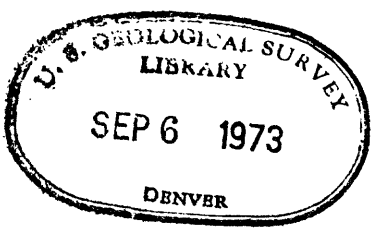
UNITED STATES DEPARTMENT OF THE INTERIOR

GEOLOGICAL SURVEY

COMPUTER PROGRAMS FOR TIME AND SPACE ADAPTIVE DECONVOLUTION OF SINGLE CHANNEL MARINE SEISMIC REFLECTION DATA

by

Don C. Riley



Open-file report

1973

73-235

NOTE: THIS REPORT IS THE BEST COPY AVAILABLE

TABLE OF CONTENTS

ILLUSTRATIONS

FORWARD

This is a cookbook. I have tried to put together an adequate description of the processing package including recipes for its usage. This is not intended to be an essay of any sort on digital seismic signal processing. It is meant to get the user off the ground and flying with a minimal expenditure of time and trouble. The program has been designed to be transparent to the user, the only communication being through a keyword parameter list. Every parameter has a corresponding default value so that if the parameter is not set by the user the program automatically supplies it. Under such a system it is possible to process an entire data tape with the user only supplying his name.

Perhaps the most important lesson that one soon learns from processing data of any sort is that of quality. There is no substitute for the highest possible data recording quality in the field; to ignore this is false economy. Subsequent digital processing cannot be expected to extract seismic signals from garbage masquerading as data. Careful recording of a properly conditioned input seismic signal (i.e., adequate dynamic range, good gain ranging and proper prefiltering) will result in data that are suitable for digital processing.

It was necessary to write several different programs that essentially do the same thing due to the fact that recorded data were in several different formats. In addition to the resulting confusion, the processing time for the two incompatible format tapes is significantly longer. It is highly recommended that any future digital acquisition be done in the SEG interchange format. SEG interchange format is basically 16 bit, two's compliment, 9 track data. Processing package C is designed for such data. Not only is this format more or less a standard, but the data can be directly read into most machines without any intervening unpacking routine.

Most of the processing could be easily implemented on a modest-sized "minicomputer" system. In fact, some of the processing could be done in real time as the data were being recorded, although it should still be recorded for possible reprocessing offline. All of the programs have been fully tested and debugged on the Stanford Computation Center IBM 360 model 67. Many of the references to specific implementation are oriented toward this computer system, however, the seismic processing programs may be readily adapted to most similar sized machines.

The author would like to greatly acknowledge the constant encouragement and technical guidance given by Professor Jon F. Claerbout and John P. Burg of Stanford throughout this study.

INTRODUCTION

Single channel seismic reflection profiling has become an economical and efficient method of gathering large amounts of data at sea. This geophysical tool has been used extensively by the U.S. Geological Survey in realizing their objective of reconnaissance surveying of the continental shelves of the United States. Profiling in shallow to moderate water depths encounters one of the classic problems in exploration seismology; that of short-period sea floor multiples. As the sea floor and water/air interface provide an excellent energy trap, the records frequently are plagued with intense and persistent reverberations masking a large part of the desired signal. Consequently, many of the profiles recorded present a difficult task for geologic interpretation largely due to the multiple problem.

As we move deeper in time in the record much of the reverberant energy has attenuated, but now we encounter the problem that deep reflected events are buried in the ambient noise field, for although the sparker and/or airgun provides us with a highly repetitive source it does not provide an overly strong seismic signal. In deep water, the sea floor multiples separate out and are usually clearly recognizable on the records by their strength, time of arrival, and increasing dip. Although easy to identify, at present there are no reliable techniques for their elimination to sub-visible. This is the current subject of investigation and will be treated in a subsequent report.

If the data are available in digital form, then a wide variety of statistical and mathematical techniques may be brought to bear on the above cited problems. Digital methods have been widely and successfully used in the exploration industry in the past few years. This fact, coupled with the extremely wide dynamic range available only with digital recording, demonstrates the need for a digital acquisition system for future surveys. The single, most effective enhancement tool for single channel data is predictive deconvolution. This has proved to be highly useful in attenuating statistically stationary reverberating data. By stationary, we mean that the statistics of the time series or seismogram does not change with time. A wide-sense stationary assumption strictly means that the mean and autocorrelation functions are constant with time. However, most seismic data, expecially those collected on the continental shelves, are generally non-stationary in both space and time. The conventional approach[1/] is somewhat overextended in these situations since it is based on stationary assumptions. The technique of predictive deconvolution presented in this report represents a radical departure from the usual methods in that the prediction error operator is optimal at every point in time and space. This time and space adaptive deconvolution technique forms the basis of the processing system to be described.

1/ See K. L. Peacock and S. Treitel, 1969, Predictive deconvolution: theory and practice: Geophysics, v. 34, p. 155, for details on what may be termed the conventional approach to deconvolution.

OBJECTIVES

The principal objective is to enhance the recorded seismic reflection data to achieve structural and stratigraphic mapping capability in the presence of multiple reflections and ambient seismic noise. Due to the large amount of data recorded on a typical survey, a necessary constraint is that any such processing schemes be economical to apply (on the order of $3.00 to $4.00 per line mile). Further, it is desirable to make the processes as automatic as possible, requiring minimal training and time in setting up the parameters necessary to process a particular profile. Finally, with an eye toward possible future acquisitions, the programs developed should possess real time speed and capability necessary for shipboard processing with a small computer system.

The software system that has been developed meets the cost and simplicity constraints while producing good enhancement of the seismograms. However, the current programs require about 200 to 255K bytes of storage. This was, for the presently used computer, necessary to achieve the desired low cost since at every opportunity core storage was exchanged for higher execution speed. By programming carefully and exchanging speed back for core the algorithms should be easily implemented on a mini-computer.

THEORETICAL CONSIDERATIONS

In this section we will first review what it is that seismic deconvolution is theoretically designed to do. The initial earth model will be necessarily idealized. Then we will add real-life complications and see where the conventional methods come apart. Finally, a more general approach will be presented in a hopefully self-justifying manner.

We start by assuming the basic geometry of reflection seismology, that is plane layers and plane waves. When we set off a surface disturbance we initiate a down-going wave in the first layer. From here on out the reflection and transmission coefficients of the various layers take over and produce a seismogram recorded at the surface. What we record is an upgoing wave. But what we get is not what we want. What we wanted to learn was the spatial distribution of the reflectors, we wanted the straight through waves off of each interface. What we got was that plus a lot of extra energy returning to the surface in the form of multiple or interbed reflections.

