The Generalized Frequency-Dependent Surface-Consistent Statics Problem

Chuck Sword

Introduction

In a recent paper, Taner and Koehler (1981) showed that it is possible to find, surface-consistently, how the amplitude of traces varies as a function of both frequency and shot and geophone position. In this paper, I will describe a further extension of the surface-consistent method. Previous methods found surface-consistent amplitudes and time-shifts; I am attempting to find surface-consistent phases as a function of frequency. So far, I have not succeeded in applying the method to real data, but I have gotten some promising results with synthetic examples.

My paper is divided into four parts. In the first part, I review the problem of surface consistency, show what it means to find surface-consistent phase functions, and discuss the problems inherent in dealing with real data. In the second and third parts, I discuss problems that I have encountered in implementing this scheme, and how I have dealt with them. In the fourth part, I show the results of processing some synthetic data. The first problem was how to solve least-squares systems that involve phase; the problem stems from the fact that while it is impossible to determine from a lone trace whether its phase at a particular frequency is φ, or $\varphi + 2\pi$, or $\varphi + 4\pi$, etc., the final result will differ depending on which of these phases is assumed to be the correct one. I believe that this problem is now solved. The second problem is how to deal with null spaces. This problem was discussed by Wiggins et al (1976). The effects of null spaces become much more serious when one attempts to solve the statics problem at more than one temporal frequency, and at the same time, these effects become much more difficult to remove when one is dealing with phases rather than with amplitudes. I am not sure if I have a good solution to this problem.

Surface Consistency and How to Generalize It

In this section I will give a brief historical review of the problem, and show some of the similarities and differences between my approach and previous approaches. Then I will discuss the problems in applying my generalized method to real data.

In their well-known 1974 paper, Taner et al discussed the idea of modeling static corrections using the formula:

$$T_{ij} = S_i + G_j + Y_{\frac{(i+j)}{2}} + (j-i)^2 M_{\frac{(i+j)}{2}} \tag{1}$$

where i is the shot position index, j is the geophone position index, T_{ij} represents the observed time shift at a shot-geophone position i, j, and S, G, Y, and M represent time shifts due to shot, geophone, midpoint, and residual moveout anomalies. T_{ij} is assumed to be known, and an iterative least-squares procedure can be devised to find S, G, Y, and M.

In a later paper, Taner and Koehler (1981) carried this idea a step further. Suppose that any trace $F_{ij}(t)$ can be described as a convolution of effects due to geology, shot position, geophone position, and offset. Then we can write:

$$F_{ij}(t) = S_i(t) * G_j(t) * Y_{\frac{(i+j)}{2}}(t) * H_{\frac{(i-j)}{2}}(t) \tag{2}$$

Going into the frequency domain, we can rewrite equation (2) as:

$$F_{ij}(\omega) = S_i(\omega) G_j(\omega) Y_{\frac{(i+j)}{2}}(\omega) H_{\frac{(i-j)}{2}}(\omega) \tag{3}$$

By taking the log of both sides, the equation becomes linear:

$$\ln F_{ij}(\omega) = \ln S_i(\omega) + \ln G_j(\omega) + \ln Y_{\frac{(i+j)}{2}}(\omega) + \ln H_{\frac{(i-j)}{2}}(\omega) \tag{4}$$

One interesting effect of taking the log is that the real part of $\ln F_{ij}(\omega)$ is related to the amplitude of $F_{ij}(\omega)$, while the imaginary part is related to the phase of $F_{ij}(\omega)$. Thus, by separately equating the real and imaginary parts of equation (4), we obtain two linear equations: one that describes surface-consistent amplitudes as a function of frequency, and one that describes surface-consistent phases as a function of frequency.

Once equation (4) had been split into two parts, Taner and Koehler went on to solve the real (amplitude-related) equation, but they did not attempt to solve the second, phase-related equation. Instead, they made the assumption that the phase component can be approximated by a linear function of ω in the frequency domain, which corresponds to a simple time shift in the time domain. In so doing, they made their problem much easier. Their problem was now reduced to just two procedures: find the static shifts, just as they did in

(Taner et al 1974), and find surface-consistent amplitudes as a function of frequency.

My paper generalizes the above method. Rather than making the approximation that phase is a linear function of ω, I will assume that phase can vary arbitrarily with frequency, just as Taner and Koehler assumed that amplitude could vary arbitrarily with frequency. This assumption leads, as I will show later, to several complications, since there are certain difficulties involved in solving phase-related problems by least-squares methods.

Before discussing how to solve the generalized surface-consistency problem that has just been posed, let's first examine what this problem means and how closely it is related to the problems that are encountered in real data.

Imagine an idealized seismic experiment. An impulsive source at location i is set off, and trace $U_{ij}(t)$ is recorded at geophone position j. The geophone produces no phase or amplitude changes in the signal, and there are no shot or geophone statics.

So much for the ideal case. Now consider reality. The shot has some sort of waveform, which may vary from shot position to shot position, and there are shot statics that vary with position. The geophone is imperfect as well, introducing phase shifts and amplitude losses, and of course there are geophone statics. Assuming that the shot and geophone effects are reasonably linear, and that the static shifts are not dependent on the ray angle, we can consider the shot and geophone effects to be convolutions onto the original ideal trace $U_{ij}(t)$. We can then speak of the time functions $S_i(t)$ and $G_j(t)$, which represent the waveforms to be convolved onto the trace $U_{ij}(t)$, so we can say that our actual data is represented by $S_i(t) * G_j(t) * U_{ij}(t)$. Note that $S_i(t)$ varies not only with time, but with shot position as well, and that the corresponding statement may be made about $G_j(t)$. This is simply saying that the shot waveform and shot statics vary with the shot position, and that the geophone impulse response and geophone statics vary with the geophone position.

So now we have justified the $S_i(t)$ and $G_j(t)$ terms of equation (4). But how do we justify the $Y_{(i+j)/2}(t)$ and $H_{(i-j)/2}(t)$ terms of (4)? Here is where equation (4) diverges from reality. We might say that the common-midpoint term, $Y_{(i+j)/2}(t)$, corresponds to the waveform that would be received in an idealized zero-offset experiment at a given midpoint. Then $U_{ij}(t)$ could be considered to be composed of the common-midpoint waveform after it has undergone a normal-moveout stretching which depends on offset h ($h \equiv (i-j)/2$), and after it has undergone amplitude changes because of reflection coefficients that vary as a function of angle. The amplitude variations as a function of h can probably be handled as a convolution. The NMO stretching, however, certainly can not. Therefore, if we write $U_{ij}(t) = Y_{(i+j)/2}(t) * H_{(i-j)/2}(t)$, we are departing from reality.

We have a problem, then, when it comes to dealing with normal moveout. It might be supposed that we could simply apply a normal-moveout correction to the data in order to eliminate this problem. The difficulty, however, is that we would like to apply the correction to $U_{ij}(t)$, but all we have to work with is the data $S_i(t)*G_j(t)*U_{ij}(t)$. If we apply an NMO correction to this data, we destroy the validity of the convolution, since the NMO correction takes the form of a time-varying stretch.

So this is the problem with using real data. On the one hand, we need to do an NMO correction in order to determine $Y_{(i+j)/2}(t)$. On the other hand, if we perform such a correction, we destroy (or at least distort) the information that would allow us to determine $S_i(t)$ and $G_j(t)$. The solution I propose is rather crude and has not yet been fully tested:

1) On a common-midpoint gather, choose a horizon to be flattened and determine its normal hyperbolic moveout.

2) Apply a time shift to the gather, corresponding to the normal moveout at that given horizon. In other words, flatten the horizon, but by applying a time shift rather than by performing a stretch.

3) Window the data so as to look at only a short period of time around the horizon.

If the time window is not too wide, a time shift should be as effective as a stretch in removing the effect of normal moveout, thus allowing us to write with some validity, $U_{ij}(t) = Y_{(i+j)/2}(t)*H_{(i-j)/2}(t)$. And since we use a time shift rather than a stretch, then if the static shifts are not too large, we can say that the final trace equals $S_i(t)*G_j(t)*U_{ij}(t)$. Therefore, equation (4) should hold in this situation. However, if the time window is too narrow, we will have difficulty determining the Fourier transform of the data. The phase especially will be affected by the truncation problem (Mitrofanov, 1979).

Thus we see the problems involved in dealing with real data. But what are the possible applications of using this procedure? One application, of course, is the removal of complicated statics effects; these effects may include a varying shot waveform (if the data was recorded using an explosive source) and strange filtering effects caused by the ground directly underneath the shot and geophone. Another possible application is the removal of multiple reflections (Morley, 1982). I should note, however, that although I have mentioned the removal of shot waveforms, I am not suggesting that this statics method could be used to do deconvolution. When I write of removing the effect of the shot waveform, I only mean removing the effects of a *varying* shot waveform. If my procedure works properly, the data will be adjusted so that it appears to have been taken using a non-varying, but not necessarily zero-phase, shot.

