45-DEGREE WAVE EQUATION MIGRATION

Allan Jacobs

The 45-degree equation is normally employed in an implicit scheme which has not been published in a SEP report before. A review of the theoretical background, stability properties, dispersion relations, inaccuracies, and programming quirks of the method is in order. After these topics are explored, a sample program will be appended.

1. Theory and Numerical Implementation

The derivation of the 45-degree approximation starts with the wave equation.

$$p_{xx} + p_{zz} = \frac{1}{v^2} p_{tt}$$

A coordinate transformation is applied, introducing retarded time coordinates for upcoming waves.

$$x' = x$$
$$t' = t+z/v_o$$
$$z' = z$$
$$p'(x',t',z') = p(x,t,z)$$

Using the chain rule, we get the wave equation in the new coordinate system. Dropping primes, this is

$$p_{xx} + p_{zz} + (2/v_o)p_{tz} + [v_o^{-2} - v^{-2}]p_{tt} = 0$$

If we assume a homogeneous earth and a proper pick for v_o, this becomes

$$p_{xx} + p_{zz} + (2/v)p_{tz} = 0 \tag{1}$$

Of these terms, p_{zz} is the smallest. We drop it temporarily to get

$$p_{xx} + (2/v)p_{tz} = 0$$

and differentiate by z

$$p_{zzt} = -(v/2)p_{xxz} \tag{2}$$

Next we differentiate equation (1) by t giving

$$p_{xxt} + p_{zzt} + (2/v)p_{ztt} = 0 \tag{3}$$

and substitute (2) into (3)

$$p_{xxt} - (v/2)p_{xxz} + (2/v)p_{ttz} = 0$$

which we recast in the form

$$(v/2)p_{xxt} - (v/4)p_{xxz} + p_{ttz} = 0 \tag{4}$$

Equation (4) is relation which we will discretize and implement on the computer. In the process, a computation star with three time levels and two z-levels will be constructed. We first define a set of variables and indices

$$z_j = j\Delta z$$

$$t_k = k\Delta t$$

$$a = \frac{v\Delta t\Delta z}{8(\Delta x)^2}$$

$$b = \left(\frac{v\Delta t}{4\Delta x}\right)^2 \tag{5}$$

Next we let T denote a tridiagonal matrix with $\frac{1}{(\Delta x)^2}(-1,2,-1)$ along its diagonal. This matrix is approximately equivalent of the negative of the second derivative over x. Finally, we let p_j^k denote a vector defined along the x-axis at z-level j and t-level k.

Equation (4) now takes the form

$$0 = -aT(P_{j+1}^{k+1}+P_j^{k+1}-P_{j+1}^{k-1}-P_j^{k-1}) + bT(P_{j+1}^{k+1}-P_j^{k+1}+2P_{j+1}^k-2P_j^k+P_{j+1}^{k-1}-P_j^{k-1})$$

$$+(P_{j+1}^{k+1}-2P_{j+1}^k+P_{j+1}^{k-1}-P_j^{k+1}+2P_j^k-P_j^{k-1}) \qquad (6)$$

which we diagram. Let $\gamma=1/6$ improving the approximation to ∂_{xx}.

-1	1		1	1		-1	1		1	-1	
2	-2	+ aT	0	0	+ bT	-2	2	+ γT	-2	2	= 0
-1	1		-1	-1		-1	1		1	-1	

2. Stability

A rather tedious proof will now establish that (6) is unconditionally stable. We rewrite (6) as

$$[I+aT+bT]P_{j+1}^{k-1} + [-2I+2bT]P_{j+1}^k + [I-aT+bT]P_{j+1}^{k+1}$$

$$= [I-aT+bT]P_j^{k-1} + [-2I-2bT]P_j^k + [I+aT+bT]P_j^{k+1} \qquad (7)$$

so as to separate the $j+1$ z-level from the j z-level. Using the notation in Clarebout's book FGDP we denote the delay operator by Z and take the Z transform over the time index k. At this time, we also replace I by 1 and the matrix T by a scalar T, as described in FGDP. This is permissible because the eigenvalues of the matrix T are real numbers lying between 0 and 4. The Z-transform is

$$[(1+aT+bT)Z^2 + (-2+2bT)Z + (1-aT+bT)]Q_{j+1}(Z)$$

$$= [(1-aT+bT)Z^2 + (-2+2bT)Z + (1+aT+bT)]Q_j(Z)$$

The input is $Q_{j+1}(Z)$ for upcoming waves so we write the transfer function as

$$H(Z) = \frac{(1+aT+bT)Z^2 + (-2+2bT)Z + (1-aT+bT)}{(1+aT+bT) + (-2+2bT)Z + (1-aT+bT)Z^2} \tag{8}$$

Equation (8) corresponds to an all-pass filter so the z-dependence of the migration process is stable. Time recurrence stability will be investigated by an analysis of the roots of the denominator of (B).

The roots occur when

$$(1-aT+bT)Z^2 + (-2+2bT)Z + (1+aT+bT) = 0$$

now we introduce new variables $A=aT$, $B=bT$. The poles of the transfer function are now found to lie in the complex Z-plane at

$$\frac{1-B}{1-A+B} \pm \sqrt{\left(\frac{-1+B}{1-A+B}\right)^2 - \frac{1+A+B}{1-A+B}} \tag{9}$$

From (9), the condition that the roots be real is that

$$B < (1/4)A^2 \tag{10}$$

A. Case of real roots, both > 1.

This happens when (10) is satisfied and when the smaller of the two roots is greater than unity. This last occurs when

$$\frac{1-B}{1-A+B} - \sqrt{\left(\frac{1-B}{1-A+B}\right)^2 - \frac{1+A+B}{1-A+B}} > 1$$

which we rearrange to give

$$\sqrt{\left(\frac{1-B}{1-A+B}\right)^2 - \frac{1+A+B}{1-A+B}} < \frac{A-2B}{1-A+B} \tag{11}$$

In order for this to happen, the right hand side of (11) must be positive. In other words, ($A>2B$ and $B-A>-1$) or ($A<2B$ and $B-A<-1$). In addition to one of these, the inequality (11) must also hold. Since both sides are positive we are justified in squaring both sides and do enough algebra to find that it is necessary that $B-A>-1$. Thus the condition for real roots, both greater than 1 is that (10) hold, and that

$$A>2B \quad \text{and} \quad B-A>-1 \tag{12}$$

B. Case of real roots, both <-1.

This happens when (10) is satisfied and when the larger of the two roots is less than -1.