We now wish to take the seismograms recorded at the surface and decompose them into an ideal record of just the straight through reflected events. This is the job which so-called seismic deconvolution or decomposition intends to do. We first need to set up some criteria for separating the true geologic reflected events from the reverberations. The reflection series in the earth will be assumed to be randomly distributed, that is, we expect arrivals from them to occur at unpredictable times. The multiple reflections or reverberations are completely determined from this random series. Knowing the random distribution of reflection coefficients we could theoretically predict all of the multiples observed at the surface. Thus, we are led to partition the energy on the seismogram into that which is predictable and that which is unpredictable. If we can remove all of the predictable energy, then we are left with the unpredictable part of the seismogram which, in the framework of our simple model, is the subsurface reflection series.

The conventional way of attempting to do this is to design a filter from a portion of the data that, when convolved with the data, yields the best predicted estimate, in a least squares sense. Then substract this predicted estimate from the original data. This operation is call prediction-error filtering, since the output series is the error or difference between the data and its predicted estimate. In the least squares design procedure, the solution is found such that the sum of the squared errors is a minimum. But since the output of our operator is the error we actually are minimizing the mean output power of our filter. Minimizing the mean output power does the correct thing in that it minimizes the error in the prediction equations. This is reasonable if we consider the infinite number of possible paths through our layered medium as possible seismograms. That seismogram which contains only the straight-through arrivals also contains the minimum amount of energy of all the possibilities.

Now it is obvious that no such plane layered medium exists in the real world (if there were such structure there would be really no need for seismic exploration). However, it is still reasonable to assume the minimum output-power criteria as a means of reducing the masking effect of short-period reverberations. In addition, the model fails to include the non-stationary conditions (in time) resulting from dispersion, attenuation, etc. As we shall see, by considering predictive deconvolution within a data-adaptive framework, the stationarity assumption may be relaxed. Optimum processing of highly non-stationary data is possible since the filtering is able to respond to the environment in which it is working.

In the design of digital deconvolution filters we are interested in making the filtering procedure optimal throughout the data that we are intending to process. We observe that seismic reflection data generally exhibit a limited degree of stationarity in both time and space. The early part of the seismograms may be dominated by prominent near-surface effects decaying into an ambient noise field at the end of the record. Spatial non-stationarity is often introduced by lateral variations in the reflective properties of the seafloor or near-surface layers. Clearly, an ideal deconvolution scheme then, needs to be able to rapidly adapt in both time and space.

There are several ways in which we may compute time-variable filters in order to accommodate non-stationarity (see fig. 1). We could select several time gates representing distinct noise types and compute the necessary autocorrelation lags for each, with possible spatial averaging. Then solve for the prediction error filters using Levinson recursion and apply the several filters to the appropriate data. Another approach would be to use the Burg technique within the selected time gates to directly compute the prediction error filters and apply the filters to the suitable data. This second method allows shorter time gates than the first since it avoids the "end-effects" problems involved in estimating an autocorrelation function from a short data sample. The third method, which is presented in this paper, is to use a continuously adaptive system that is derived from the Burg technique. We sequentially process the entire length of data continuously updating the filter at every time point or as often as is statistically reasonable. The system is self-optimizing in the face of changing environments and it is for this reason that it should be useful in dealing with non-stationarity.

We will begin by noting some attributes of adaptive systems and prediction error operators. An adaptive system is characterized by two properties: the system is adjustable, and the system adjusts itself in order to meet some performance criteria. In this paper we shall define a prediction error operator to be a physically realizable filter whose first filter coefficient is unity. The optimum prediction error filter is the one which has minimum output power. Now, we wish to build an adaptive realization of the optimum prediction error operator. To do this we need to 1) make the first filter weight of the system be unity, and 2) continuously adjust the parameters of the system to minimize the average squared-output.

DECONVOLUTION IN A NON-STATIONARY ENVIRONMENT

SOME APPROACHES ARE

1. COMPUTE AUTOCORRELATION LAGS WITHIN SEVERAL PRE-SELECTED TIME GATES, COMPUTE PREDICTION ERROR FILTER FOR EACH USING LEVINSON RECURSION, APPLY FILTERS TO THE APPROPRIATE DATA.
2. USE THE BURG TECHNIQUE DIRECTLY COMPUTING THE PREDICTION ERROR FILTER FOR EACH OF THE SEVERAL TIME GATES, APPLY FILTERS TO THE CORRESPONDING DATA.

   USE AN ADAPTIVE FORM OF THE BURG TECHNIQUE CONTINUOUSLY FILTERING THE DATA, UPDATE THE FILTER AT EVERY TIME-POINT BASED ON STATISTICS FROM A MOVING TIME GATE.

Figure 1

ADAPTIVE FILTER LADDER

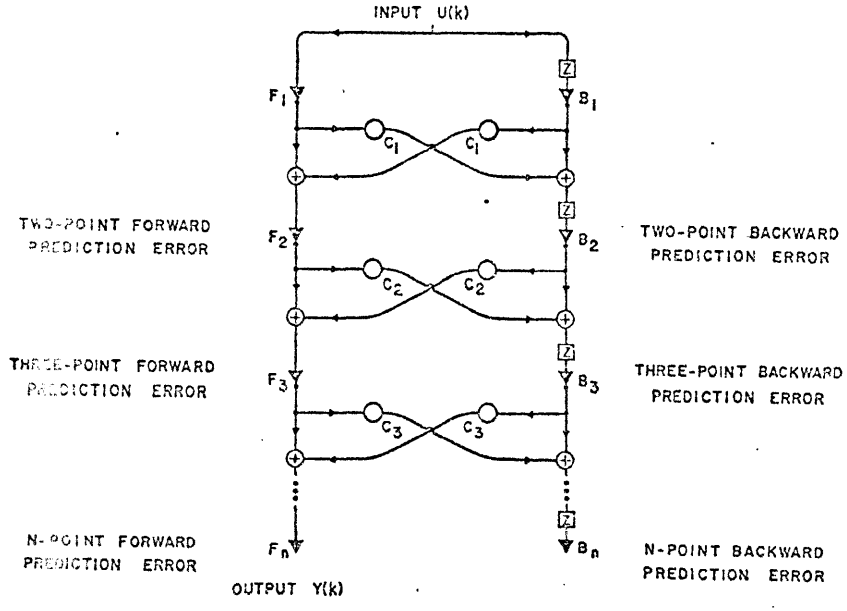


Figure 3

RECURSIVE SOLUTION OF PREDICTION ERROR FILTER (P.E.F.)