Solving for Phases by Least-Squares Methods

Now that we have seen some justification for doing so, let us return to equation (4) and redefine some terms. First we define $F_{ij}'(\omega) \equiv \ln F_{ij}(\omega)$, and make analogous definitions for the other terms of (4). Then we throw away the primes. Then, for convenience, we throw away the "/ 2" parts of $Y_{(i+j)/2}(\omega)$ and $H_{(i-j)/2}(\omega)$. Now we can write:

$$F_{ij}(\omega) = S_i(\omega) + G_j(\omega) + Y_{i+j}(\omega) + H_{i-j}(\omega) \tag{5}$$

But we have not considered the possibility of noise in the data. Ideally, we should write:

$$D_{ij}(t) = S_i(t) * G_j(t) * Y_{i+j}(t) * H_{i-j}(t) + N_{ij}(t) \tag{6}$$

where $D_{ij}(t)$ represents the actual recorded data, and $N_{ij}(t)$ represents the noise on each trace. It proves more convenient, however, to write:

$$D_{ij}(t) = S_i(t) * G_j(t) * Y_{i+j}(t) * H_{i-j}(t) * N_{ij}(t), \tag{7}$$

because then when we go through the transformation to the frequency logarithmic domain, we come up with this analog to (5):

$$D_{ij}(\omega) = S_i(\omega) + G_j(\omega) + Y_{i+j}(\omega) + H_{i-j}(\omega) + N_{ij}(\omega), \tag{8}$$

which can be analyzed in terms of a least-squares problem.

Let us look at the least-squares problem that we are setting up for ourselves. At a given frequency, our minimization problem is:

$$N_{ij} = D_{ij} - S_i - G_j - Y_{i+j} - H_{i-j}; \tag{9}$$

Find S_i, G_j, Y_{i+j}, and H_{i-j}

such that $\sum\limits_{i,j} | N_{ij} |^2$ is minimized.

This is relatively straightforward to do for the real part of S_i, G_j, Y_{i+j}, and H_{i-j}. The difficulty comes with the imaginary part. The imaginary part of $D_{ij}(\omega)$ corresponds to the phase in the original data at frequency ω. Since multiples of 2π can be added to this phase without causing any disagreement with the original time-domain data $D_{ij}(t)$, an extra element of indeterminacy is added to the problem. Not only do we need to know which values of S_i, G_j, Y_{i+j}, and H_{i-j} minimize equation (9), we need to figure out what multiple of 2π to add to the imaginary part of D_{ij}. To some extent, phase unwrapping might serve, but although phase unwrapping will give us the phase at one frequency relative to that at another frequency, we still must figure out the correct phase (including multiples of 2π) at our starting frequency. Thus while phase unwrapping may speed up our process once we get started (although I have not tried this idea out yet), it doesn't solve the problem of how to get the phases at our starting frequency.

Taner et al (1974) use Gauss-Seidel iteration in order to solve for S_i, G_j, Y_{i+j}, and H_{i-j}, when they are looking for simple static shifts. This is an iterative method, and I will use a variation of it to solve the current problem. The basic idea is that in order to solve the least-squares minimization problem, we first must solve over S_i, then over G_j, then over H_{i-j}, and finally over Y_{i+j}. We continue solving over these four terms until our solution converges. But what do I mean by "solving over S_i"?

Let us define:

$$n_i \equiv \sum_j | D_{ij} - S_i - G_j - Y_{i+j} - H_{i-j} |^2 \qquad (10)$$

Then we want to find S_i so that n_i is minimized, solving for S_i at one value of i (the shot coordinate) at a time. When doing so, we assume that G_j, Y_{i+j}, and H_{i-j} are known. This is what I mean by "solving over S_i". (Solving over G_j is similar, except that you sum over i and minimize n_j, which is defined similarly to n_i above. For H_{i-j} and Y_{i+j}, you must change coordinates.)

It is useful to note at this point that we can solve separately for the real and imaginary parts of S_i (and similarly for G_j, etc.). Solving to find the real part is fairly easy. Solving to find the imaginary part is quite a bit trickier, because of the phase problem mentioned above. Most papers dealing with frequency-dependent surface-consistent statics [(Taner and Koehler, 1981), (Morley, 1982)] simply solve for the real part only. This corresponds to finding the surface-consistent amplitude. Phase is dealt with only as an over-all static shift, and thus is not considered to be anything other than a linear function of frequency. (Some Soviet researchers, however, have published results that suggest that they have successfully dealt with the frequency-dependent phase problem (Mitrofanov et al, 1982).

In any case, let us look first at how to solve for the real part of S_i, using equation (10). The minimization problem is simple: find S_i such that $\partial n_i / \partial S_i = 0$. Differentiating equation (10) and solving for S_i, we find that

$$S_i = \frac{1}{N_j} \left[\sum_j D_{ij} - G_j - Y_{i+j} - H_{i-j} \right] \qquad (11)$$

Here N_j means the number of geophone points j at shotpoint i. To look at this another way, let us define

$$M_{ij} \equiv D_{ij} - G_j - Y_{i+j} - H_{i-j}, \qquad (12)$$

so that M_{ij} represents the result of subtracting the "known" information -- G_j, Y_{i+j}, and

H_{i-j} -- from the data D_{ij}. Then we easily see that in order to minimize n_i, we must find a value for S_i which represents the average, over all data from shotpoint i, of M_{ij}:

$$S_i = \frac{1}{N_j}\sum_i M_{ij} \tag{13}$$

This concept of taking an average will prove important when we look at phase.

When we try to find the phase (imaginary) component of S_i, we are again attempting to minimize n_i in equation (10), which we will rewrite using the definition of M_{ij} from equation (12):

$$n_i \equiv \sum_j |M_{ij} - S_i|^2 \tag{14}$$

This equation is not accurate, however, since it pre-supposes that we know the phase of M_{ij}. As we have noted previously, a component of M_{ij} (namely, D_{ij}) is ambiguous because of an arbitrary multiple of 2π which can be added to it. So in actuality, we should write:

$$n_i \equiv \sum_j |M_{ij} + 2\pi L_{ij} - S_i|^2 \tag{15}$$

where L_{ij} is some arbitrary integer which can vary with both i and j.

Determining L_{ij} could be quite difficult, but fortunately it is possible to proceed without solving this particular problem explicitly. Instead we note that in equation (15) we want to find both an S_i and an L_{ij} that will minimize n_i. Clearly, n_i will be minimized when the values of $|M_{ij} + 2\pi L_{ij} - S_i|$ are minimized for each j. But for any arbitrary S_i, j, and M_{ij}, the expression $|M_{ij} + 2\pi L_{ij} - S_i|$ will be minimized when L_{ij} is chosen so that $-\pi < M_{ij} + 2\pi L_i - S_i \leq \pi$. Thus we can rewrite equation (15) as

$$n_i \equiv \sum_j |\mathrm{wrap}(M_{ij} - S_i)|^2 \tag{16}$$

where wrap(x) is the function that adds (or subtracts) multiples of 2π to an arbitrary x until $-\pi < \mathrm{wrap}(x) \leq \pi$. In effect, wrap($x$) is a sort of modulo (not modulus) function.

In equation (16) we have gotten rid of the L_{ij} term, so we won't have to worry about it any more. In exchange, however, we have converted a straightforward linear least-squares minimization problem into something a bit more complicated. Previously, n_i varied smoothly with S_i, so it was not too difficult to find the S_i that minimized n_i for an arbitrary i. Now, however, the n_i in equation (16) is no longer smooth as S_i varies. It is still continuous, but its first derivative is not. We cannot find the absolute minimum of n_i simply by solving for $\partial n_i / \partial S_i$, since there are probably a number of local minima. It is still possible to find an absolute minimum, however, by trying various values of S_i in a systematic way.

Once I derived this technique (with the help of some extremely useful suggestions by Jeff Thorson), I attempted to solve the least-squares minimization problem, using some synthetic data that I knew to be noise-free. Clearly, then, I would have a correct answer if the N_{ij} in equation (9) turned out to be identically zero for all values of i and j. I discovered that I could converge to the correct answer if the initial guesses for S_i, G_j, Y_{i+j}, and H_{i-j} were reasonably close to the correct answer, but otherwise $\sum_{i,j} | N_{ij} |^2$ would converge on some non-zero value. I suspect that this problem is somehow caused by the sharp discontinuities in the derivative of n_i and in the other analogous terms (n_j, etc.).

