanics,* McGraw-Hill, New York, NY.

Randall, M.J., 1967. Fast programs for layered half-space problems, *Bull. seism. Soc. Am.,* **57,** 1299–1316.

Schwab, F., 1970. Surface-wave dispersion computations: Knopoff's method, *Bull. seism. Soc. Am.,* **60,** 1491–1520.

Schwab, F. & Knopoff, L., 1970. Surface-wave dispersion computations, *Bull. seism. Soc. Am.,* **60,** 321–344.

Schwab, F. & Knopoff, L., 1972. Fast surface wave and free mode computations, in *Methods in Computational Physics,* vol II, pp. 87–180, ed. Bolt, B.A., Academic Press, New York, NY.

Schwab, F., Nakanishi, K., Cuscito, M., Panza, G.F., Liang, G. & Frez, J., 1984. Surface-wave computations and the synthesis of theoretical seismograms at high frequencies, *Bull. seism. Soc. Am.,* **74,** 1555–1578.

Thomson, W.T., 1950. Transmission of elastic waves through a stratified solid medium, *J. appl. Phys.,* **21,** 89–93.

Thrower, E.N., 1965. The computation of the dispersion of elastic waves in layered media, *J. Sound Vib.,* **2,** 210–226.

Watson, T.H., 1970. A note on fast computation of Rayleigh wave dispersion in the multilayered half-space, *Bull. seism. Soc. Am.,* **60,** 161–166.

Woodhouse, J.H., 1974. Surface waves in a laterally varying layered structure, *Geophys. J. R. astr. Soc.,* **37,** 461–490.

APPENDIX A: MATHEMATICAL DETAILS

Appendicies A1–A4 list the elements of the propagator matrices for Love waves (A1) and Rayleigh waves (A2–A4). For ease of exposition, the layer index-i is generally suppressed on the layer quantitites $\gamma_i, \mu_i, r_i, s_i, t_i, S_{\alpha_i}, C_{\alpha_i}, S_{\beta_i}$ and C_{β_i}.

Note that the propagator elements are always real. They are also finite, since when

$$c \to \alpha_i, \quad \frac{1}{r_i} S_{\alpha_i} \to k d_i,$$

and when

$$c \to \beta_i, \quad \frac{1}{s_i} S_{\beta_i} \to k d_i.$$

The listed boundary matrices U', V are those for a top free-surface and a bottom half-space respectively.

A1 Love-wave propagator matrix

$$T_{11} = C_\beta \qquad T_{12} = \frac{1}{\mu s} S_\beta$$

$$T_{21} = \mu s S_\beta \qquad T_{22} = C_\beta$$

$$U' = [0, 1] \qquad V = \begin{bmatrix} 1 \\ \mu_\ell s_\ell \end{bmatrix}$$

A2 Rayleigh-wave propagator matrix

$$T_{11} = \gamma(2C_\alpha - tC_\beta)$$

$$T_{12} = \gamma\left(-\frac{t}{r} S_\alpha + 2s S_\beta\right)$$

$$T_{13} = \frac{\gamma}{\mu}\left(\frac{1}{r} S_\alpha - s S_\beta\right)$$

$$T_{14} = -\frac{\gamma}{\mu}(C_\alpha - C_\beta)$$

$$T_{21} = \gamma\left(2r S_\alpha - \frac{t}{s} S_\beta\right)$$

$$T_{22} = \gamma(-tC_\alpha + 2C_\beta)$$

$$T_{23} = -T_{14}$$

$$T_{24} = \frac{\gamma}{\mu}\left(-r S_\alpha + \frac{1}{s} S_\beta\right)$$

$$T_{31} = \mu\gamma\left(4r S_\alpha - \frac{t^2}{s} S_\beta\right)$$

$$T_{32} = -2t\mu\gamma(C_\alpha - C_\beta)$$

$$T_{33} = T_{11}$$

$$T_{34} = -T_{21}$$

$$T_{41} = -T_{32}$$

$$T_{42} = \mu\gamma\left(-\frac{t^2}{r}S_\alpha + 4sS_\beta\right)$$

$$T_{43} = -T_{12}$$

$$T_{44} = T_{22}$$

$$U' = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad V = \begin{bmatrix} 1 & s \\ r & 1 \\ 2\mu r & \mu t \\ \mu t & 2\mu s \end{bmatrix}_\ell$$