$$\text{IF} \quad \begin{bmatrix} \Phi(0) & \Phi(1) \\ \Phi(1) & \Phi(0) \end{bmatrix} \cdot \begin{Bmatrix} 1 \\ C_1 \end{Bmatrix} = \begin{Bmatrix} P_2 \\ 0 \end{Bmatrix} \quad \text{, THEN}$$

$$\begin{bmatrix} \Phi(0) & \Phi(1) & \Phi(2) \\ \Phi(1) & \Phi(0) & \Phi(1) \\ \Phi(2) & \Phi(1) & \Phi(0) \end{bmatrix} \cdot \left\{ \begin{bmatrix} 1 \\ C_1 \\ 0 \end{bmatrix} + C_2 \begin{bmatrix} 0 \\ C_1 \\ 1 \end{bmatrix} \right\} = \left\{ \begin{bmatrix} P_2 \\ 0 \\ \Delta_2 \end{bmatrix} + C_2 \begin{bmatrix} \Delta_2 \\ 0 \\ P_2 \end{bmatrix} \right\} = \begin{Bmatrix} P_3 \\ 0 \\ 0 \end{Bmatrix}$$

TWO-POINT FORWARD BACKWARD P.E.F.

WHERE $\Delta_2 = \Phi(2) + \Phi(1)C_1$, $C_2 = -\Delta_2/P_2$ AND $P_3 = P_2(1-C_2^2)$

THUS THE THREE-POINT P.E.F. IS OBTAINED FROM THE KNOWN TWO-POINT FILTERS BY ADJUSTING THE SINGLE PARAMETER C_2

Figure 2

PROPERTIES OF THE ADAPTIVE FILTER LADDER

1. PARAMETERS , C_j , ARE ADJUSTED TO MINIMIZE THE AVERAGE OUTPUT POWER.
2. AND THE FIRST FILTER WEIGHT IS UNITY, THUS BY DEFINITION IT IS A PREDICTION ERROR OPERATOR.
3. FOR $|C_j|<1$, THE FILTER IS MINIMUM-PHASE AND IS STABLE EVEN AT VERY FAST ADAPT RATES.
4. PARAMETERS OF THE SYSTEM CORRESPOND TO REFLECTION COEFFICIENTS OF AN IDEAL LAYERED MEDIA MODEL.

Figure 4

There are an infinite number of possible adaptive schemes which will do the job, that is, perform as prediction error operators. However, we may expect some to work better than others. In particular, there are those which become unstable at fast adaption rates, notably the steepest descent or gradient search algorithms. We would like to find that realization which performs best out of all the possibilities. One way to start is to consider the situation when the inputs to the system are stationary. Then, if we find the system that produces the minimum average power output for this case, it should also be a good candidate for the non-stationary problem.

However, we already know how to design the best or optimum prediction error operator for a stationary time series. In addition, we also have a highly useful recursive scheme for obtaining the optimum filter, as figure 2 illustrates. Can we design an adaptive realization that performs Levinson recursion? To answer this requires that we think carefully about what is going on in building up an N+1-point filter from an N-point prediction error filter. We note that the form of the new filter is simply the combination of the old N-point filter and its time-reverse slid back one point. The parameter, C_N, is the only new information we need and it is determined in order to satisfy the N+1-point filter equations. The N+1-point equations require that C_N be such that the N+1-point filter have minimum power output. Equivalent statements can be made in terms of the filtered outputs. Here we will call the output of the N-point filter the N-point forward error series. Let's also process the data with the same filter turned-around and slid back one point and call this the N-point backward error series. Now we wish to go ahead and compute the next higher order output, that is, the N+1-point forward and backward error series. We saw that in the filter recursion we got the next higher order filter by combining the two N-point forward and backward filters through the parameter C_N. The correct procedure in terms of the filtered outputs is identical. The new N+1-point forward error series is obtained by combining the old N-point forward and backward error series through the parameter C_N. Likewise, the N+1-point backward error series is obtained by multiplying the N-point forward error series by C_N and adding to the N-point backward error series. We adjust C until the average squared-error or output power is minimized. Thus, we may conceive of an equivalent procedure that executes the recursion in terms of filtered outputs.

Figure 3 illustrates a direct realization of an adaptive system that performs Levinson recursion. Each stage or step in the ladder completes one iteration. The C's in the drawing are adjustable gains that multiply the signal going through it by strength C. The Z-boxes just delay the signal one time unit. We first build-up the two-point forward and backward filtered outputs by combining the one-point forward filtered output (which is just the input series) with the one-point backward series (which is the unit delayed input). The parameter, C_1, is adjusted to minimize the average output power just below the first pair of summers. The two-point forward prediction error operator is clearly (1 , C_1) and the backward filter is (C_1 , 1) delayed unity. The recursion proceeds down the steps of the ladder going to higher order filtered outputs. At each step the parameter, C_j, controls the amount of lower order errors fed forward to create the next

higher order forward and backward filtered outputs. We can continue on out to the Nth stage where we pick up our final N-point forward prediction error series which is the desired output.

We can see that the first coefficient in the forward prediction error filter is always unity by noting the straight-through path down the left leg of the ladder. All other paths include some delay. If the parameters of the system are adjusted to minimize the average squared-outputs of the forward and backward filters, the system will then be the optimum prediction error operator. Furthermore, since the particular structure we chose is, in fact, a realization of the Levinson method we also gain some of its desirable properties (see fig. 4). Namely, for $|C_j| < 1$, the filter is always stable and minimum-phase. Also, we know that the C's in the Levinson recursion correspond to the reflection coefficients in an ideal layered media model. Thus, the adjustable parameters of our adaptive system take on physical meaning as the reflection coefficient series in a hypothetical layered model. This may prove useful in designing realistic constraints for some filtering applications.

Now that we have constructed an adaptive system which is a prediction error operator having some nice properties, we ought to consider what it is we wish to do with it. We know that for stationary inputs the filtering is optimal when we correctly compute the reflection coefficients. Thus, we would set the parameters in the ladder to their optimal values as computed from a sample of the data. But as time goes on we may observe that the statistics of the data have changed. We should now recompute the filter using the new data. In fact, for an arbitrary non-stationary time series we should recompute the filter at every time point. Therefore, what we want to do with our adaptive system is adjust the parameters often; if necessary, as each new data point arrives. The adjustments will be made in such a way as to minimize the average output power at each time point. Thus, as the spectral content of the input changes the reflection coefficients of the system and hence the filter adapts. In effect, we are <u>continuously recomputing the prediction error filter while simultaneously processing the data</u>.