$$\frac{1-B}{1-A+B}+\sqrt{\left(\frac{1-B}{1-A+B}\right)^2-\frac{1+A+B}{1-A+B}}<-1$$

which we rearrange to give

$$\sqrt{\left(\frac{1-B}{1-A+B}\right)^2-\frac{1+A+B}{1-A+B}}<\frac{A-2}{1-A+B} \tag{13}$$

For (13) to hold, its right hand side must be positive, giving the conditions ($A>2$ and $B-A>-1$) or ($A<2$ and $B-A<-1$). Given this, we can square both sides of (13) and get $B-A>-1$ after some algebra. Thus the condition for real roots, both less than -1, is that (10) hold and that

$$A>2 \quad \text{and} \quad B-A>-1 \tag{14}$$

C. Case of real roots, one >1 and the other <-1.

This is the hardest case. We start with the inequalities

$$\frac{1-B}{1-A+B}+\sqrt{\left(\frac{1-B}{1-A+B}\right)^2-\frac{1+A+B}{1-A+B}}>1 \tag{15}$$

$$\frac{1-B}{1-A+B} - \sqrt{\left(\frac{1-B}{1-A+B}\right)^2 - \frac{1+A+B}{1-A+B}} < -1 \qquad (16)$$

We take up inequality (15) first, writing it as

$$\sqrt{\left(\frac{1-B}{1-A+B}\right)^2 - \frac{1+A+B}{1-A+B}} > \frac{2B-A}{1-A+B} \qquad (17)$$

Inequality (17) is automatically true if its right hand side is negative; if $(A > 2B$ and $B-A > -1)$ or $(A < 2B$ and $B-A < -1)$.

Otherwise, we need to square both sides and simplify. The condition we then obtain is that $(A < 2B$ and $B-A > -1)$ or $(A > 2B$ and $B-A < -1)$ to guarantee that the right hand side is positive and that $B-A < -1$, so the overall condition for this special case is that $(A > 2B$ and $B-A < -1)$.

Inequality (17) and therefore inequality (15) holds when

$$\begin{aligned}&(A > 2B \text{ and } B-A > -1) \text{ or } (A < 2B \text{ and } B-A < -1) \text{ or}\\ &(A > 2B \text{ and } B-A < -1)\end{aligned} \qquad (18)$$

Inequality (16) also has to be satisfied so we rewrite it as

$$\sqrt{\left(\frac{1-B}{1-A+B}\right)^2 - \frac{1+A+B}{1-A+B}} > \frac{2-A}{1-A+B} \qquad (19)$$

This inequality is automatically true when its right hand side is negative, when $(A > 2$ and $B-A > -1)$ or $(A < 2$ and $B-A < -1)$. Otherwise we have to again square both sides of an inequality, this time inequality (19), and simplify. This process leads to the inequalities $(A > 2$ and $B-A < -1)$. Therefore, inequality (19), and therefore (16), is true when

$$\begin{aligned}&(A > 2 \text{ and } B-A > -1) \text{ or } (A < 2 \text{ and } B-A < -1) \text{ or}\\ &(A > 2 \text{ and } B-A < -1)\end{aligned} \qquad (20)$$

Both (15) and (16) are true when both (18) and (20) are simultaneously true. Thus the condition for real roots, one greater than unity and the other less than -1, is that (10) be satisfied, and that

$$(A < 2B \text{ and } B-A < -1) \text{ or } (A > 2B \text{ and } B-A < -1) \qquad (21)$$

D. Case of complex roots.

If the roots of (9) are complex then they must occur as a complex conjugate pair. The squared modulus of each of these roots is equal to the product of one with the other. We rewrite (9) as

$$Z = \frac{1-B}{1-A+B} \pm i \sqrt{\frac{1+A+B}{1-A+B} - \left(\frac{1-B}{1-A+B}\right)^2}$$

The modulus squared is

$$|Z|^2 = \frac{1+A+B}{1-A+B}$$

which is always greater than 1.

Finally, we note that the condition for complex roots is that $B > (1/4)A^2$.

E. Wrapup

The above proof has succeeded in partitioning the A-B plane into disjoint regions which completely cover that plane's first quadrant. Since $A=aT > 0$ and $B=bT > 0$, this turns out to be the only part of the plane of interest. Since the cover is complete, the poles of the transfer function (8) must all lie outside of the unit circle.

The partition is sketched in Figure 1.

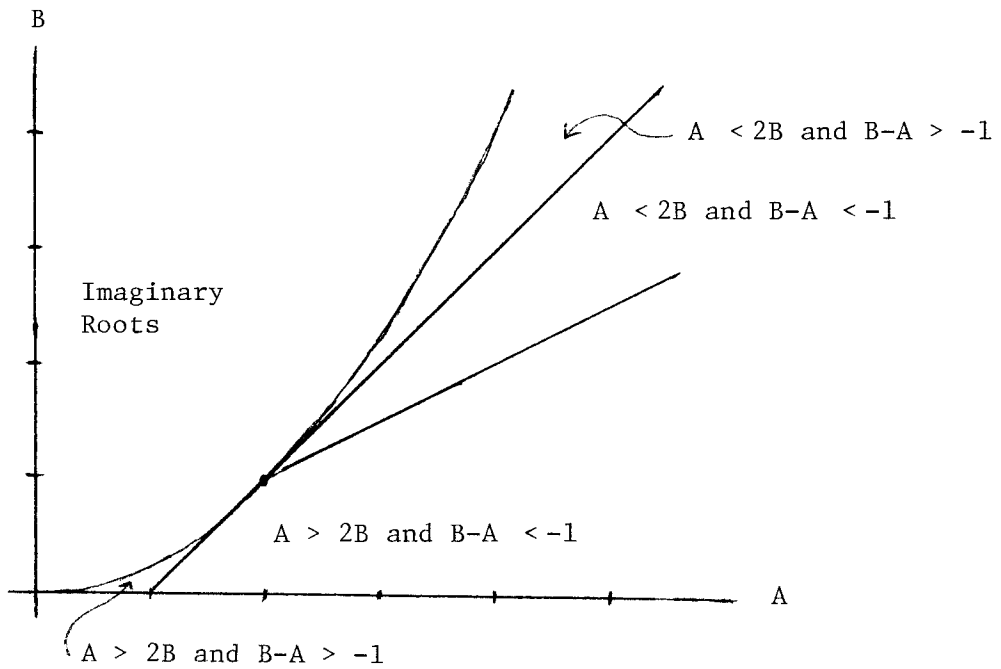


Figure 1

3. *Frequency Dispersion*

Many of the properties and problems associated with the use of the 45 degree equation can be conveniently studied in the Fourier transform domain. The best approximations are those which fit the dispersion relation of the exact wave equation as closely as possible over the widest possible domain. Errors associated with mismatching of the dispersion circle of the exact wave equation due to undersampling of the data and approximations to the derivatives in the wave equation are lumped under the term frequency dispersion.