A3 R-wave standard delta matrix

Define

$$Q_m = \left(\frac{t_i^m}{r_i s_i} + 2^m r_i s_i\right) S_{\alpha_i} S_{\beta_i} \quad \text{for} \quad 0 \le m \le 4.$$

Then

$$\bar{T}_{11} = \gamma^2[-4t + (t^2+4)C_\alpha C_\beta - Q_2]$$

$$\bar{T}_{12} = \frac{\gamma^2}{\mu}[(2+t)(1 - C_\alpha C_\beta) + Q_1]$$

$$\bar{T}_{13} = \frac{\gamma}{\mu}\left[\frac{1}{s}C_\alpha S_\beta - rS_\alpha C_\beta\right]$$

$$\bar{T}_{14} = \frac{\gamma}{\mu}\left[sC_\alpha S_\beta - \frac{1}{r}S_\alpha C_\beta\right]$$

$$\bar{T}_{15} = -\bar{T}_{12}$$

$$\bar{T}_{16} = \frac{\gamma^2}{\mu^2}[2(1 - C_\alpha C_\beta) + Q_0]$$

$$\bar{T}_{21} = \mu\gamma^2[-2t(t+2)(1 - C_\alpha C_\beta) - Q_3]$$

$$\bar{T}_{22} = 1 + C_\alpha C_\beta - \bar{T}_{11}$$

$$\bar{T}_{23} = \gamma\left[\frac{t}{s}C_\alpha S_\beta - 2rS_\alpha C_\beta\right]$$

$$\bar{T}_{24} = \gamma\left[2sC_\alpha S_\beta - \frac{t}{r}S_\alpha C_\beta\right]$$

$$\bar{T}_{25} = 1 - \bar{T}_{22}$$

$$\bar{T}_{26} = \bar{T}_{12}$$

$$\bar{T}_{31} = \mu\gamma\left[4sC_\alpha S_\beta - \frac{t^2}{r}S_\alpha C_\beta\right]$$

$$\bar{T}_{32} = -\bar{T}_{24}$$

$$\bar{T}_{33} = C_\alpha C_\beta$$

$$\bar{T}_{34} = -\frac{s}{r}S_\alpha S_\beta$$

$$\bar{T}_{35} = \bar{T}_{24}$$

$$\bar{T}_{36} = -\bar{T}_{14}$$

$$\bar{T}_{41} = \gamma\mu\left[\frac{t^2}{s}C_\alpha S_\beta - 4rS_\alpha C_\beta\right]$$