As figure 5 illustrates, we will assign some people to man the controls at each step in the ladder. Everyone has the job of minimizing the average squared-output of their stage by twiddling their reflection coefficient. If everyone does a good job, then the total average output power of the system as a whole will be minimized. What principle governs the way the people make their adjustments? The only information they have is how the system responded to their adjustments in the past. The best guess that they can make for the future is that the response will be similar. If they discover that things have changed, further adjustment is needed. The people learn and adapt to their new environment and gradually forget what happened in the distant past.

Now, we need to formulate an adjustment algorithm to replace the people working on the ladder. We will build-in the same principle that they used in learning and adapting. Figure 6 illustrates the procedure that minimizes the average power output of each stage. Since the forward and backward error

THE WORKER AT EACH STAGE TRIES TO
MINIMIZE THE AVERAGE OUTPUT POWER

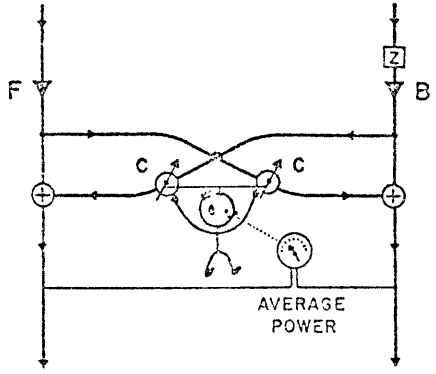


Figure 5

ESTIMATING REFLECTION COEFFICIENTS C_j

POWER OUT OF THE JTH STAGE IS THE SUM OF THE
SQUARED FORWARD AND BACKWARD PREDICTION ERRORS.

THE AVERAGE POWER IS

$$P_j = \frac{1}{2N}\sum_{k}^{N}\left[F_j(k) + C_j B_j(k)\right]^2 + \left[C_j F_j(k) + B_j(k)\right]^2$$

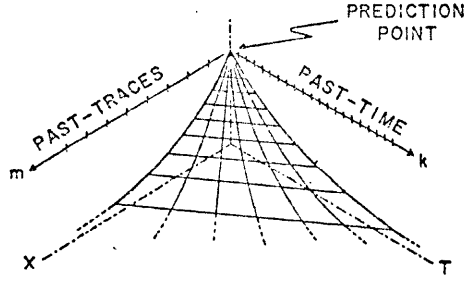
FOR WHICH A MINIMUM IS OBTAINED WHEN

$$C_j = \frac{-2\sum F_j(k) B_j(k)}{\sum F_j(k)^2 + B_j(k)^2} \qquad |C_j| < 1$$

Figure 6

TIME-SPACE WEIGHTING FUNCTION

TIME-SPACE AVERAGED C IS COMPUTED FROM THE
WEIGHTED AVERAGE CROSS-POWER AND AUTO-POWER OF
PAST VALUES OF THE FORWARD AND BACKWARD ERRORS



$$w(k,m) = \exp(-k\Delta T/T_T - m/T_X)$$

$T_T \triangleq$ RELAXATION TIME (sec.) ; $T_X \triangleq$ RELAXATION DISTANCE (traces)

Figure 7

ESTIMATING REFLECTION COEFFICIENTS OF COLORED NOISE

POWER SPECTRA OF LEFT AND
RIGHT HALF OF INPUT SERIES
GENERATED WITH KNOWN
COEFFICIENTS ($C_1 C_2 C_3 C_4$)

ESTIMATED

COEFFICIENTS

$\hat{C}_1\ \hat{C}_2\ \hat{C}_3\ \hat{C}_4$

POWER SPECTRA OF LEFT AND
RIGHT HALF OF FOUR-POINT
FORWARD PREDICTION ERROR
SERIES OUTPUT

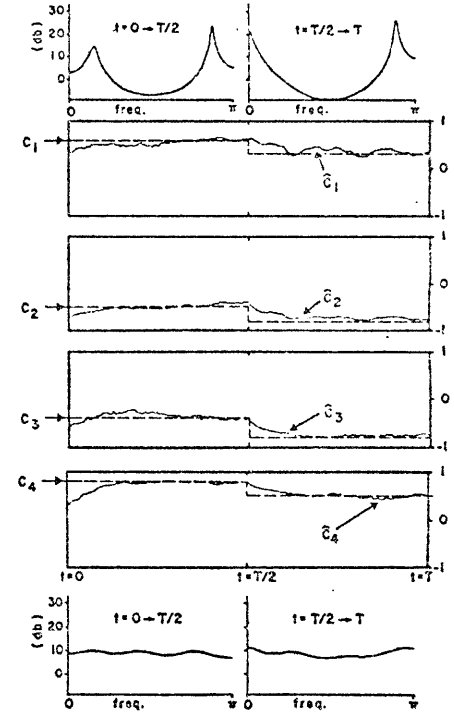


Figure 8

time series have the same average power output, we will average the power observed on the left branch with that of the right. This forward and backward averaged power output of the jth stage is $\bar{P}_j$. Varying only C_j we find that value which minimizes $\bar{P}_j$. The numerator of the optimum value of C_j is just the negative average cross-power and the denominator is the averaged auto-powers of the j-point forward and backward prediction errors. The summation will be taken over past values in time and space of the numerators and denominators. In effect, we are training the system to respond based on what happened in the past. Since we may expect changes to occur, the statistics close to the point we are trying to predict will be weighted more heavily than those farther away. Then as the predictable portion of the input time series changes the change will be quickly noted and incorporated into the adjustments.

Figure 7 illustrates one possible way of biasing the statistics in the summation to accomplish this. Here we compute the reflection coefficients from a weighted average of the cross-power (numerator) and the auto-power (denominator) of past values of the forward and backward prediction errors. Thus, the close-in statistics used to compute the filter are more heavily weighted against those farther away which are gradually "forgotten". The particular time-space weighting function shown in the slide is the product of a time exponential and a space exponential. The C's using this form are conveniently and economically computed from the current and immediate past values of the numerators and denominators by simple feedback. As the filtering proceeds, the time-space window moves along with the filter. The data are continuously processed and the filter is continuously updated. The result is that the prediction error operator adapts in time and space to continually minimize the indicated weighted average power output. The 'relaxation time' and 'relaxation distance' of the adaption are defined as the 1/e decay time and distance along the time and space coordinates axes.

In order to be responsive to rapid changes in the data, the system should be able to adapt rapidly. This means that relatively fast relaxation of past statistics is desirable. However, it is also necessary that the reflection coefficients be reliably estimated to insure accurate filtering. Thus, there must exist a trade-off between speedy adaption and reliable estimation of the filter parameters. It turns out, though, that we may obtain good estimates from a relatively small amount of data.