Since this method failed, I cast around for another way of minimizing equation (16). I recalled that for the real (amplitude) component, n_i was minimized by setting S_i equal to the average of M_{ij} [recall equation (13)]. But how could I extend this idea to to a problem involving cyclic phases? I had the following problem:

Suppose you have a group of n phases, θ_0, θ_1, θ_2,... θ_n. All of these phases are in the range $-\pi < \theta \leq \pi$. How would you find the average phase? Clearly you can't simply add the phases together and divide by n. As an example: Suppose you had only 2 phases, $3\pi/4$ and $-3\pi/4$ [see Figure (1a)]. The simple arithmetic average is 0π or 0. But obviously a much better answer is π. The problem gets worse as n increases.

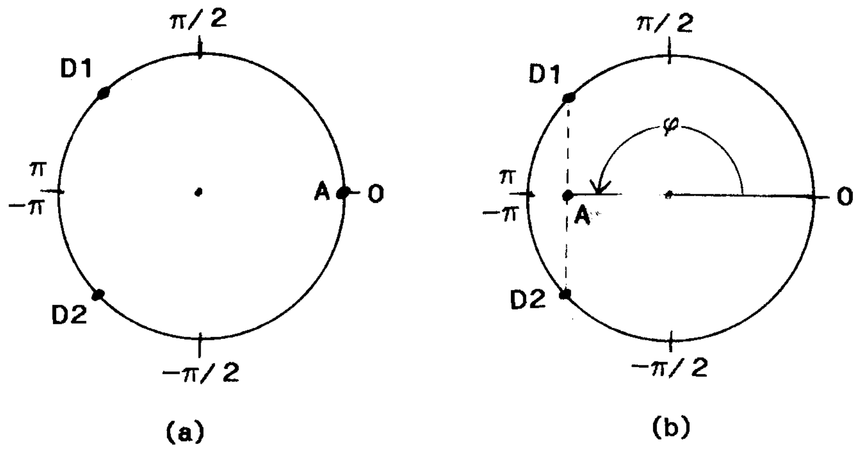


FIG. 1. Finding the average phase. (a) shows why an arithmetic average is not a good way of finding the average phase. D1 and D2 represent two data points, and A represents their arithmetic average. (b) show that a better average is found by finding the center of mass of D1 and D2, and letting the angle of that point be the average phase.

From Figure (1b) we see a way of finding the solution: plot the phases as points on a circle, and find the average position (i.e. the center of mass) of all these points. Compute the phase of the average position relative to the origin, and this will serve as the "average phase". I tried applying this method to my iteration problem. For a given i, I plotted all the relevant values of M_{ij} on a unit circle, and found the phase of the center of mass. I set S_i equal to this phase. I did this for all values of i. Then I did the analogous thing with G_j, etc. The process converged! Slowly but inexorably, as I repeatedly iterated over S, G, H, and Y, N_{ij} went to zero for all values of i and j.

I have since found that more iterations are required for convergence as the size of the problem increases (i.e. as $i_{\max}$ and $j_{\max}$ increase), and as the noise level increases. Of course, as the problem grows larger, the time required per iteration also increases. It may be that for problems beyond a certain size, or for a signal-to-noise ratio below a certain value, convergence will never occur, and the problem will become insoluble. In addition, I am no longer solving an exact least-squares problem, that is, if noise is present, I am no longer exactly solving the minimization problem given in equation (9). In fact, I am not really sure what problem I am now solving. This is the only method I have gotten to work, however. Methods based on minimizing the L_2 norm (least-squares) and the L_1 norm (the median) both fail to converge to zero.

The Problem of Null Spaces

It might seem that the generalized frequency-dependent surface-consistent statics problem has now been solved. Given D_{ij} at a given frequency, we are able to solve for both the real and imaginary parts of S_i, G_j, Y_{i+j}, and H_{i-j}. We should now be able to find these surface-consistent components for all values of ω, exponentiate in order to undo the logarithm that we took previously, and transform into the time domain. Then we could deconvolve the data by applying the newly-found S_i, G_j, and H_{i-j} waveforms, and thus remove the statics. Unfortunately, it's not that easy. We now encounter the problem of null spaces.

I will not discuss the general null space problem here; a good discussion of the problem is given by Wiggins et al (1976). Instead, I will merely give the result. Using a method similar to that given in the appendix of the paper by Taner et al (1974), it is possible to show that if S_i, G_j, Y_{i+j}, and H_{i-j} represent a good decomposition of D_{ij}, that is, if they minimize $\sum_{i,j} | N_{ij} |^2$, then an equally good decomposition is given by:

$$\begin{aligned} S_i' &= S_i + K_0^S + iK_1^S + i^2K_2^S \\ G_j' &= G_j + K_0^G + jK_1^G + j^2K_2^G \\ Y_{i+j}' &= Y_{i+j} + K_0^Y + (i+j)K_1^Y + (i+j)^2K_2^Y \\ H_{i-j}' &= H_{i-j} + K_0^H + (i-j)K_1^H + (i-j)^2K_2^H \end{aligned} \tag{17}$$

There are several several equations relating the various constant K factors. Only six of these factors turn out to be independent; one possible listing of the independent factors is: K_0^S, K_0^G, K_0^H, K_1^S, K_1^G, K_2^S. As an example, it should not be too difficult to see that $K_0^S + K_0^G + K_0^H + K_0^Y$ must equal zero if the new decomposition is to be as valid as the old one, and thus that K_0^Y can be considered to be dependent on the other three constants.

The six constants K_0^S, K_0^G, K_0^H, K_1^S, K_1^G, K_2^S can be considered to be the components of our null space. These six constants are entirely arbitrary, and once they have been fixed, they can be used to determine the other six constants in equations (17) above. There are, in fact, other components of the null space, caused by truncation effects at the ends of the data set, but we are not concerned with them here.

One trouble with the null-space components is that when we use conventional Gauss-Seidel iteration, they do not go away. As Wiggins et al (1976) pointed out, the null space components depend only on the choice of initial vectors S_i, G_j, Y_{i+j}, and H_{i-j}. The simple iteration scheme that I have described will not make the null-space components grow, but neither will it make them go away. Minor variations on the Gauss-Seidel iteration method have been suggested in order to minimize the null-space terms (Taner et al, 1974), but these, I have found, are not effective in reducing the null space in the phase component of the data. The problem, presumably, is related to those multiples of 2π that plagued us in the previous section of this paper.

Before I describe my (not entirely satisfactory) approach to solving the null-space problem, let us examine why null spaces can cause significant difficulties in multi-frequency data. Suppose that we have a data set with no static shifts and with a shot waveform that looks like a delta function. Then $S_i(t)$ should look like Figure (2a). Now suppose that some of our null-space components become non-zero. For instance, if K_0^S takes on one non-zero value for high frequencies, and a different non-zero value for low frequencies, then Figure (2b) is the result of plotting S_i'. If K_1^S now takes on one non-zero value for low frequencies, and another for high frequencies, then Figure (2c) is the result. Note that this is a very simplified example. In reality, every different frequency can take on a different value of K_0^S and K_1^S. Even in this simple case, however, we can begin to see problems. For instance, in Figure (2b) it is possible to remove the effects of the null-space component by simple deconvolution, but in Figure (2c), deconvolution clearly won't be sufficient. And keep in mind the fundamental difficulty, which is that the functions in Figures (2b) and (2c) satisfy the minimization condition (9) just as completely as the function in Figure (2a) does. We need to come up with an additional condition for S_i, G_j, etc. to fulfill, and we need to find some scheme that will force S_i, G_j, etc., to fulfill this condition.