$$\bar{T}_{42} = -\bar{T}_{23}$$

$$\bar{T}_{43} = -\frac{r}{s}S_\alpha S_\beta$$

$$\bar{T}_{44} = \bar{T}_{33}$$

$$\bar{T}_{45} = \bar{T}_{23}$$

$$\bar{T}_{46} = -\bar{T}_{13}$$

$$\bar{T}_{51} = -\bar{T}_{21}$$

$$\bar{T}_{52} = \bar{T}_{25}$$

$$\bar{T}_{53} = -\bar{T}_{23}$$

$$\bar{T}_{54} = -\bar{T}_{24}$$

$$\bar{T}_{55} = \bar{T}_{22}$$

$$\bar{T}_{56} = -\bar{T}_{12}$$

$$\bar{T}_{61} = \mu^2\gamma^2[8t^2(1 - C_\alpha C_\beta) + Q_4]$$

$$\bar{T}_{62} = \bar{T}_{21}$$

$$\bar{T}_{63} = -\bar{T}_{41}$$

$$\bar{T}_{64} = -\bar{T}_{31}$$

$$\bar{T}_{65} = -\bar{T}_{21}$$

$$\bar{T}_{66} = \bar{T}_{11}$$

$$\bar{U}' = [0, 0, 0, 0, 0, 1] \qquad \bar{V} = \begin{bmatrix} 1 - rs \\ \mu(t - 2rs) \\ \mu s(2 - t) \\ -\mu r(2 - t) \\ -\mu(t - 2rs) \\ \mu^2(4rs - t^2) \end{bmatrix}_\ell$$

A4 Fast delta matrix elements

Define

$$\varepsilon_i = \rho_{i+1}/\rho_i; \qquad \eta_i = 2(\gamma_i - \varepsilon_i\gamma_{i+1})$$

and

$$a_i = \varepsilon_i + \eta_i; \qquad a_i' = a_i - 1; \qquad b_i = 1 - \eta_i; \qquad b_i' = b_i - 1.$$

We also use the identities $a_i' + b_i = a_i + b_i' = a_i b_i - a_i' b_i' = \varepsilon_i$.

$$\bar{T}_{11} = ab$$

$$\bar{T}_{12} = aa'$$

$$\bar{T}_{13} = 0$$

$$\bar{T}_{14} = 0$$

$$\bar{T}_{15} = bb'$$

$$\bar{T}_{16} = a'b'$$

$$\bar{T}_{21} = ab' C_\alpha C_\beta - a' b(S_\alpha/r)(S_\beta/s)$$
$$\bar{T}_{22} = a^2 C_\alpha C_\beta - a'^2 (S_\alpha/r)(S_\beta/s)$$
$$\bar{T}_{23} = \varepsilon C_\alpha (S_\beta/s)$$
$$\bar{T}_{24} = -\varepsilon (S_\alpha/r) C_\beta$$
$$\bar{T}_{25} = b'^2 C_\alpha C_\beta - b^2 (S_\alpha/r)(S_\beta/s)$$
$$\bar{T}_{26} = \bar{T}_{21}$$

$$\bar{T}_{31} = ab' C_\alpha (sS_\beta) - a' b(S_\alpha/r) C_\beta$$
$$\bar{T}_{32} = a^2 C_\alpha (sS_\beta) - a'^2 (S_\alpha/r) C_\beta$$
$$\bar{T}_{33} = \varepsilon C_\alpha C_\beta$$
$$\bar{T}_{34} = -\varepsilon (S_\alpha/r)(sS_\beta)$$
$$\bar{T}_{35} = b'^2 C_\alpha (sS_\beta) - b^2 (S_\alpha/r) C_\beta$$
$$\bar{T}_{36} = \bar{T}_{31}$$

$$\bar{T}_{41} = -ab' (rS_\alpha) C_\beta + a' b C_\alpha (S_\beta/s)$$
$$\bar{T}_{42} = -a^2 (rS_\alpha) C_\beta + a'^2 C_\alpha (S_\beta/s)$$
$$\bar{T}_{43} = -\varepsilon (rS_\alpha)(S_\beta/s)$$
$$\bar{T}_{44} = \varepsilon C_\alpha C_\beta$$
$$\bar{T}_{45} = -b'^2 (rS_\alpha) C_\beta + b^2 C_\alpha (S_\beta/s)$$
$$\bar{T}_{46} = \bar{T}_{41}$$

$$\bar{T}_{51} = -ab' (rS_\alpha)(sS_\beta) + a' b C_\alpha C_\beta$$
$$\bar{T}_{52} = -a^2 (rS_\alpha)(sS_\beta) + a'^2 C_\alpha C_\beta$$
$$\bar{T}_{53} = -\varepsilon (rS_\alpha) C_\beta$$
$$\bar{T}_{54} = \varepsilon C_\alpha (sS_\beta)$$
$$\bar{T}_{55} = -b'^2 (rS_\alpha)(sS_\beta) + b^2 C_\alpha C_\beta$$
$$\bar{T}_{56} = \bar{T}_{51}$$