Figure 8 illustrates the performance of the adaptive filter ladder when it encounters a statistical step discontinuity in the input data. Synthetic time series were generated with four known reflection coefficients. The first half of the series was produced with different values for the C's than the second half. The measured spectrum of the first half of the series is shown in the upper center portion of the figure, and that of the second half is in the upper right. The true values, C_j, of the coefficients are shown by the dashed lines in the next four frames, which are to be compared with the estimated values, $\hat{C}_j$, which are the dotted lines. As the prediction error filter moves from left to right we see that the estimated values soon track the true values. Then mid-way into the data the true reflection coefficients abruptly change. The adaptive filter soon adjusts itself to

the new environment and the estimated coefficients begin to move in the direction of the true values. Note that the exponential behavior of the estimated coefficients following the step generally obeys the decay rate corresponding to the relaxation time set for the adaption. One should also note that for small reflection coefficients, as in $C_1 = 0.3$, the variance of the estimate is greater than for larger coefficients. This is reasonable since the small amount of correlation introduced due to a weak reflector is more difficult to detect than high correlation due to a strong reflector. The bottom frame shows the measured spectra of the resulting prediction error series. As we expect, the input series has been reasonably 'whitened' and the reverberating peaks removed.

The initial trial of this method on field data was with single fold airgun and sparker profiles recorded in the Chukchi Sea. Figure 9 is the original profile that has been bandpass filtered. We note a large amount of short-period reverberation energy is present, especially in the first 0.5 sec. Below this the character of the noise is more variable. It is here that we may expect our continually adaptive method of deconvolution to be useful. Figure 10 is the same data, but prior to the bandpass filter the adaptive filter has been applied. The flanks of the folded structure have become more clearly defined both within the patch of seafloor multiples at the top and in the more complex pattern below. At about 1.2 sec. they are also indications of a near-flat lying reflector possibly representing a detachment surface relating to thrusting of the overlying structure. These results were obtained using only nine coefficients in the adaptive ladder.

The primary reason that the deconvolved data are not as easy to interpret as we would like is because the filter has attempted to whiten the entire spectrum. A common practice used to improve resolution is to predict greater than unit distance. However, the filter operator used in such a procedure is clearly not minimum-phase since the energy of the filter is not maximized toward the front of the filter. We might consider that rather than gap the first N filter coefficients, the N leading reflection coefficients be held to zero. The result is that we may predict greater than unit span and the filter is always minimum-phase.

In this paper we have seen how to construct a simple adaptive system to do predictive deconvolution. The advantages of this method are as follows: 1) the filtering is continuous. The parameters of the system are continuously updated while the data are simultaneously processed. 2) Since the reflection coefficients estimated are always less than unity the system is always stable regardless of the adaption rate. The filter is always minimum-phase. 3) The parameters of the system that we adjust directly relate to the physical problem we are trying to solve. We can introduce realistic a priori constraints for the deconvolution process. 4) The system has real-time capability. The stages in the adaptive ladder could easily be built as discrete hardware elements and cascaded to form any desired length operator. The method is also suitable for implementation on a mini-computer due to its modest core requirements. 5) All lower order filtered outputs are at all times available as well as the Nth order forward