We start with the 45 degree equation and derive a dispersion relation in the retarded time frame. The Fourier transform of the 45 degree equation is given by

$$ik_z\left(\frac{k^2v^2}{4} - \omega^2\right) + \frac{iv\omega k^2}{2} = 0 \tag{22}$$

where k is the transform variable associated with x, suffix deleted. We let

$$K = kv/\omega$$
$$K_z = k_z v/\omega$$

so that (1) can be rewritten as

$$K_z = \frac{0.5\ K^2}{1-0.25\ K^2} \tag{23}$$

The dispersion relation for the exact wave equation is, by similar means, found to be

$$k^2 + k_z^2 = (\omega/v)^2$$

or

$$K_z = -(1-K^2)^{1/2} \tag{24}$$

which in retarded time coordinates becomes

$$K_z = 1 - (1-K^2)^{1/2} \tag{25}$$

In practice, we have to sample the data in s and t, and are forced to use approximations to the partial derivatives in the wave equation. The approximation used in this paper is

$$\partial_{xx} = \delta_{xx}/(\Delta x)^2 \quad \frac{1}{1 + \gamma\ \delta_{xx}}$$

where ∂_{xx} is the discrete second difference operator on the x-axis $(1,-2,1)$, and γ is a constant of magnitude 1/6 or 1/7. The Fourier transform of this can be found by using a formula from FGDP

$$\text{F.T. } (\delta_{xx}) = -4 \sin^2 \frac{k\Delta x}{2}$$

so that

$$\hat{k}^2 = -\text{ F.T .}(\delta_{xx}) = \frac{4}{(\Delta x)^2} \quad \frac{\sin^2(k\Delta x/2)}{1+4\gamma \sin^2(k\Delta x/2)}$$

where $\hat{k}$ and $\hat{K}$ are approximations to k and K, respectively. Letting

$$m = \frac{\omega \Delta x}{2v}$$

and noting that

$$\frac{k\Delta x}{2} = \frac{vk}{\omega} \quad \frac{\omega \Delta x}{2v} = \frac{vkm}{2\omega} = Km$$

we set

$$\hat{K}^2 = m^{-2} \frac{\sin^2 Km}{1+4\gamma \sin^2 Km} \tag{26}$$

We insert this last relation into (22) and (25) wherever K appears. The results are approximations to the 45 degree equation and the exact wave equation, respectively, that take into account the effects of discrete sampling of the data.

This gives us three dispersion relations of practical value: those belonging to the exact wave equation, the sampled wave equation, and the sampled 45 degree equation.

$$\begin{aligned} K_z &= 1 - (1-K^2)^{1/2} \\ K_z &= 1 - \left(1-m^{-2} \frac{\sin^2 Km}{1+4\gamma \sin^2 Km}\right)^{1/2} \\ K_z &= \frac{0.5m^{-2} \sin^2 Km}{1+(4\gamma-0.25m^{-2}) \sin^2 Km} \end{aligned} \tag{27}$$

In (K,K_z) space, or dip-space, the sampled equations are periodic with period $m\pi$. As the sampling density along x decreases the approximation given by the sampled 45 degree equation to the dispersion relation of the exact wave equation deteriorates. In general the dispersion curve for the 45 degree equation lies outside the wave equation circle, the fit to that curve worsening as vk_x/ω increases. Near the points corresponding to dips of 90 degrees there may be aliasing as well, especially as m begins to exceed π in magnitude.

The effect of frequency dispersion is to incorrectly migrate dipping beds. The timing error associated with the use of the 45 degree equation for a eflecting bed with dip θ is given by

$$\Delta t_{dip} = (z/\omega)(\hat{k}_z - k_z) = (z/v)(\hat{K}-K) \tag{28}$$

A table of timing errors versus dip angles for three values of the parameter m follows.

Δt_{dip} v/2 vs. Dip in Degrees

m = 0.5		m = 1.0		m = 2.0	
0	0.0000	0	0.0000	0	0.0000
10	0.0000	10	0.0003	10	0.0015
20	-0.0006	20	0.0045	20	0.0190
30	-0.0031	30	0.0208	30	0.0695
40	-0.0095	40	0.0577	40	0.1575
50	-0.0226	50	0.1214	50	0.2808
60	-0.0455	60	0.2143	60	0.4355
70	-0.0810	70	0.3352	70	0.6166
80	-0.1320	80	0.4810	80	0.8127
90	-0.2000	90	0.6471	90	1.0000

If the 45 degree equation is used to make a depth section and proper location of events with dips with high angles is desired, it is evident that a high sampling density along the surface of the earth is necessary.

4. Anisotropy Dispersion

Anisotropy dispersion is the effect of group velocity errors in wave equation approximations. If we start with the wavefront from a point scatterer and attempt to collapse that wavefront to a point using some migration scheme, it follows that the failure to get a complete collapse is properly studied by looking at anisotropy dispersion.