$$\bar{T}_{61} = a' b'$$
$$\bar{T}_{62} = aa'$$
$$\bar{T}_{63} = 0$$
$$\bar{T}_{64} = 0$$
$$\bar{T}_{65} = bb'$$
$$\bar{T}_{66} = ab$$

$$\bar{U}' = \mu_1^2 [2t_1, -t_1^2, 0, 0, -4, 2t_1] \qquad \bar{V} = \begin{bmatrix} 0 \\ 1 \\ s_\ell \\ -r_\ell \\ -r_\ell s_\ell \\ 0 \end{bmatrix}$$

A5 Fast delta matrix algorithm

This algorithm is similar to the fast Schwab–Knopoff algorithm, except that the determinant is computed using the delta matrix formalism rather than the Knopoff decomposition. The resulting algorithm is algebraically simpler, and is some 12 per cent more efficient.

The algorithm is expressed in terms of a single row vector X of six components. Only five are actually needed, and these correspond to the 'reduced' version of the algorithm.

Factorizations optimized for numerical computation are expressed in terms of the parameters p_a, q_a, y_b, z_b ($a = 1, 2, 3, 4$; $b = 1, 2$). Note that there are no resultant factors containing terms in C_α^2, S_α^2, C_β^2 or S_β^2, which would give rise to the instability problem.

Let

$$X_i = [x_1, x_2, \ldots, x_6]; \qquad X_{i+1} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_6].$$

In every iteration, $x_6 = x_1$ (and $\hat{x}_6 = \hat{x}_1$), so dropping these terms gives the reduced algorithm. We include them here only for the sake of completeness.

(1) Initialize:

$$X_1 = \mu_1^2 [2t_1, -t_1^2, 0, 0, -4, 2t_1] \quad (t_1 = 2 - c^2/\beta_1^2).$$

(2) Layer Recursion: (repeated for $i = 1, 2, \ldots, n$)

$$p_1 = C_\beta x_2 + sS_\beta x_3 \qquad q_1 = C_\alpha p_1 - rS_\alpha p_2$$
$$p_2 = C_\beta x_4 + sS_\beta x_5 \qquad q_2 = -\frac{1}{r} S_\alpha p_3 + C_\alpha p_4$$
$$p_3 = \frac{1}{s} S_\beta x_2 + C_\beta x_3 \qquad q_3 = C_\alpha p_3 - rS_\alpha p_4$$
$$p_4 = \frac{1}{s} S_\beta x_4 + C_\beta x_5 \qquad q_4 = -\frac{1}{r} S_\alpha p_1 + C_\alpha p_2$$

$$y_1 = a' x_1 + a q_1$$
$$y_2 = a x_1 + a' q_2$$
$$z_1 = b x_1 + b' q_1$$
$$z_2 = b' x_1 + b q_2$$

$$\hat{x}_1 = b' y_1 + b y_2$$
$$\hat{x}_2 = a y_1 + a' y_2$$
$$\hat{x}_3 = \varepsilon q_3$$
$$\hat{x}_4 = \varepsilon q_4$$
$$\hat{x}_5 = b' z_1 + b z_2$$
$$\hat{x}_6 = a z_1 + a' z_2 = \hat{x}_1$$

(3) Dispersion function: $D(c, k) = \hat{x}_2 + s_\ell \hat{x}_3 - r_\ell (\hat{x}_4 + s_\ell \hat{x}_5)$, where the $\hat{x}$s are the final values obtained from the previous recursion.