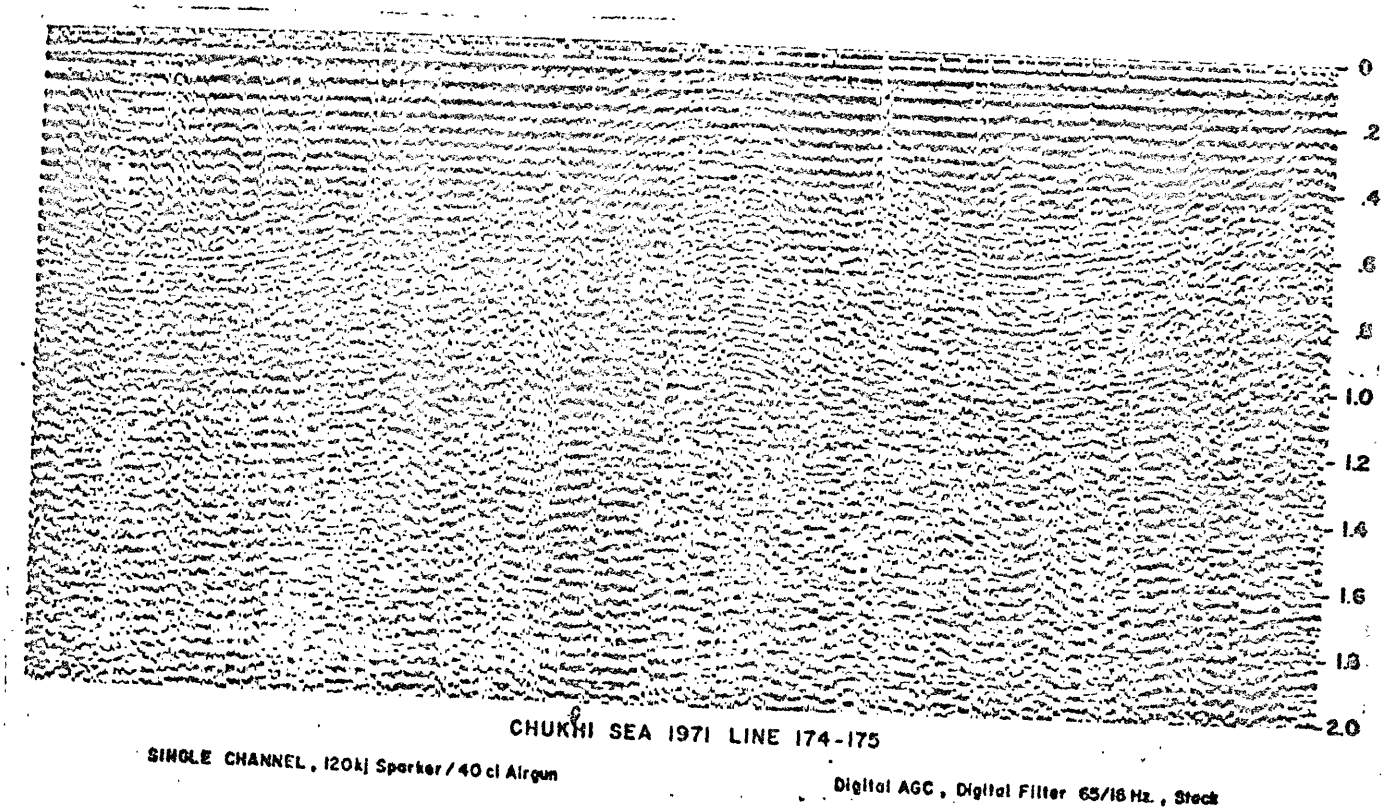


Figure 9

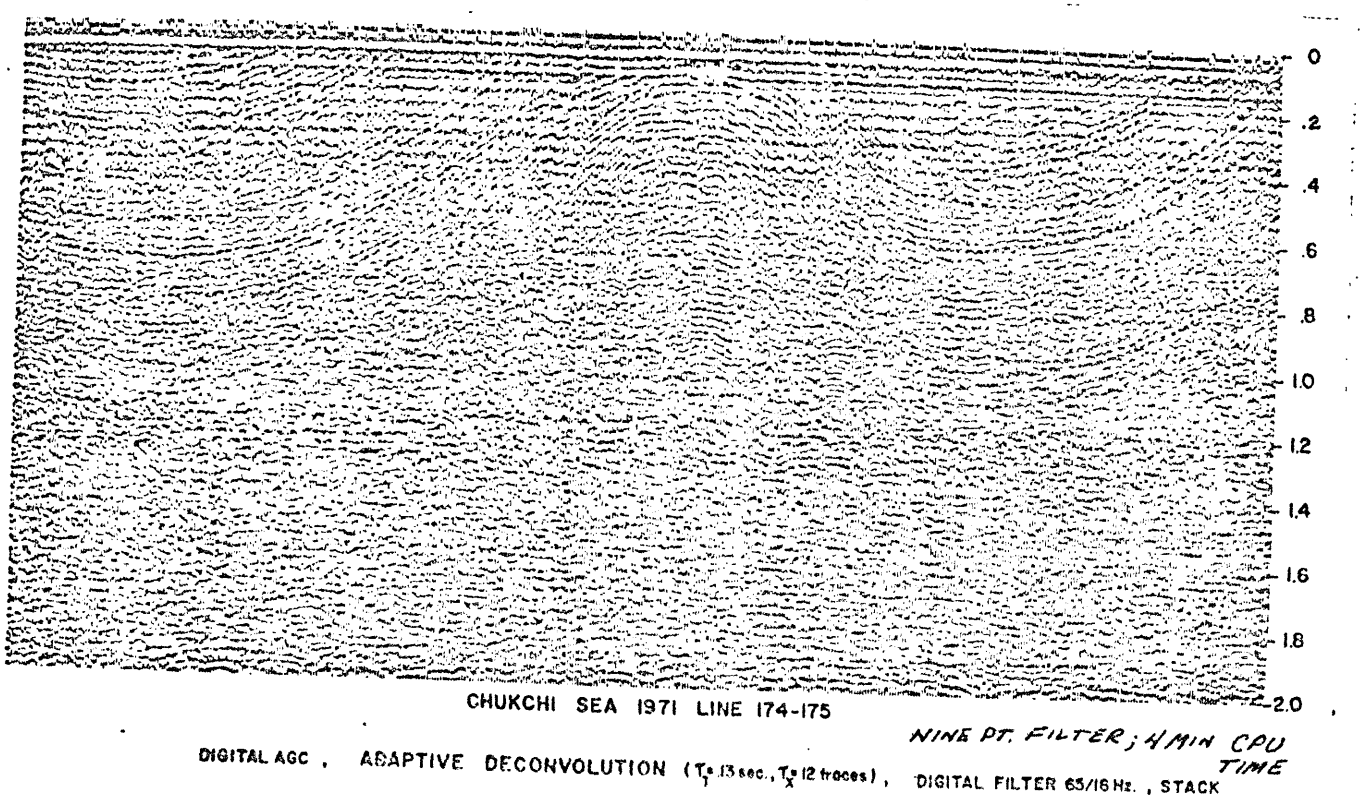


Figure 10

prediction error series. Displaying these simultaneously provides valuable information to determine the filter length or number of stages to be used. Thus, for these reasons we may expect that this approach represents a more general and a more effective method for deconvolution of non-stationary seismic data than methods based on stationary filters and statistics.

MAIN PROGRAMS

There are three versions of the Seismic Processing Package which are designed to process the three existing types of field data:

(1) 1971 reformatted 9-track reels

(2) 1971 field reels recorded on Teledyne 27110

(3) 1972 Analog/Digital reels

1. Package C. This processes the 1971 field data that has been reformatted by Teledyne. The data are in 9-track, 800 bpi, SEG interchange format. The first file on the tape is a line header. The data are written in the second file with each 4-second recording occupying one physical record. The first 100 bytes of each record is a shot header and is to be skipped. The data samples are in 2-byte (16 bit) two's complement representation.

2. Package CF. This processes the 1971 field data that were recorded directly in the field by a Teledyne 27110 digital recorder. These data are on 7-track 556 bpi tape. Each 4-second recording is placed in one physical record on the tape with the first 32 bytes record label information. Each data sample occupys 4-track bytes (24 bits). Assembly program DCDECT unpacks these data into standard IBM 16 bit two's complement representation.

3. Package CN. This processes the 1972 data that has been converted from analog to digital on the CDC 1700 at NCER. The data are on 7-track 556 bpi tape. Each 4-second recording is formatted into 2-2000 byte records multiplexing 2 gain channels. The samples are each 2 7-track bytes (12 bits) in one's complement representation. Assembly routine DCDECN unpacks these data into IBM 16-bit two's complement representation.

The primary difference between these programs is the initial handling of the input data. There are some slight differences in the default parameters and the separate main program listing for each package should be consulted when using a particular package.

Each package contains essentially four different processing sequences and is controlled by the NJØB parameter.

Following is a partial listing of the three main programs. The processing sequences for various NJØB parameter settings are given.