In order to set the group velocity as a function of K it is convenient to define intermediate variables s_k and s_ω, defined by taking derivatives using the appropriate dispersion relation in an unshifted time frame.

$$s_k = \frac{\partial k_z}{\partial k} \qquad s_\omega = \frac{\partial k_z}{\partial \omega} \tag{29}$$

For the exact wave equation s_k is given by

$$s_k = -\frac{\omega}{v}\frac{\partial}{\partial k}(1-K^2)^{1/2} = \frac{\omega}{v}(1-K^2)^{-1/2}\,K\,\frac{v}{\omega}$$

which simplifies to

$$s_k = K(1-K^2)^{-1/2}$$

Correspondingly, s_ω is given by

$$s_\omega = -\frac{1}{v}(1-K^2)^{1/2} - \frac{\omega}{v}(1-K^2)^{-1/2}\,K\,(vk/\omega^2)$$

$$s_\omega = -\frac{1}{v}(1-K^2)^{1/2} - \frac{1}{v}K^2(1-K^2)^{-1/2}$$

$$s_\omega = -\frac{1}{v}(1-K^2)^{-1/2} \tag{30}$$

For the sampled wave equation, we start with the dispersion relation

$$K_z = -\left(1-m^{-2}\frac{\sin^2 Km}{1+4\gamma\sin^2 Km}\right)^{1/2}$$

and take derivatives. The variable s_k is found to be

$$s_k = \frac{\omega}{2v}\left(1-m^{-2}\frac{\sin^2 Km}{1+4\gamma\sin^2 Km}\right)^{-1/2} m^{-2}\frac{\partial}{\partial k}\,\frac{\sin^2 Km}{1+4\gamma\sin^2 Km}$$

$$s_k = \frac{m^{-2}\omega}{2v}\left(1-m^{-2}\frac{\sin^2 Km}{1+4\gamma\sin^2 Km}\right)^{-1/2}\frac{vm}{\omega}\,\frac{\sin^2 Km}{(1+4\gamma\sin^2 Km)^2}$$

$$s_k = \frac{1}{2m}\,1-m^{-2}\left(\frac{\sin^2 Km}{1+4\gamma\sin^2 Km}\right)^{-1/2}\frac{\sin^2 Km}{(1+4\gamma\sin^2 Km)^2} \tag{31}$$

Similarly, s_ω is

$$s_\omega = \frac{1}{v} K_z - \frac{\omega}{2v} \left(1-m^{-2} \frac{\sin^2 Km}{1+4\gamma \sin^2 Km}\right)^{-1/2} m^{-2} \frac{\sin (2Km)}{(1+4\gamma \sin^2 Km)^2} Km \frac{1}{\omega}$$

$$s_\omega = -\frac{\omega}{v^2} \left(1-m^{-2} \frac{\sin^2 Km}{1+4\gamma \sin^2 Km}\right)^{-1/2} -$$

$$\frac{m}{2v} K \left(1-m^{-2} \frac{\sin^2 Km}{1+4\gamma \sin^2 Km}\right)^{-1/2} (1+4\gamma \sin^2 Km)^{-2}$$

Finally, for the sampled 45 degree equation, s_k is found by considering the dispersion relation for the 45 degree equation in an unretarded time frame.

$$K_z = - \frac{1+(4\gamma-0.75\ m^{-2}) \sin^2 (Km)}{1+(4\gamma-0.25\ m^{-2}) \sin^2 Km}$$

The derivative s_k is

$$s_k = -\frac{1}{m} \frac{(4\gamma - 0.75m^{-2}) \sin 2Km}{1 + (4\gamma - 0.25m^{-2}) \sin^2 Km}$$

$$+ \frac{1}{m} \frac{1 + (4\gamma - 0.75m^{-2}) \sin^2 Km}{(1 + 4\gamma - 0.25m^{-2}) \sin^2 Km} (4\gamma - 0.25m^{-2}) \sin 2Km$$

$$s_k = \frac{1}{2m} \frac{\sin 2Km}{[1 + (4\gamma - 0.25m^{-2}) \sin^2 Km]^2} \qquad (33)$$

and the ω derivative s_ω is found to be

$$s_\omega = -\frac{1}{v} K_z - \frac{1}{2m} \frac{\sin 2\,Km}{[1+(4\gamma-0.25\ m^{-2})\ \sin^2 Km]^2} K \frac{m}{v}$$

$$s_\omega = -\frac{1}{v} \frac{1+(4\gamma-0.75\ m^{-2})\ \sin^2 Km}{1+(4\gamma-0.25\ m^{-2})\ \sin^2 Km} - \frac{K}{2v} \frac{\sin (2Km)}{[1+(4\gamma-0.25\ m^{-2})\ \sin^2 Km]^2} \quad (34)$$

Wavefronts, to reiterate, move with a velocity given by the group velocity of the medium (or computational grid) in which they propagate. This velocity ratio of s_ω to s_k . In general group velocity is a function of position and direction of propagation. It turns out that the anistropy of group velocity associated with the 45 degree approximation is important to an understanding of the errors associated with the use of the 45 degree equation.

The equation for the wavefront from a point scatterer is given by

$$(x,t) = vt\ (s_k\ ,-s_\omega) \quad (35)$$

From this relation it is possible to construct a travel time curve for the wavefront in (x,t) space due to a point scatterer. It is also possible to construct the wavefront in (x,z) space which is the response of the 45 degree equation to an impulsive point input. The theoretical curves generated agree well with the output from this program--examples of both curves are provided.

The wavefront curves show that the 45 degree equation allows the propagation of anamolous waves with K magnitudes greater than 1. The spurious amplitudes are seen to lie above the impulsive source in the example provided and form a characteristic heart shape on the plot. These waves can be attenuated through the use of dip-filtering and numerical viscosity.

As in the case of the phase velocity a table errors can be formed. This time the position and timing errors are given by

$$(\Delta x, \Delta t) = z \left(-\frac{\partial}{\partial k}\ ,\ \frac{\partial}{\partial \omega} \right) \left(\hat{k}_z - k_z \right)$$

5. *Implementation*

This section will try to explain how the implementation of the 45 degree equation appended operates. Some of the subroutines and systems calls used are peculiar to the computer used at the SEP so an effort will be made to explain them so as to minimize the confusion. Besides these asides, the section will attempt to follow thorugh a single cycle of execution and only touch upon initialization steps.