A6 Abo-Zena algorithm

In Section 7 we developed a modified Abo-Zena recursion

$$X_{i+1} = A' X_i A + B' X_i B + A' X_i B - (A' X_i B)'.$$

Here we identify the following matrices:

$$A = \gamma_i \begin{bmatrix} 2C_\alpha & -tS_\alpha/r & S_\alpha/(r\mu) & -C_\alpha/\mu \\ 2rS_\alpha & -tC_\alpha & C_\alpha/\mu & -rS_\alpha/\mu \\ 4\mu rS_\alpha & -2t\mu C_\alpha & 2C_\alpha & -2rS_\alpha \\ 2t\mu C_\alpha & -\mu t^2 S_\alpha/r & tS_\alpha/r & -tC_\alpha \end{bmatrix}_i$$

$$B = \gamma_i \begin{bmatrix} -tC_\beta & 2sS_\beta & -sS_\beta/\mu & C_\beta/\mu \\ -tS_\beta/s & 2C_\beta & -C_\beta/\mu & S_\beta/(s\mu) \\ -\mu t^2 S_\beta/s & 2\mu tC_\beta & -tC_\beta & tS_\beta/s \\ -2\mu tC_\beta & 4\mu sS_\beta & -2sS_\beta & 2C_\beta \end{bmatrix}_i$$

$$\begin{aligned} A'X_iA = \gamma_i^2\, \mathrm{asym}[&-2tx_1 - 4\mu t x_2 + 2\mu t^2 x_5 + 4\mu^2 t^2 x_6, \\ &2/\mu x_1 + 4x_2 - 2tx_5 - 4\mu t x_6, \\ &0, \quad 0, \\ &-t/\mu x_1 - 2tx_2 + t^2 x_5 + 2\mu t^2 x_6, \\ &1/\mu^2 x_1 + 2/\mu x_2 - t/\mu x_5 - 2tx_6]_i \end{aligned}$$

$$\begin{aligned} B'X_iB = \gamma_i^2\, \mathrm{asym}[&-2tx_1 - 2\mu t^2 x_2 + 4\mu t x_5 + 4\mu^2 t^2 x_6, \\ &t/\mu x_1 + t^2 x_2 - 2tx_5 - 2\mu t^2 x_6, \\ &0, \quad 0, \\ &-2/\mu x_1 - 2tx_2 + 4x_5 + 4\mu t x_6, \\ &1/\mu^2 x_1 + t/\mu x_2 - 2/\mu x_5 - 2tx_6]_i. \end{aligned}$$

Note that $A'X_iA$ and $B'X_iB$ are independent of the eigenfunctions $C_{\alpha_i}, S_{\alpha_i}; C_{\beta_i}, S_{\beta_i}$. They also happen to be independent of x_3 and x_4. A close examination shows that the coefficients of $A'X_iA + B'X_iB$ are precisely the constant terms (those independent of C_α and C_β) in the propagator delta matrix elements $\bar{T}_{ij}$ listed in A3.

Let (see eq. 50)

$$X_i = \mathrm{asym}[x_1, x_2, \ldots, x_6]; \qquad X_{i+1} = \mathrm{asym}[\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_6].$$

Then the Abo-Zena algorithm can be written in the following form.

(1) Initialize: $X_1 = \mathrm{asym}[0, 0, 0, 0, 0, 1]$.

(2) Layer Recursion: (for $i = 1, 2, \ldots, n$)

$$g_1 = -4t_i \qquad g_4 = (2 + t_i)/\mu_i$$

$$g_2 = -2\mu_i t_i (2 + t_i) \qquad g_5 = 4 + t_i^2$$

$$g_3 = 8\mu_i^2 t_i^2 \qquad g_6 = 2/\mu_i^2$$

$$\hat{x}_1 = g_1 x_1 + g_2 x_2 - g_2 x_5 + g_3 x_6 + (y_{12} - y_{21})$$

$$\hat{x}_2 = g_4 x_1 + g_5 x_2 + g_1 x_5 + g_2 x_6 + (y_{13} - y_{31})$$

$$\hat{x}_3 = (y_{14} - y_{41})$$

$$\hat{x}_4 = (y_{23} - y_{32})$$

$$\hat{x}_5 = -g_4 x_1 + g_1 x_2 + g_5 x_5 - g_2 x_6 + (y_{24} - y_{42})$$

$$\hat{x}_6 = g_6 x_1 + g_4 x_2 - g_4 x_5 + g_1 x_6 + (y_{34} - y_{43})$$

where y_{ab} are the elements of the matrix product $Y = A'X_iB$.