Three utility subroutines are used in these programs which were available at the Stanford Computation Center. 'MCLØCK(DATE,TIME,WEEKDA)' returns the execution date, time of day, and day of week in 8-byte EBCDIC format. 'PCLØCK(IJS)' is a partition interval timer in centiseconds. 'NØERRD(BUF, LEN,IERR,TAPEDD)' is a tape reading utility which reads one physical record from tape (DD'TAPEDD') of length 'LEN' in to buffer 'BUF'.

```
//LISTIT JOB 'C740,314','RILEY LIST PGMS'                                  JO
/* SERVICE LIST
C  U S G S  S E I S M I C  P R O C E S S I N G  P A C K A G E  C           US
C                                                                          US
      INTEGER TRACE*2(3950),BUF,LVERS(18)                                  US
      REAL*4 CX(20,1000),BRHO(10),BSIG(10),PLOC(10)                        US
      REAL*8 DATE,TIME,LINEID,TAPEDC,USBR                                  US
      COMMON /BLK1/BUF(2000)                                               US
      COMMON /BLK2/Y(1000),LX,NJUMP,NLPAK,RHO,CRHO,NDOTS                   US
      COMMON /BLK3/X(1000)                                                 US
      EQUIVALENCE (TRACE(1),BUF(26))                                       US
      NAMELIST /JOB/NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,RHO,
     1NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,NOTRJ,NJOB,TAPEDC,USBR,NPAGES
     2,RLINCH/ADAPT/FHIGH,NZC ,FHIGH2,FLOW2,LENBP,IFLGAP,LCN,DWARM,WSTRT
     3,ZTAU,XTAU/BIFLT/NPTS,PLOC,BSIG,BRHO/JOBIN/NREEL,LINEID,RANGE,WVEL
     4,SRATE,RECEND,NDEC,ITIME,RHO,NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,
     5NOTRJ,NJOB,TAPEDD,USER,NPAGES,RLINCH,FHIGH,NSTR,LENBP,IFLGAP,LCN,
     6DWARM,WSTRT,ZTAU,XTAU,FHIGH2,FLOW2,NPTS,PLOC,BSIG,BRHO,NZC
      DATA NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,NBITS,NBITS2,
     1NOTR,NOTRJ,NJOB,TAPEDD,FHIGH,NSTR,FLOW2,FHIGH2,LENBP,IFLGAP,LCN,
     2DWARM,WSTRT,ZTAU,XTAL,NPTS,NPAGES,RLINCH,PLOC,BRHO,BSIG,MODADD,
     3USER,NZC/999,'DEFERRED',400.,4875.,.002,1.53,2,540,5,4,2000,3,3,
     4'FT20F001',63.,1.  15.,63.,9,2,9,.2,.2,.12,10.,2,1,7.63,0.,1.536,
     58*0.,.5,.5,8*0.,.99,.99,8*0.0,9,'U.S.G.S.',1/
C                                                                          US
C_____________LIST__OF__PRINCIPAL__VARIABLES_____________________________
C
C  NAME (DEFAULT VALUE)          DEFINITION
C
C       <PARAMETERS USED IN ALL JOBS>
C  USBR    (U.S.G.S.)      USER'S NAME (8 ALPHANUMERIC)
C  NREEL   (9999)          USGS REEL NUMBER (INTEGER)
C  LINEID  (DEFERRED)      USGS LINE NUMBER (8 ALPHANUMERIC)
C  RANGE   (400.)          SHOT-RECEIVER OFFSET (FT.)
C  WVEL    (4875.)         WATER VELOCITY (FT/SEC)
C  SRATE   (.002)          INPUT SAMPLE RATE (SEC)
C  RECEND  (1.53)          RECORD LENGTH TO PROCESS (SEC)
C  NDEC    (2)             DEGREE OF DECIMATION (INTEGER)
C  ITIME   (540)           INTERNAL TIME ESTIMATE (SEC.,INTEGER)
C  RFC     (0.74)          DC REJECT FILTER COEFFICIENT
C  NBITS   (5)             NUMBER OF BITS PER TRACE (INTEGER)
C  NBITS2  (4)             SECONDARY TRACEWIDTH EVERY 'MODADD' TRACES (INT)
C  MODADD  (9)             MODULO TO ADD SECONDARY BITS (INTEGER)
C  NOTR    (2000)          NUMBER OF TRACES TO BE PROCESSED (INTEGER)
C  NOTRJ   (3)             PROCESSING FREQUENCY (1,2,OR 3)
C  NSTR    (1)             NUMBER OF STARTING TRACE ON TAPE (INTEGER)
C  NJOB    (3)             JOB SELECTION SWITCH (1,2,3,OR 4)
C  TAPEDD  (FT20F001)      TAPE DATA DEFINITION TAG (8 ALPHANUMERIC)
C  RLINCH  (7.63)          RECORD SECTION LENGTH IN INCHES (MAX.=15.25)
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,2,3>
C  NLEAK   (10)            INTEGRATION PARAMETER FOR AGC (INTEGER)
C  NJUMP   (24)            UPDATE FREQUENCY OF AGC GAIN TRACE (INTEGER)
C  LENBP   (9)             NUMBER OF COEFFICIENTS IN BANDPASS (INTEGER,ODD)
C  PHIGH   (63.)           PRIOR HIGH CUTOFF (CYCLES/SEC)
C  FLOW2   (15.)           POST-PROC. LOW CUTOFF (CYCLES/SEC)
C  PHIGH2  (63.)           POST-PROC. HIGH CUTOFF (CYCLES/SEC)
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,3>
C  NPTS    (2)             NO. POINTS ON BIFILT PROFILE (INTEGER)
C  PLOC    (0.,1.53)       LOCATION OF PROFILE TIES (SEC)
C  PRHO    (.5,.5)         RHO PROFILE COEFFICIENTS
```

```
C   BSIG    (.99,.99)       SIGMA PROFILE COEFFICIENTS
C        <ADDITIONAL PARAMETERS USED IN JOBS 2,3>
C   NZC     (1)             NO. OF LEADING ZERO REFLECTION COEFF.(INT)
C   IFLGAP  (2)             GAP BETWEEN ADAPTIVE FILTER COEFFICIENTS (INT)
C   LON     (9)             NO. OF REFLECTION COEFF. IN LADDER (INTEGER)
C   DWARM   (.2)            DURATION OF STATIONARY WARM-UP CYCLE (SEC)
C   WSTRT   (.2)            STARTING POSITION OF WARM-UP (SEC)
C   ZTAU    (.12)           RELAXATION TIME TO 1/E (SEC)
C   XTAU    (10.)           RELAXATION DISTANCE TO 1/E (TRACES)
C
C                (ALL VARIABLES ARE REAL EXCEPT AS INDICATED)
C                                                                       USC
C-----------------------------------------------------------------------LSC
C                                                                       USC
C            PROCESSING SEQUENCES:                                      USC
C                                                                       USC
C        NJOB=1    /FIELD/NMO/DCDEC/AGC/BPASS/BIFILT/BPASS2/SEISGM/     USC
C        NJOB=2    /FIELD/NMO/DCDEC/AGC/BPASS/BAFL/BPASS2/SEISGM/       USC
C        NJOB=3    /FIELD/NMO/DCDEC/AGC/BPASS/BAFL/BIFILT/BPASS2/SEISGM/ USC
C        NJOB=4    /FIELD/NMO/DCDEC/SEISGM/                             USC
C-----------------------------------------------------------------------USC
C            VERSION Q                          DON C. RILEY AUGUST 1972
C                                                                       USC
      CALL MCLOCK(DATE,TIME,WEEKDA)                                     USC
      CALL PCLOCK(IJS)                                                  USC
      RHO=0.74
      NLEAK=10
      NJUMP=24
      WRITE(6,876)                                                      USC
      READ(5,JOBIN,ERR=977)
    2 LENGTH=RECEND/SRATE                                               USC
      IF(NSTR.LE.1) GO TO 4
      M=NSTR-1
      DO 3 I=1,M
    3 CALL NOERRC(BUF,LEN,IERP,TAPEDD)
    4 LX=LENGTH/NDFC                                                    USC
      RLINCH=AMIN1(RLINCH,15.3)                                          C
      IF(RLINCH.GT.7.65) NPAGES=2                                       C
      SRATE2=SRATE*NDFC                                                 USC
      CRHO=(1.+RHO)/2.                                                  USC
      KTEST=24*NOTRJ                                                    USC
      WRITE(6,JOB)
      IF(NJOB.EQ.4)  GO TO 40                                           USC
      FLOW=-FHIGH
      IF(MOD(LENBP,2).EQ.0)  LENBP=LENBP+1                              USC
      CALL GFILT(FLOW,FHIGH,SRATE2,LENBP,FLOW2,FHIGH2,LX)               USC
      IF(NJOB.EQ.1) GO TO 30                                            USC
      WRITE(6,ADAPT)                                                    USC
      NSTRT=WSTRT/SRATE2
      NWARM=DWARM/SRATE2
      ZTAT=ZTAU/SRATE2
      IF(NJOB.EQ.2) GO TO 40                                            USC
   30 WRITE(6,BIFLT)
      CALL BIFILT(LX,BSIG,BRHO,PLOC,NPTS,SRATE2)
   40 CALL TAXIS(LX,SRATE2,NPAGES,RLINCH)                               C
      CALL NMOSET(RANGE,WVFL,SRATE,LENGTH)                              USC
      ITIME=ITIME*100
      CALL FIO999(44,LVERS(1),18,18)                                    USC
C                                                                       USC
      GO TO (1727,1707,1747,1737),NJOB                                  USC
```

```
C                                                                       USG