Before proceeding further it will prove expedient to discuss the variables and names used in the algorithm. The most important variables are upll, upl2, upl3, up21, up22, and up23. These are vectors lying along the x-axis and are the wave variables. Their indices refer to their location in the computational grid. In the time-reversed frame in which it is easiest to migrate for machine-dependent reasons, the 45 degree equation can be written as

$$\begin{bmatrix} -\rho^2 & \rho^2 \\ 2\rho & -2\rho \\ -1 & 1 \end{bmatrix} + aT \begin{bmatrix} -\rho^2 & -\rho^2 \\ 0 & 0 \\ 1 & 1 \end{bmatrix} + bT \begin{bmatrix} -\rho^2 & \rho^2 \\ -2\rho & 2\rho \\ -1 & 1 \end{bmatrix} + \gamma T \begin{bmatrix} \rho^2 & -\rho^2 \\ -2\rho & 2\rho \\ 1 & -1 \end{bmatrix} = 0$$

Indexed positions on the computational grid.

11	21
12	22
13	23

where both sides operate on the wave variable. The upper left hand corner of each grid then operates on the vector upll, the middle left part of each grid on the vector upl2, the lower left hand corners on upl3, the upper right hand corner of each grid on up21, and so on.

In each slot of the computational star are located a number of variables. These are defined as follows:

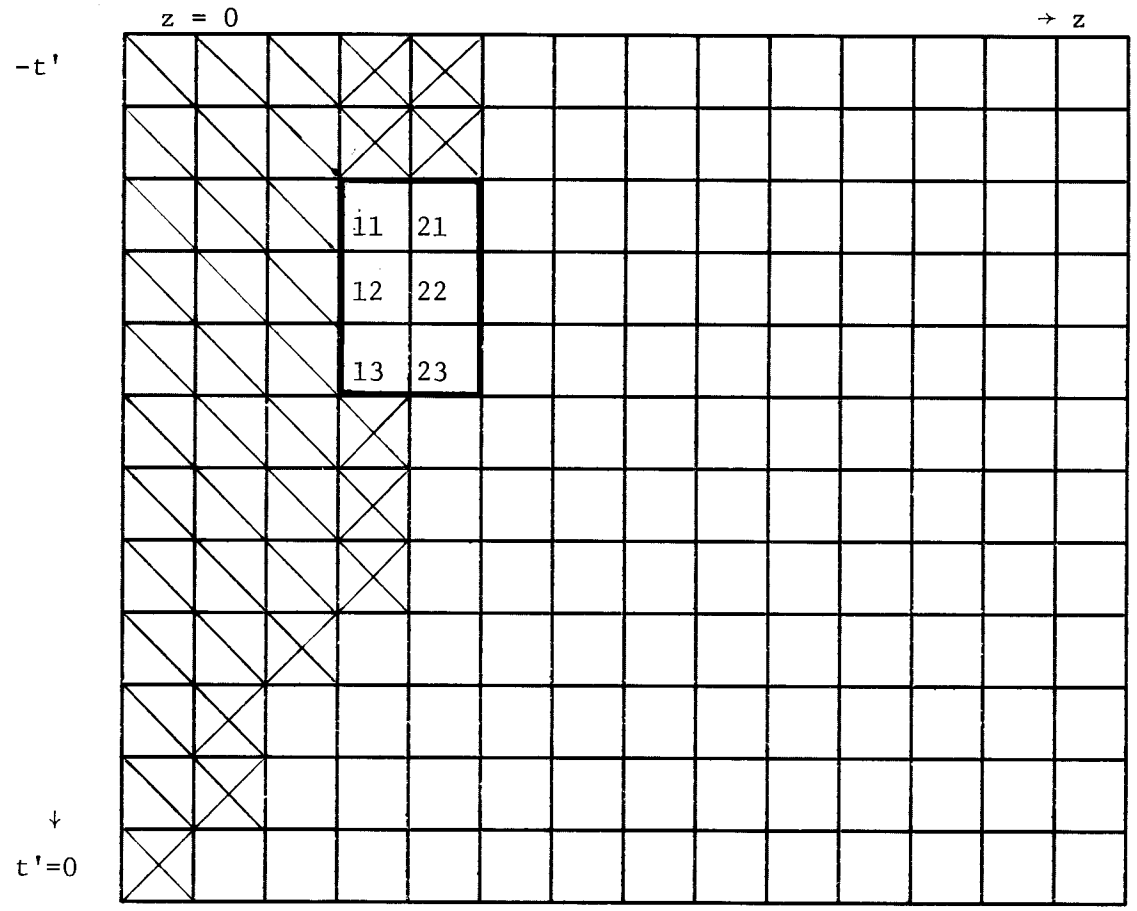
a = the '15 degree' constant
b = the '45 degree' constant
rho numerical viscosity set equal to 0.99
gamma improves approximation to ∂_{xx}; set equal to 1/6
T operationally the negative of the second difference operator on the x-axis

Most of the computation involving these variables is done in the subroutine called 'dset', named that because its output is a vector 'dtri' defined along the x-axis and equivalent to the 'd' which is fed into subrountine tri in FGDP and in this algorithm. The other place in which this kind of number crunching occurs is in the subroutine triset, which calculates the coefficients stri, btri, and ctri to be fed into subroutine tri.

Now for the aside into input-output on the SEP computer. All of the I/O in this program is done through the subroutines called 'uread', 'uwrite', and 'endpt'. These generally take a variable called 'index' which is the size of the vector to be read into storage in bytes, as well as a parameter 'ifld' which is a file descriptor which functions as an address for the location of the first element of the file to which it is assigned. The routine 'endpt' is a subroutine which moves the pointer to the beginning (or in the case of a diffraction program to the end) of the file named by the file descriptor referred it. In most cases, a local variant of 'uread', 'uwrite', and 'endpt' can easily be substituted. A potential source of difficulty and misunderstanding occurs just above the line marked 20. In these lines the file descriptors are interchanged. This means that what was once the input file is now the output file and vice versa. The algorithm proceeds then, by migrating one z-step on file 1 and overwriting the results on file 2. At the next z-step, file 2 is used as input and the output is written on file 1, and so on. This procedure allows a sequential access of files with something close to a minimum of storage (the calculation can also be done in place but that will not be presented here because the procedure is both difficult and slow).

We are now ready for a cycle through a single z-step.

The first step takes place where 'do 20 jz=1,nz-1' appears. The index jz , as the name implies is the z coordinate of the wavefield. It, along with the index kt , also indicates the location of the lower left hand corner of the computational star of 'upl1', 'upl2', ... described above in a grid in (x,-t') space.