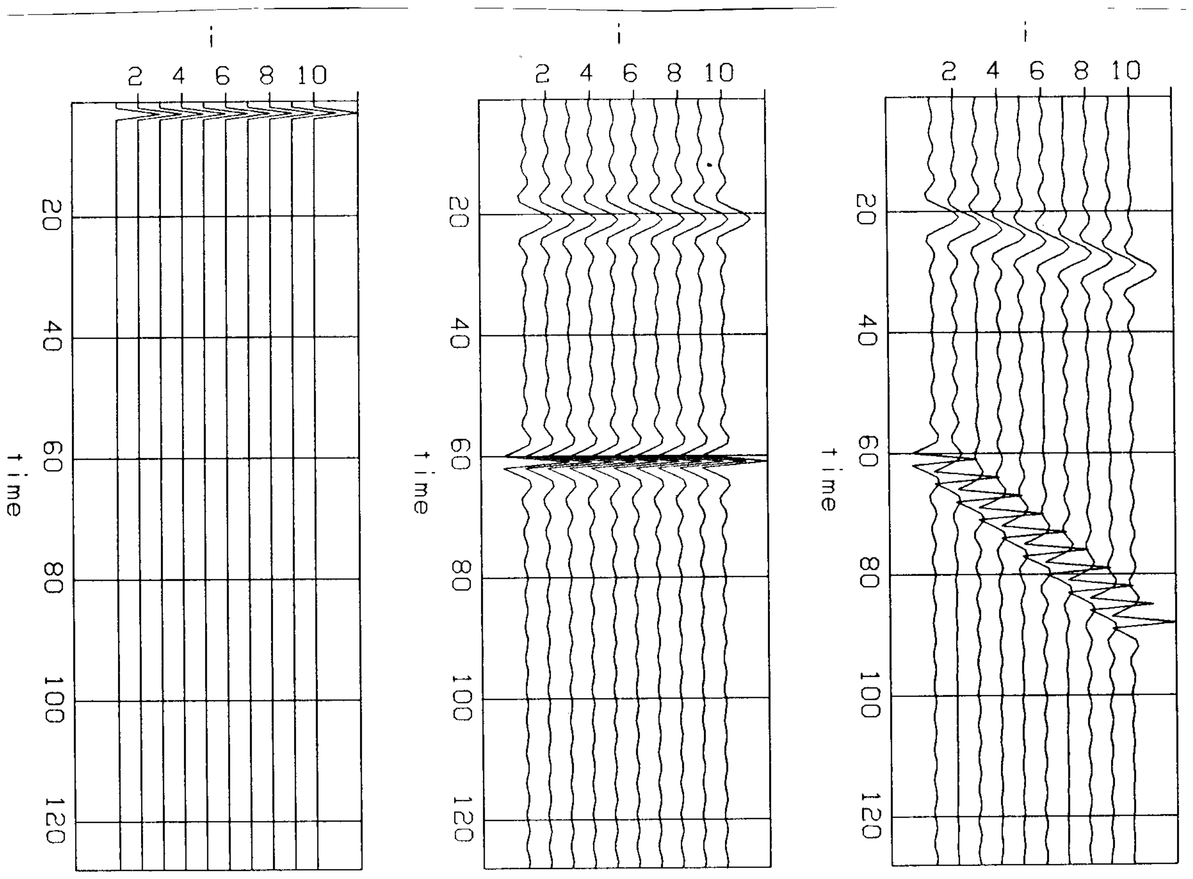


FIG. 2. The effect of null spaces. (a) shows the ideal result. (b) shows the effect when null-space component K_0 varies irregularly with frequency, and (c) shows the effect when K_1 varies irregularly as well.

I have come up with a condition that I can't really express quantitatively, and an iteration scheme that seems to work but that isn't completely convincing. Let's start with the condition, which I will call (probably incorrectly) the parsimony condition: we want the S, G, Y, and H components to look (in the time domain) as much like delta functions as possible. Notice, first of all, that this condition, unlike the minimization condition (9), looks at the entire waveform rather than at one frequency at a time. This is good, since as long as we treat each frequency independently, we are unlikely to get something that will look good when we go back to the time domain. Next, notice what it will not do. Even in the case where there are no static shifts, our parsimony condition does not guarantee that $S_i(t)$ will end up looking like Figure (3a). Figure (3b) or (3c) is just as likely to result. These latter two figures clearly show the presence of null space components, but the important thing is that these null space components no longer vary irregularly as a function of frequency ω.

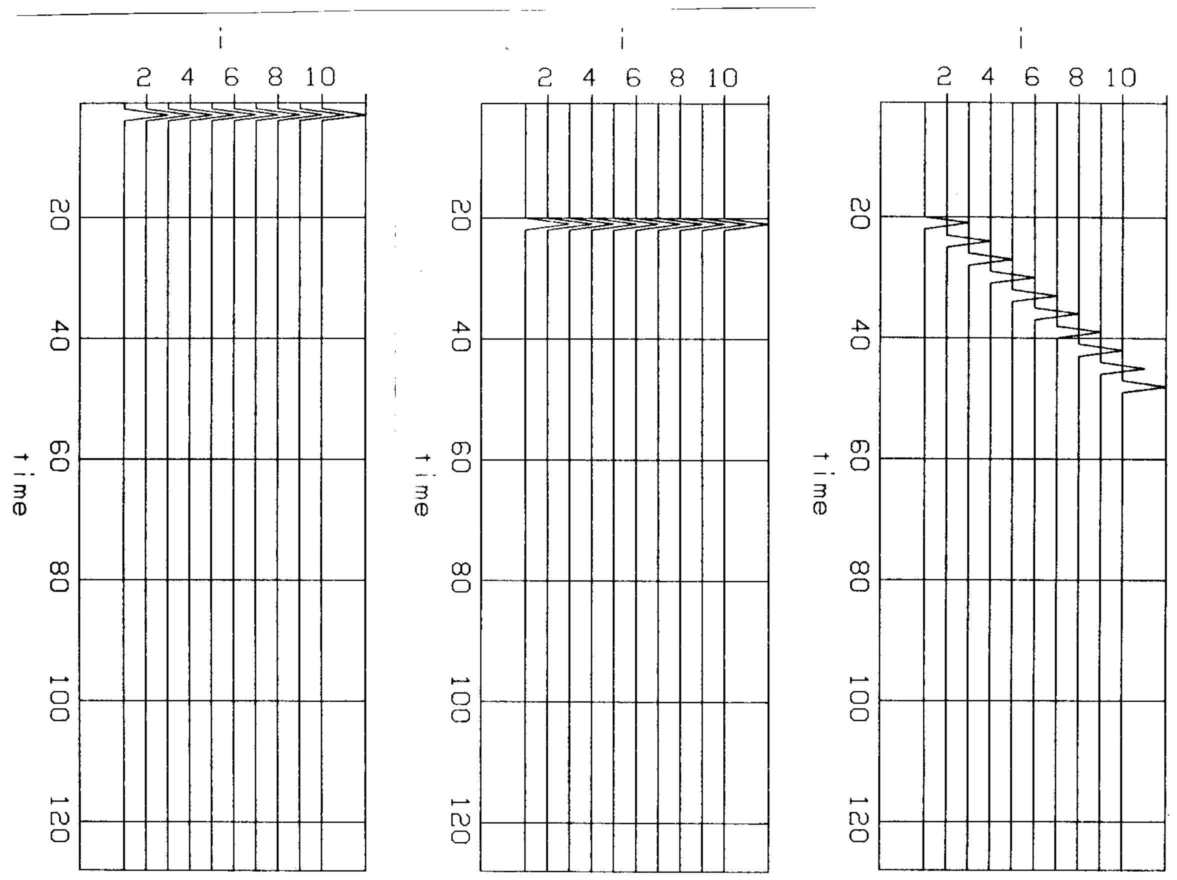


FIG. 3. Results of suppressing the null-space components. (a) shows an ideal result, (b) shows the result when null-space component K_0 varies regularly with frequency, and (c) shows the result when K_1 varies regularly as well.

Now for the conditioning scheme. First of all, note that it differs according to which component (S, G, Y, or H) is being conditioned. This results from the fact that some of the null-space components are dependent on other components. Recall that I chose the independent components to be: K_0^S, K_1^S, K_2^S, K_0^G, K_1^G, and K_0^H. This means that I must find three components for each frequency of S, two for each frequency of G, one for each frequency of H, and none at all for Y. But since I am not sure how to solve for it, I am going to ignore the K_2^S term. Also, note that all of the K terms have a real and an imaginary part, corresponding to amplitude and phase respectively. I don't intend to discuss the real (amplitude) part, since removing the amplitude component from the null space is presumably fairly straightforward if one uses the variation on the Gauss-Seidel method that was mentioned in the appendix of the paper by Taner et al (1974). With these things in mind, let's see how to go about solving for K_1^S and K_0^S.

Let us first consider the case where the K_1^S component has already been removed somehow, and look at how to go about solving for K_0^S. Recall from Figure (2b) that this is analogous to deconvolving the data. We need to find a single K_0^S for each frequency, and when this component is subtracted from each trace (in the log-frequency domain), the result should be a set of traces that are a little bit more deconvolved than they were before. Thus we want to look at all the traces at a given frequency, and come up with a K_0^S that, overall, best compresses the set of traces that make up S_i.

One naive approach might be to look at all the traces at a given frequency, and measure the difference in phase between each trace and an ideal zero-phase trace. Then it would be possible to average all these phase differences, and thus come up with a value for K_0^S. The problem is that a zero-phase trace would be centered at $t=0$, and it is not very reasonable to assume (as we are) that all of the data traces can be forced to look like wavelets centered at $t=0$. In other words, it is assumed in this method not only that all of the S_i traces were convolved with the same wavelet, but that they all have the same time-shift as well. I have thus come up with a deconvolution method that is based on the assumption that all of the traces are identical. This is not too useful.