(3) Dispersion function: $D = v_1' X_\ell v_2$, where X_ℓ is the result of the last iteration in the previous recursion, and

$$v_1 = [1, r, 2\mu r, \mu t]_\ell'; \qquad v_2 = [s, 1, \mu t, 2\mu s]_\ell'.$$

A7 Kennett's RT method

Here we list the partitions of the 2×2 unphased interface matrix $\bar{S}_i = Q_i^{-1} Q_{i+1}$ described in Section 8. The subscript i is suppressed and a dashed symbol will denote a quantity with subscript $(i+1)$. The up and down partitions of $\bar{S}_i$ will appear as in eq. (64) by applying the permutation (3142) to the columns of Q_i.

Define $m = \mu_{i+1}/\mu_i = \mu'/\mu$. Then

$$\bar{S}_{\mathrm{uu}} = \frac{1}{2}\gamma \begin{bmatrix} 2 - tr'/r + m(2r'/r - t') & -2s' + t/r + m(2s' - t'/r) \\ -2r' + t/s + m(2r' - t'/s) & 2 - ts'/s + m(2s'/s - t') \end{bmatrix},$$

$$\bar{S}_{\mathrm{ud}} = \frac{1}{2}\gamma \begin{bmatrix} 2 + tr'/r - m(2r'/r + t') & 2s' + t/r - m(2s' + t'/r) \\ 2r' + t/s - m(2r' + t'/s) & 2 + ts'/s - m(2s'/s + t') \end{bmatrix}.$$

$\bar{S}_{\mathrm{du}} = \bar{S}^*_{\mathrm{ud}}$ and $\bar{S}_{\mathrm{dd}} = \bar{S}^*_{\mathrm{uu}}$ where the * indicates that the signs of the off-diagonal terms are reversed.

A8 Laplace's method

This is a general method of calculating the determinant of a square matrix A of any order. It works by summing over all minors and complementary minors associated with a given set of rows (or columns) of A.

Let M denote the set of minors of A with any fixed subset of rows (columns) of A. Let n be the dimension of A and p the number of rows (columns) selected. Then there are ${}^nC_p$ minors M, each of order p.

Let M^c represent the unique minor complementary to M; that is, M^c is the minor constructed from all rows and columns of A not contained in M. The order of M^c is therefore $(n - p)$. It is assumed that all rows and columns of M and M^c have the same relative rank as contained in the original matrix A.

Laplace's Theorem states that

$$\det|A| = \sum (-1)^s M M^c.$$

The sum is taken over all ${}^nC_p$ minors M, and s (which determines the sign of each term in the sum) is equal to the sum of all row *and* column indicies of A that are contained in M.

Example

Let A be a 4×4 matrix with elements a_{ij} for $i, j = 1, 2, 3, 4$. Choose M to be the set of minors associated with the first row of A. There are four such minors, all of order 1, and equal to

$$M_1 = a_{11}, \qquad M_2 = a_{12}, \qquad M_3 = a_{13}, \qquad M_4 = a_{14}.$$

Let $A_{\bar{i}\bar{j}}$ denote the order-3 minor obtained from A by deleting row-i and column-j. Then Laplace's formula reduces to the well-known formula

$$\det|A| = a_{11}A_{\bar{1}\bar{1}} - a_{12}A_{\bar{1}\bar{2}} + a_{13}A_{\bar{1}\bar{3}} - a_{14}A_{\bar{1}\bar{4}}.$$

The signs are obtained from $s_1 = 1 + 1$; $s_2 = 1 + 2$; $s_3 = 1 + 3$; $s_4 = 1 + 4$.