After entering the do loop 'endpt' is called to set pointers to the beginning of both the output files. The algorithm then proceeds, in an inner loop, to comb downward (since this is along the reversed time axis, the combing proceeds from positive to zero values of (t=t'-z/v) with the computational star. It first reads a vector of values from the x-axis into upl3 (lying at the lower left hand corner of the star) and feeds all the 'up' variables into the routine 'dset', except for 'up23' which is unknown.

The output from 'dset' is 'dtri', which is input along with a number of coefficients into routine 'tri'. 'Tri' solves the usual tridiagonal system of equations defined along the x-axis and involving the operator

T. Most of the x-derivatives involving T are calculated in 'dset'. In any case, the output is 'up23', which is written into the current output file.

If we are not at a point for which t < 0, then we proceed to shift the values held in the grid of 'up' variables upwards in that grid. The values associated with up13 are reassigned to up12, those in up12 are placed in up11, and so on. This operation, done in the subroutine 'ushift', is equivalent to moving the computational star down in (z,-t') space as diagrammed above. At this point up 13 and up 23 have incorrect values - a situation remedied in the next iteration of the inner do loop.

The only major points remaining for discussion are stopping conditions for the do loops and the location of the final migrated depth section. Since we are migrating zero offset sections at this point, the explosive reflector model holds. In this case, provided we have migrated with half the media velocity, the geophones will be at the reflectors at time t=0 . Our initial time transformation was

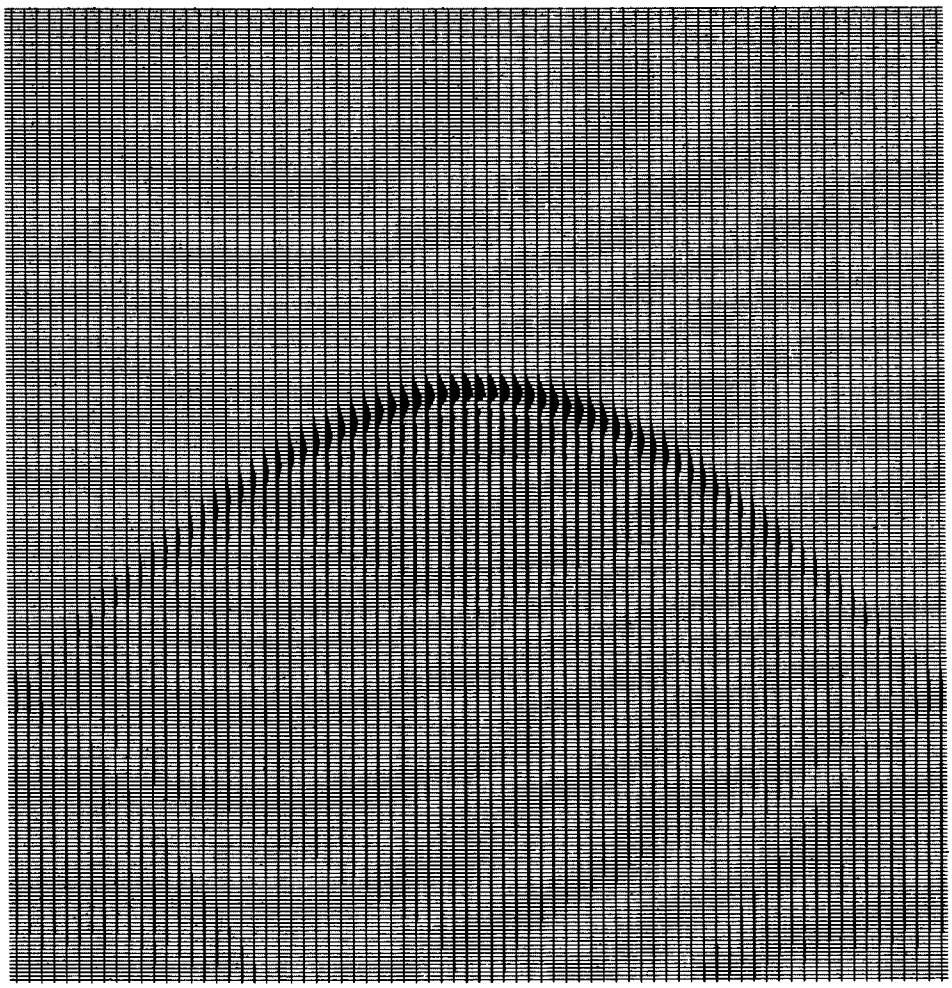
$$t = t' - z/v$$

so we subtrace delz/vel for every z-step undertaken by the outer do loop and subtract delt for every t-step taken in the inner do loop. The time at the beginning of the iteration over the time index is clocked with the variable called 'time' and the time within the inner iteration is clocked with 'tstop'. When 'tstop' is found to be less than zero the inner do loop is exited and another step in z is taken. If it is found that 'time' is less than zero, it is decided that there remain neither z-steps nor t-steps to be taken and both program cycles are exited. The final data lies along a diagonal in (-t',z) space and is easily seen to be located in the last file to be written on.

One last detail is that though diffraction proceeds similarly it was found to be necessary to initiate each comb of the inner do loop differently. The quality of the final diffraction was significantly improved if this initial step in t' was done not with the six-point operator but with a corresponding four-point operator given by the 15 degree wave equation.