The previous method can be modified to be a bit more realistic. Again we look at each trace at the given frequency, and again we measure a phase difference at each trace. This time, however, we measure the phase difference differently. Previously, if the phase of trace l at frequency ω_0 were written as $\varphi_l(\omega_0)$, we would say that the phase difference for this trace was $\varphi_l(\omega_0) - 0 = \varphi_l(\omega_0)$, since we were finding the phase difference between the trace and a zero-phase wavelet. But now let us write the phase difference as $\varphi_l(\omega_0) - i\omega_0 t_{l,max}$, where $t_{l,max}$ is the time at which trace l has its maximum value [See Figure (4)]. In other words, previously we were attempting to make all the traces look like delta functions at the origin. Now we are again trying to make all the traces look like delta functions, but with the spike located at what is currently the maximum-valued point on each trace. Once the shifts are found, again we take the average shift, call it K_0^S, and apply it to each of the traces. This method is iterative; as we shift each frequency component, the maximum point of each trace will probably move, thus affecting how much shifting will be applied to the next frequency component. In the deconvolution test cases that I have tried, it has been necessary to run the traces only once through the whole range of frequencies.

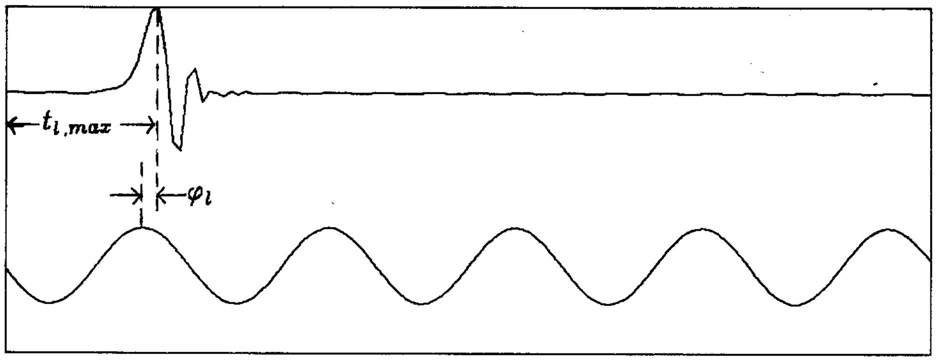


FIG. 4. Finding the phase shift on a trace. In this figure, the lower trace represents a single-frequency component of the upper trace. We want to see how much we need to shift the phase at this frequency in order to make the upper trace spikier. To find the desired phase shift at this frequency, we measure the distance between the maximum of the original trace ($t_{l,max}$) and a peak on the single-frequency trace. This will allow us to find φ_l.

In the case where K_1^S is a problem, one more refinement must be added to the method. We simply averaged the phase shifts in order to find K_0^S (and I should note that this averaging was performed using the "center of mass of points on a circle" method), but now we want to find an additional phase shift that varies linearly with shot position. The problem, of course, is that the measured phase shifts all fall in the range $-\pi < \Delta\varphi \leq \pi$, so that finding a linearly changing shift is difficult if the shift per shotpoint is too great. The answer to this problem, fortunately, is simple. The question is, given a series of phase shifts φ_l, how do you find a linear shifting term K such that $\varphi_l \approx$ wrap(lK) for all values of l? The difficulty is caused by the presence of the "wrap" function, so we multiply by $i = \sqrt{-1}$ and take the exponential of both sides. Then the problem becomes, find K such that exp($i\varphi_l$) $\approx$ exp(iKl). Now the "wrap" function has gone away, and we have a problem that can clearly be solved by performing a Fourier transform over l. Remembering that l is a term that varies with position, we now have an algorithm for finding K: take each φ_l, multiply it by i and exponentiate it, take the spatial Fourier transform over the series of exponentials of φ_l, and, in the spatial frequency domain, find the position of maximum value. The spatial frequency represented by this maximum position is simply the K that we have been looking for.

Test Data

Now that we have some idea about how to go looking for frequency-dependent surface-consistent statics, let's take a look at how these methods work on synthetic data. Figure (6) shows the time-domain wavelets S_i, G_j, Y_{i+j}, and H_{i-j} that were used to generate the synthetic data from the formula $D_{ij}(t) = S_i(t) * G_j(t) * Y_{i+j}(t) * H_{i-j}(t)$. Notice that the wavelets represent delta functions with varying amounts of static shift applied. The resulting synthetic data was put through a Gauss-Seidel iteration, using as the order of iteration S, G, H, Y. This gave the results in Figure (7) after 1 iteration, and gave the results in Figure (8) after 16. Then the S_i panel from Figure (8) was put through the null-space suppression procedure, and Figure (9) was the result of five iterations of that. Figures (10-12) are the analogous results obtained using the iteration order Y, S, G, H.

Figure (12) deserves some study, because it points out a problem in my null-space suppression method. Compare Figures (9) and (12), and note how much sharper the wavelets in Figure (9) are. The problem in Figure (12) stems from the fact that my suppression method does not seem to work very well at higher frequencies. The S_i wavelets in Figure (8) already had their high-frequency components in about the right spot. The wavelets in Figure (11), on the other hand, did not, and my process did not do a very good job of correcting this problem. I am not really surprised, because as far as I know, there is no theoretical explanation of why my method should work at all, and thus I can make no predictions of when it should break down.

Conclusions

The generalized frequency-dependent surface-consistent statics problem can be solved, although at some cost in computer time. At this point, the question is not one of developing the method, but one of refining the method so that it works better and so that it corresponds better with the problems encountered in real data. In the former category, the problem of suppressing the null-space components still needs to be dealt with using some approach that has a sound theoretical basis. In the latter category, the problem of residual moveout has not been dealt with at all. Probably equation (5) should be reformulated so that it includes this effect. Finally, it remains to be seen whether there are applications that can justify the large amounts of computer time that this method requires.

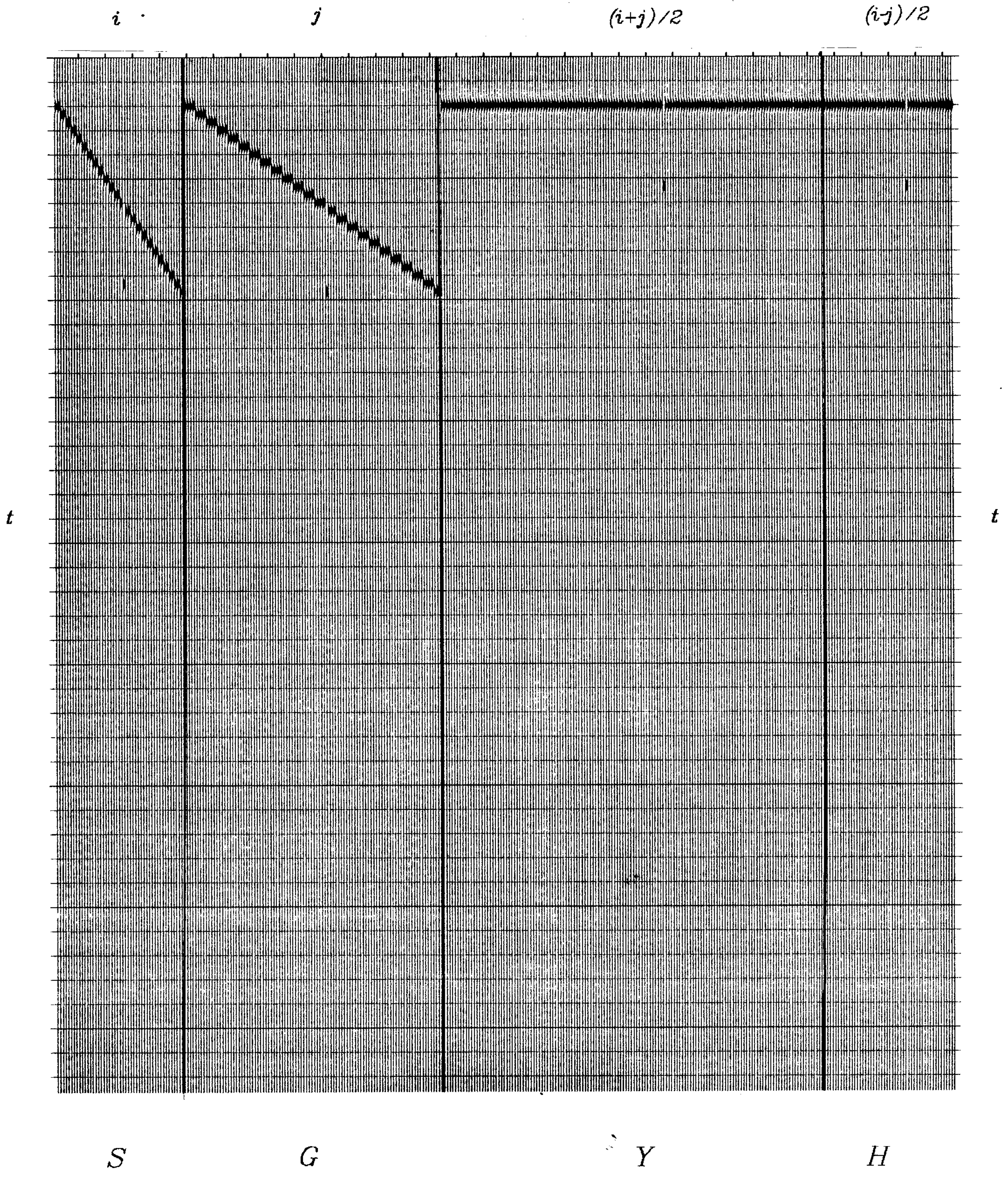


FIG. 6. These are the S, G, Y, and H components that were used to generate the synthetic data.