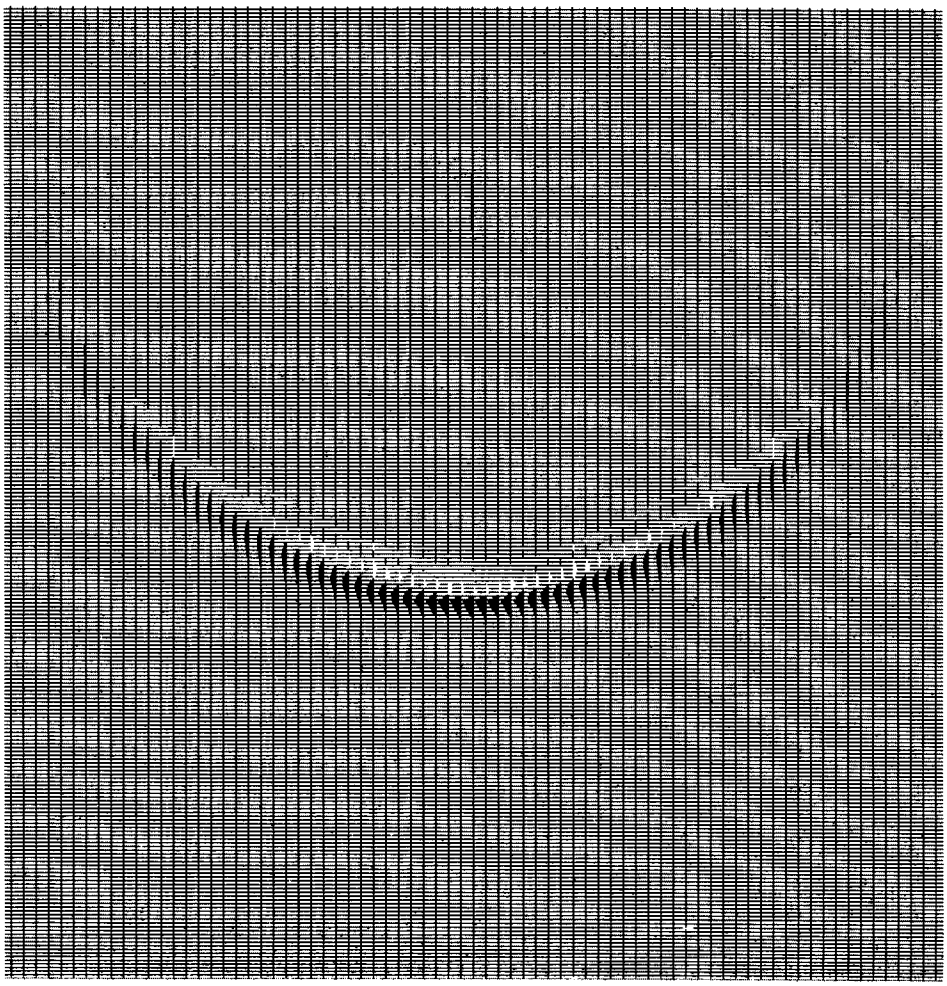
Diffraction from a point scatterer

$v = 1$
$\Delta t = 1$
$\Delta z = 2$
$\Delta x = 2$
$\rho = 0.97$
$\gamma = 6.00$



Result of migration starting with an impulse

$v = 1$
$\Delta t = 1$
$\Delta z = 2$
$\Delta x = 2$
$\rho = 0.97$
$\gamma = 6.00$



```
c       upmig45.f
c
c
c       This program implements the 45 degree wave equation.  It
c       operates on two files, '/scr/bert/migfile' and '/scr/bert/
c       migrated'.  Initially these files are time reversed and
c       unmigrated sections, with data arranged in strips of x.
c       The output must be time reversed to get the final,migrated
c       section.
c
c
c       Variables in order of occurence
c               up11    wave variable lying in the upper left hand
c                       corner of the 6-point computational grid
c                       that combs along the time axis.  The 'up'
c                       variables are vectors defined along x-axis.
c               up12    wave variables in the middle left
c               up13    wave variables in the lower left corner of
c                       the grid into which is read, at each t-step,
c                       a value from the current input file
c               up21    wave variable in the upper right corner
c               up22    wave variable in the middle right
c               up23    wave variable in the lower right.  Tri cal-
c                       culates its value in the course of a t-step.
c                       The new value is then stored in the current
c                       output file.
c               dtri    'd' vector defined along the x-axis and an
c                       input to 'tri' as described in FGDP
c               etri    another input to 'tri'
c               ftri    another input to 'tri'
c               nx      number of traces
c               nt      number of time points
c               ifilds  the address of one of the files operated on.
c                       The address is called a file descriptor.
c               ifiel   address of the other file operated on
c               ifirst  the initial value of ifilds
c               rho     numerical viscosity constant set just smaller
c                       than one - corrects for a DC pole
c               gamma   improves the approximation to the second
c                       derivative with respect to x
c               ammag   the reciprocal of gamma
c               delt    sample time interval
c               delz    desired z-step interval
c               delx    geophone spacing
c               a       15-degree variable
c                       a = (vel*delt*delz)/(8.*delx*delx)
c               b       45-degree variable
c                       b = ((vel*delt)/(4.*delx))**2
c               vel     transformation velocity
c               time    unretarded time coordinate at the start of
c                       each z-step
c               tstop   unretarded time coordinate at each t-step
c               ilast   address of the last file written on
c
c
c       Subroutines in order of occurence
c               files:  opens the files; arguments are the file
c                       addresses
c               getvar: gets migration parameters from the ter-
c                       minal and calculates 'a', 'b', and 'gamma'
```

```
c                begin:   does some more initializations
c                mzero:   zeroes the 6-point star of wave variables
c                endpt:   moves pointer to the beginning fo file
c                         addressed by its arguments
c                uread:   reads the wave amplitude from the last
c                         z-step into the 'up13' location in the
c                         6-point computational grid
c                dset:    prepares the vector called 'd' in FGDP
c                         to be fed into the subroutine 'tri'
c                triset:  calculates the other inputs to 'tri'
c                tri:     solves a tridiagonal system of linear equa-
c                         tions for the wave field at the next
c                         t-step
c                uwrite:  writes the newly obtained wave amplitude
c                         into the next location in the current
c                         output file
c                ushift:  effectively moves the computation star
c                         along the t-axis in (z,t) space
c                finish:  closes the files and indicates which file
c                         was last written on and should be plotted
c
c
c       declarations and initializations
c
c
      dimension up11(80),up12(80),up13(80)
      dimension up21(80),up22(80),up23(80)
      dimension dtri(80),etri(80),ftri(80)
      data nx/80/,nt/100/
      call files(ifilds,ifiel,ifirst)
      call getvar(rho,gamma,delt,delz,delx,a,b,vel)
      call begin(nt,nz,nx,index,delt,time)
c
c
c       outer do-loop over the z-index
c
c
      do 20 jz=1,nz-1
      call mzero(nx,up11,up12,up21,up22)
      call endpt(ifilds,0)
      call endpt(ifiel,0)
      time=time-delz/vel
c
c
c       program is done when there are no points for which t > 0
c
c
      if(time.le.0.) go to 21
      tstop=time
c
c
c       inner do-loop over the t-index
c
c
      do 10 kt=1,nt-1
      call uread(nx,up13,index,ifilds)
      call dset(nx,up11,up12,up13,up21,up22,a,b,dtri,rho,gamma)
      call triset(atri,btri,ctri,a,b,gamma,rho)
      call tri(atri,btri,ctri,nx,up23,dtri,etri,ftri)
      call uwrite(nx,up23,index,ifiel)
```

```
c
c
c       stop moving along t, and z-step instead when t < 0
c
c
        if((tstop-delt).lt.0.) go to 11
        ilast=ifiel
        call ushift(nx,up11,up12,up13,up21,up22,up23)
        tstop=tstop-delt
10      continue
11      continue
c
c
c       interchange the file descriptors making the current input
c       the output file for the next z-step, and vice versa
c
c
        itemp=ifilds
        ifilds=ifiel
        ifiel=itemp
20      continue
21      continue
        call finish(ifilds,ifiel,ifirst,ilast)
        stop
        end
        subroutine getvar(rho,gamma,delt,delz,delx,a,b,vel)
        call setfil(43,'param45',512)
        read(43)vel,delt,delz,delx,rho,ammag
        write(6,46)vel,delt,delz
        write(6,47)delx,rho,ammag
46      format(1x,'vel ',f6.3,' delt ',f6.3,' delz ',f6.3)
47      format(1x,'delx ',f6.3,' rho ',f6.3,' ammag ',f6.3)
        gamma=1./ammag
        a=(vel*delt*delz)/(8.*delx*delx)
        b=((vel*delt)/(4.*delx))**2
        write(6,48)a,b
48      format(1x,'a ',f6.3,3x,' b ',f6.3)
        return
        end
        subroutine uread(nx,v,index,ifld)
        dimension v(nx)
        nread=iread(ifld,v,index)
        return
        end
        subroutine uwrite(nx,v,index,ifld)
        dimension v(nx)
        nwrite=iwrite(ifld,v,index)
        return
        end
        subroutine endpt(ifld,ptrnm)
        nseek=iseek(ifld,0,ptrnm)
        return
        end
        subroutine dset(nx,up11,up12,up13,up21,up22,a,b,dtri,
     1  rho,gamma)
        dimension up11(nx),up12(nx),up13(nx),up21(nx),up22(nx)
        dimension dtri(nx)
        nx1=nx-1
        do 10 ix=2,nx1
        co=-rho*rho
```

```
      dtri(ix)=co*up11(ix)
      deriv=-up11(ix-1)+2. *up11(ix)-up11(ix+1)
      co=rho*rho*(-a-b+gamma)
      dtri(ix)=dtri(ix)+co*deriv
      co=2. *rho
      dtri(ix)=dtri(ix)+co*up12(ix)
      deriv=-up12(ix-1)+2. *up12(ix)-up12(ix+1)
      co=-2. *rho*(b+gamma)
      dtri(ix)=dtri(ix)+co*deriv
      co=-1.
      dtri(ix)=dtri(ix)+co*up13(ix)
      deriv=-up13(ix-1)+2. *up13(ix)-up13(ix+1)
      co=a-b+gamma
      dtri(ix)=dtri(ix)+co*deriv
      co=rho*rho
      dtri(ix)=dtri(ix)+co*up21(ix)
      deriv=-up21(ix-1)+2. *up21(ix)-up21(ix+1)
      co=rho*rho*(-a+b-gamma)
      dtri(ix)=dtri(ix)+co*deriv
      co=-2. *rho
      dtri(ix)=dtri(ix)+co*up22(ix)
      deriv=-up22(ix-1)+2. *up22(ix)-up22(ix+1)
      co=2. *rho*(b+gamma)
10    dtri(ix)=dtri(ix)+co*deriv
      return
      end
      subroutine mzero(nx,up11,up12,up21,up22)
      dimension up11(nx),up12(nx),up21(nx),up22(nx)
      do 10 i=1,nx
      up11(i)=0.
      up12(i)=0.
      up21(i)=0.
10    up22(i)=0.
      return
      end
      subroutine triset(atri,btri,ctri,a,b,gamma,rho)
      atri=a+b-gamma
      btri=-1.-2. *atri
      ctri=atri
      return
      end
      subroutine tri(atri,btri,ctri,nx,ttri,dtri,etri,ftri)
      dimension ttri(nx),dtri(nx),etri(nx),ftri(nx)
      nx1=nx-1
      etri(1)=1.
      ftri(1)=0.
      do 10 i=2,nx1
      den=btri+ctri*etri(i-1)
      etri(i)=-atri/den
10    ftri(i)=(dtri(i)-ctri*ftri(i-1))/den
      ttri(nx)=ftri(nx1)/(1.0-etri(nx1))
      do 20 j=1,nx1
      i=nx-j
20    ttri(i)=etri(i)*ttri(i+1)+ftri(i)
      return
      end
      subroutine ushift(nx,up11,up12,up13,up21,up22,up23)
      dimension up11(nx),up12(nx),up13(nx)
      dimension up21(nx),up22(nx),up23(nx)
      do 10 i=1,nx
```

```
      up11(i)=up12(i)
      up12(i)=up13(i)
      up21(i)=up22(i)
10    up22(i)=up23(i)
      return
      end
      subroutine begin(nt,nz,nx,index,delt,time)
      nz=nt
      index=4*nx
      time=(nt-1.)*delt
      return
      end
      subroutine files(ifilds,ifiel,ifirst)
      logical *1 infil(50),outfil(50)
      data infil/'/scr/bert/migfile'/
      data outfil/'/scr/bert/migrated'/
      ifilds=iopen(infil,2)
      ifiel=iopen(outfil,2)
      ifirst=ifilds
      write(6,51)ifilds,ifiel,ifirst
51    format(1x,'ifilds ',i4,' ifiel ',i4,' ifirst ',i4)
      return
      end
      subroutine finish(ifilds,ifiel,ifirst,ilast)
      ichk=iclose(ifilds)
      ichk=iclose(ifiel)
      if(ifirst.eq.ilast) write(6,82)
      if(ifirst.ne.ilast) write(6,83)
81    format(3x,'I am done migrating.')
82    format(3x,'The output is in /scr/bert/migfile.')
83    format(3x,'The output is in /scr/bert/migrated.')
      return
      end
```