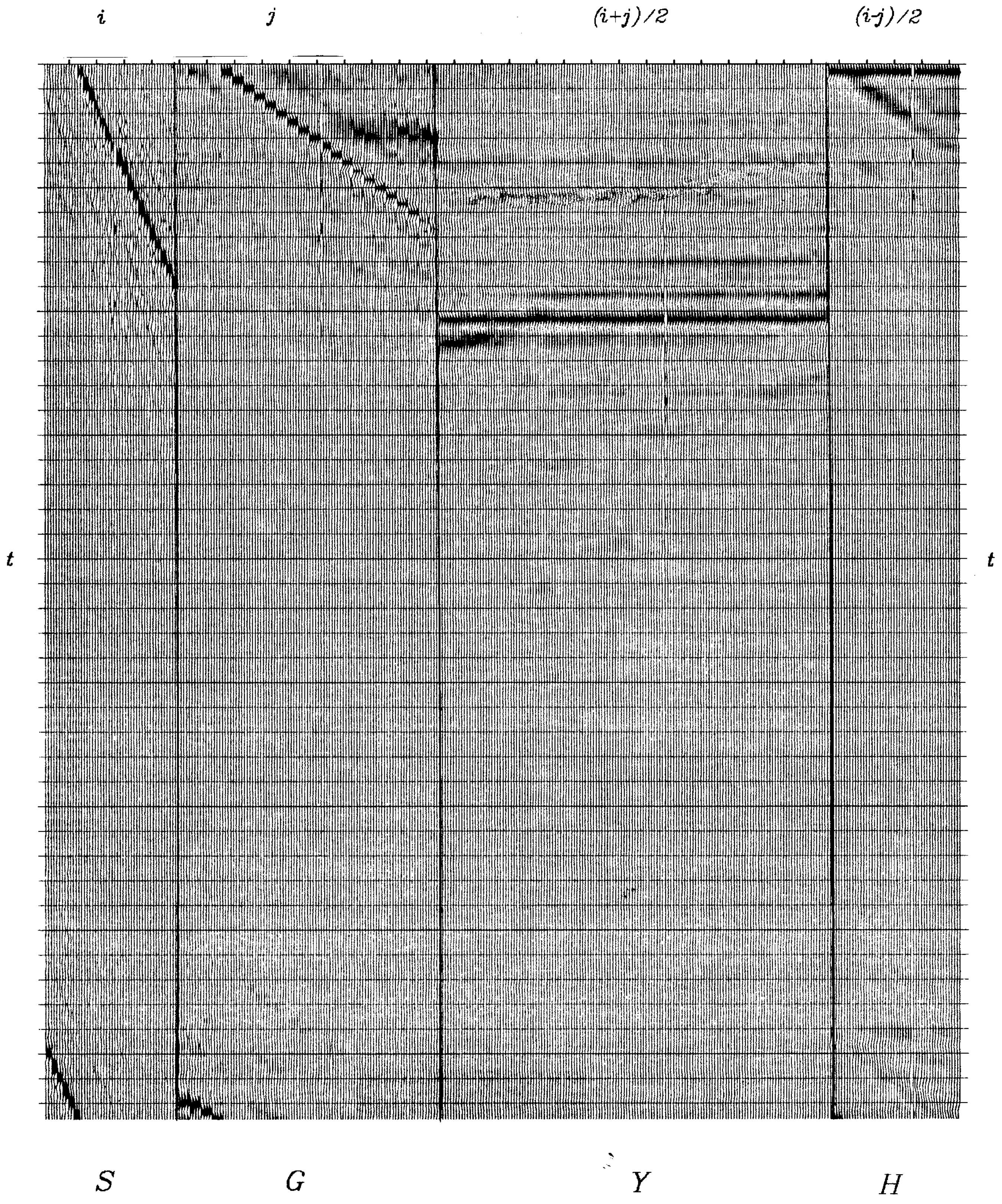


FIG. 7. These are the S, G, Y, and H components found by the the least-squares program after 1 iteration in the order S, G, H, Y. Note the wraparound effects on the time axis in this and the following figures.

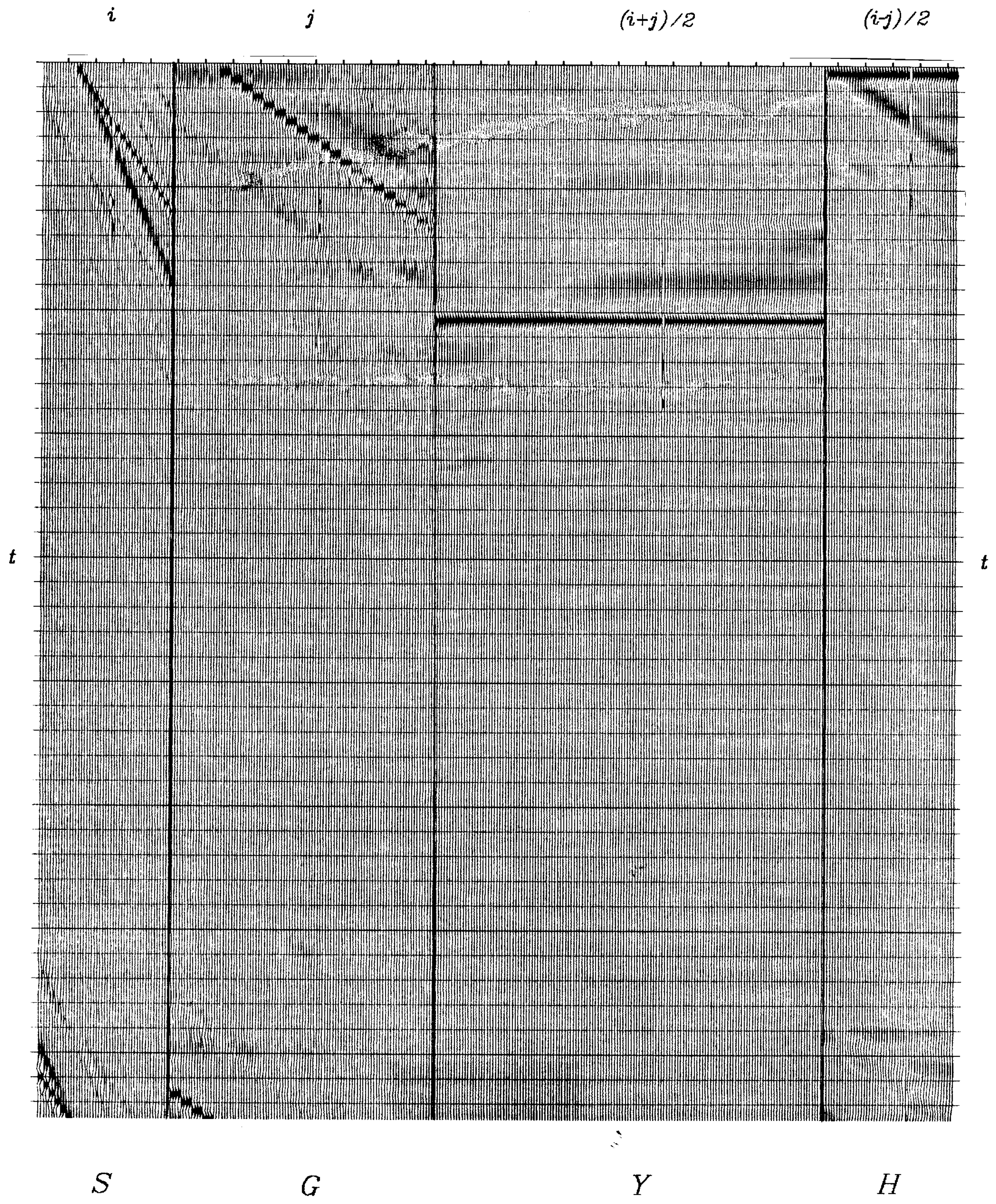


FIG. 8. These are the S, G, Y, and H components found by the the least-squares program after 16 iterations in the order S, G, H, Y.

....

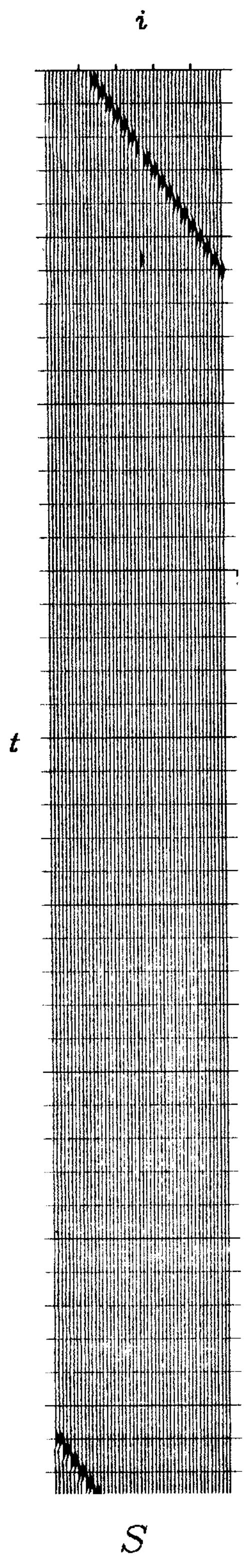


FIG. 9. This is the S component from the previous figure, after 5 iterations through the null-space suppression program.

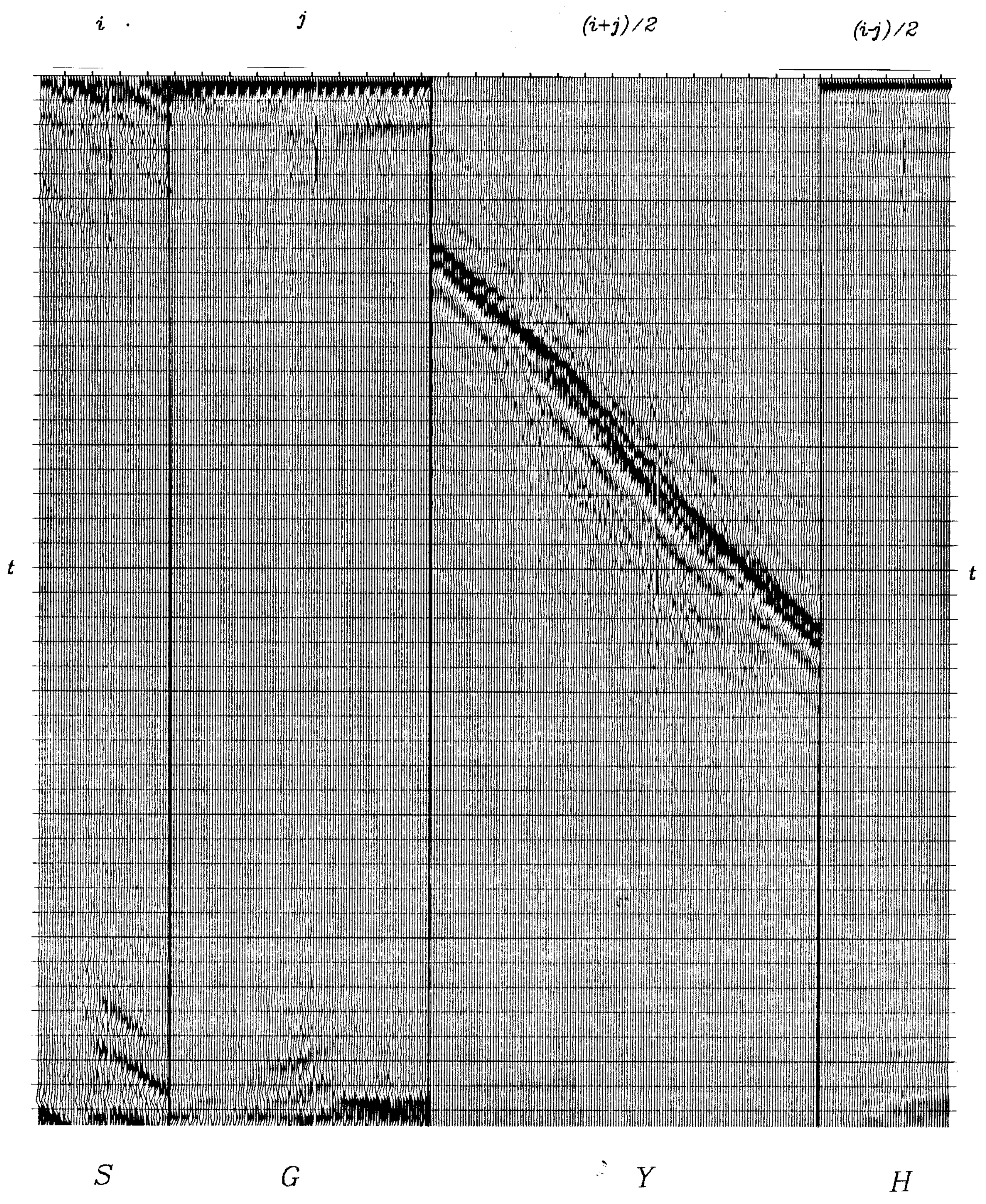


FIG. 10. These are the S, G, Y, and H components found by the the least-squares program after 1 iteration in the order Y, S, G, H.

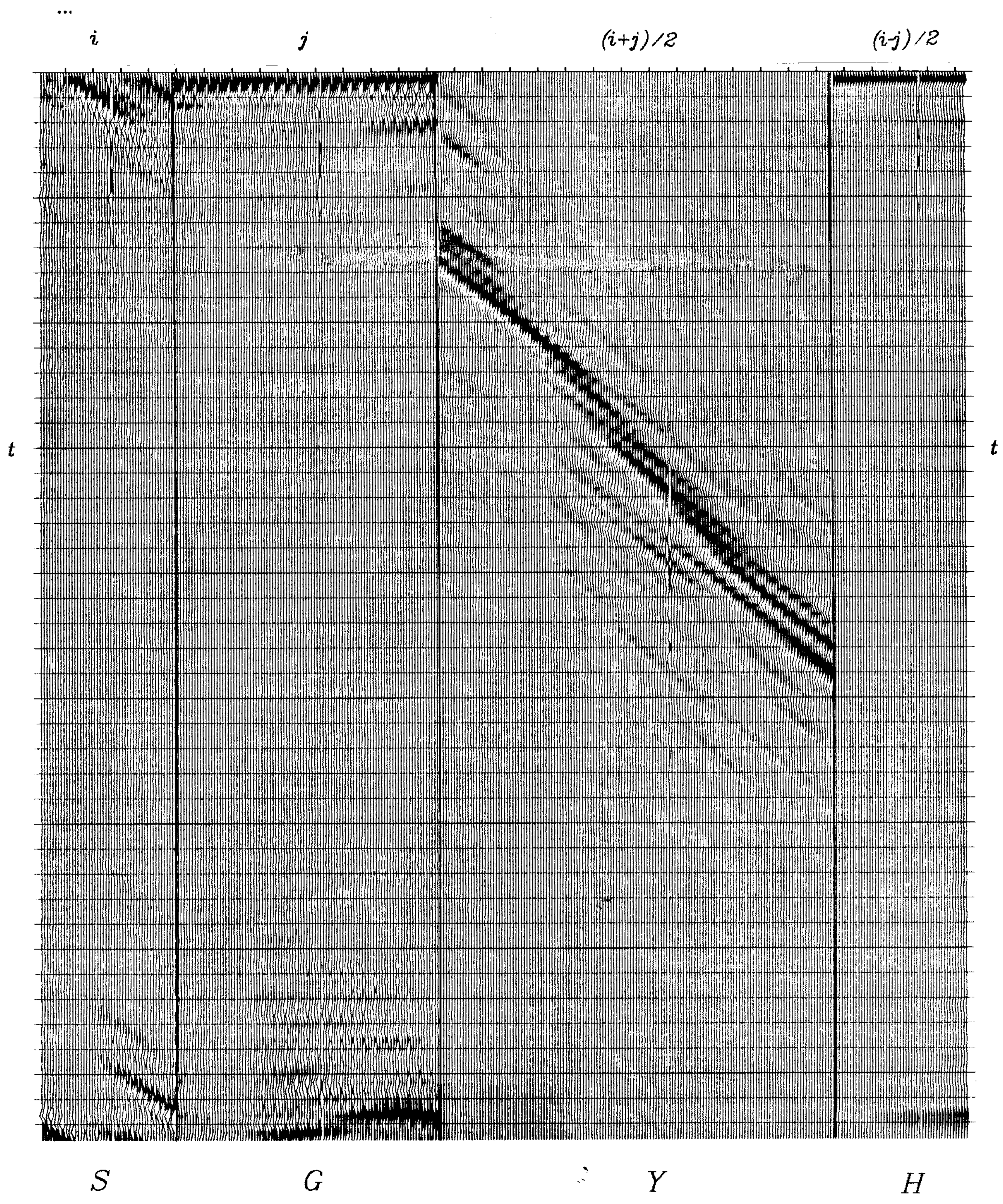


FIG. 11. These are the S, G, Y, and H components found by the the least-squares program after 16 iterations in the order Y, S, G, H.

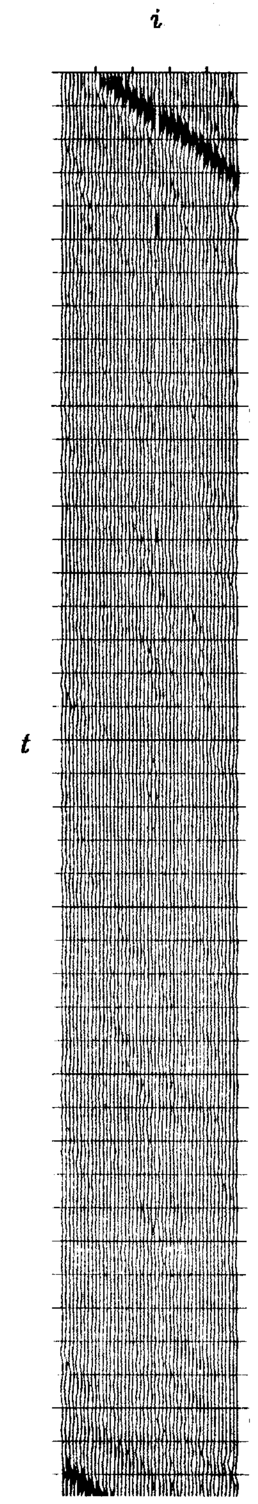


FIG. 12. This is the S component from the previous figure, after 5 iterations through the null-space suppression program. Notice that these events are not as sharp as those in Figure (9).

ACKNOWLEDGMENTS

I owe a great deal of thanks to Jeff Thorson. Without some of his valuable suggestions this project would have died an early death. And of course, I would like to thank Jon Claerbout and the sponsors of the Stanford Exploration Project.

REFERENCES

Mitrofanov, G.M., 1979, Use of smoothing windows in spectral analysis of seismic traces: Soviet Geology and Geophysics, v. 20, no. 1, p. 89-101 (in English).

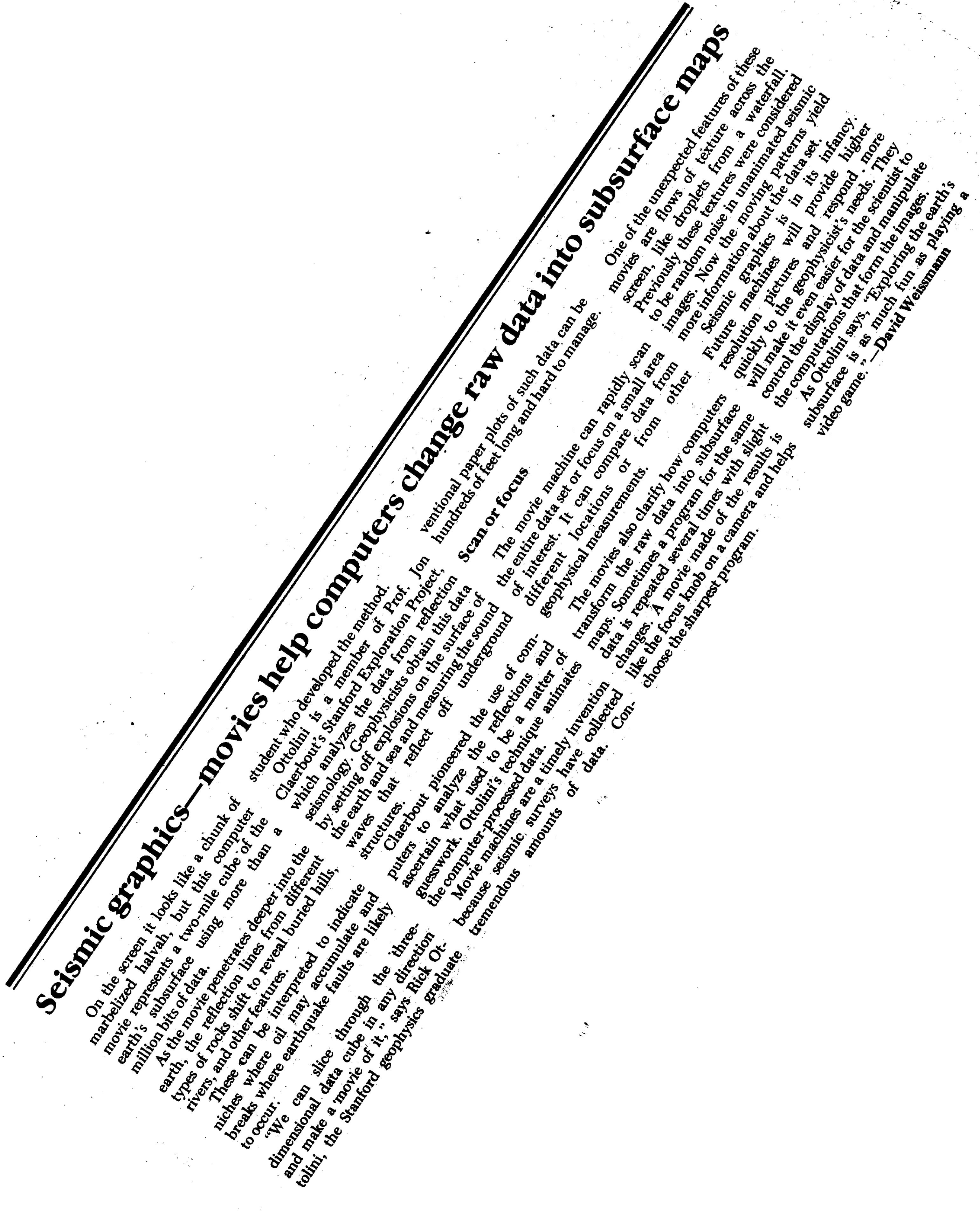
Mitrofanov, G.M., A. Sysoev, and G. Yashkov, 1982, Issledovanie vozmozhnostej spektral'no-statisticheskogo metoda (SSM) pri obrabotke i interpretatsii sejsmorazvedochnykh dannykh: in the book Sbornik dokladov vtorogo nauchnogo seminara stranchlenov SEV po neftjanoj geofizike, Moscow, v. 1, p. 41-49.

Morley, L.C., 1982, Predictive techniques for marine multiple suppression: Stanford University, Ph.D. thesis (also published as SEP-29).

Taner, M.T. and F. Koehler, 1981, Surface consistent corrections: Geophysics, v. 46, p. 17-22.

Taner, M.T., F. Koehler, and K.A. Alhilali, 1974, Estimation and correction of near-surface time anomalies: Geophysics, v. 39, p. 441-463.

Wiggins, R.A., K.L. Larner, and R.D. Wisecup, 1976, Residual statics analysis as a general linear inverse problem: Geophysics, v. 41, p. 922-938.



Seismic graphics—movies help computers change raw data into subsurface maps

On the screen it looks like a chunk of marbelized halvah, but this computer movie represents a two-mile cube of the earth's subsurface using more than a million bits of data.

As the movie penetrates deeper into the earth, the reflection lines from different types of rocks shift to reveal buried hills, rivers, and other features.

These can be interpreted to indicate niches where oil may accumulate and breaks where earthquake faults are likely to occur.

"We can slice through the three-dimensional data cube in any direction and make a movie of it," says Rick Ottolini, the Stanford geophysics graduate student who developed the method.

Ottolini is a member of Prof. Jon Claerbout's Stanford Exploration Project, which analyzes the data from reflection seismology. Geophysicists obtain this data by setting off explosions on the surface of the earth and sea and measuring the sound waves that reflect off underground structures.

Claerbout pioneered the use of computers to analyze the reflections and ascertain what used to be a matter of guesswork. Ottolini's technique animates the computer-processed data.

Movie machines are a timely invention because seismic surveys have collected tremendous amounts of data. Conventional paper plots of such data can be hundreds of feet long and hard to manage.

Scan or focus

The movie machine can rapidly scan the entire data set or focus on a small area of interest. It can compare data from different locations or from other geophysical measurements.

The movies also clarify how computers transform the raw data into subsurface maps. Sometimes a program for the same data is repeated several times with slight changes. A movie made of the results is like the focus knob on a camera and helps choose the sharpest program.

One of the unexpected features of these movies are flows of texture across the screen, like droplets from a waterfall. Previously these textures were considered to be random noise in unanimated seismic images. Now the moving patterns yield more information about the data set.

Seismic graphics is in its infancy. Future machines will provide higher resolution pictures and respond . more quickly to the geophysicist's needs. They will make it even easier for the scientist to control the display of data and manipulate the computations that form the images.

As Ottolini says, "Exploring the earth's subsurface is as much fun as playing a video game."—**David Weissmann**