C                    NJOB=2                                             USG
C                                                                       USG
 1707 CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  USG
      CALL NMO                                                          USG
      CALL DCDEC(TRACE,NDEC)                                            USG
      CALL AGC                                                          USG
      CALL BPASS                                                        USG
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)             USG
      CALL BPASS2                                                       USG
      NDOTS=NBITS                                                       USG
      CALL SEISGM                                                       USG
      DO 707 I=NOTRJ,NOTR,NOTRJ                                         USG
      NOOTS=NBITS                                                       USG
      IF(MOD(I,MOCADD).EQ.0)  NDOTS=NBITS2                              USG
      CALL NOERRC(BUF,LEN,IERR,TAPEDD)                                  USG
      IF(IERR.EQ.1) GO TO 988                                           USG
      CALL NMO                                                          USG
      CALL DCDEC(TRACE,NDEC)                                            USG
      CALL AGC                                                          USG
      CALL BPASS                                                        USG
      IF(NOTRJ.GE.2) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                USG
      IF(IERR.EQ.1) GO TO 988                                           USG
      CALL BAFLGO                                                       USG
      IF(NOTRJ.GE.3) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                USG
      CALL BPASS2                                                       USG
      CALL SEISGM                                                       USG
      IF(MOD(I,KTEST).NE.0) GO TO 707                                   USG
      CALL PCLOCK(IJE,IJS)                                              USG
      IF(IJE.GT.ITIME) GO TO 909                                        USG
  707 CONTINUE                                                          USG
      GO TO 999                                                         USG
C                                                                       USG
C                                                                       USG
C                    NJOB=1                                             USG
C                                                                       USG
 1727 DO 727 I=NOTRJ,NOTR,NOTRJ                                         USG
      NDOTS=NBITS                                                       USG
      IF(MOD(I,MCDADD).EQ.0)  NDOTS=NBITS2                              USG
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  USG
      IF(IERR.EQ.1) GO TO 988                                           USG
      CALL NMO                                                          USG
      CALL DCDEC(TRACE,NDEC)                                            USG
      CALL AGC                                                          USG
      CALL BPASS                                                        USG
      CALL BIGOT                                                        USG
      IF(NOTRJ.GE.2) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                USG
      IF(NOTRJ.GE.3) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                USG
      IF(IERR.EQ.1) GO TO 988                                           USG
      CALL BPASS2                                                       USG
      CALL SEISGM                                                       USG
      IF(MOD(I,KTEST).NE.0) GO TO 727                                   USG
      CALL PCLOCK(IJE,IJS)                                              USG
      IF(IJE.GT.ITIME) GO TO 909                                        USG
  727 CONTINUE                                                          USG
      GO TO 999                                                         USG
C                                                                       USG
C                    NJOB=4                                             USG
C                                                                       USG
 1737 DO 737 I=NOTRJ,NOTR,NOTRJ                                         USG
```

```
      NDOTS=NBITS                                                       US
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              US
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  US
      IF(IERR.EQ.1) GO TO 988                                           US
      CALL NMO                                                          US
      CALL DCDEC(TRACE,NDEC)                                            US
      IF(NOTRJ.GE.2) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                US
      IF(NOTRJ.GE.3) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                US
      IF(IERR.EQ.1) GO TO 988                                           US
      CALL SETSGM                                                       US
      IF(MOD(I,KTEST).NE.0) GO TO 737                                   US
      CALL PCLOCK(IJE,IJS)                                              US
      IF(IJE.GT.ITIME) GO TO 909                                        US
  737 CONTINUE                                                          US
      GO TO 999                                                         US
C                                                                       US
C                   NJOB=3                                              US
C                                                                       US
 1747 CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  US
      CALL NMO                                                          US
      CALL DCDEC(TRACE,NOEC)                                            US
      CALL AGC                                                          US
      CALL BPASS                                                        US
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARN,NSTRT,ZTAT,XTAU,NZC)             US
      CALL BIGOT                                                        US
      CALL BPASS2                                                       US
      NDOTS=NBITS                                                       US
      CALL SETSGM                                                       US
      DO 747 I=NOTRJ,NOTR,NOTRJ                                         US
      NDOTS=NBITS                                                       US
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              US
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  US
      IF(IERR.EQ.1) GO TO 988                                           US
      CALL NMO                                                          US
      CALL DCDEC(TRACE,NDEC)                                            US
      CALL AGC                                                          US
      CALL BPASS                                                        US
      IF(NOTRJ.GE.2) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                US
      IF(IERR.EQ.1) GO TO 988                                           US
      CALL BAFLGD                                                       US
      CALL BIGOT                                                        US
      IF(NOTRJ.GE.3) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                US
      CALL BPASS2                                                       US
      CALL SETSGM                                                       US
      IF(MOD(I,KTEST).NE.0) GO TO 747                                   US
      CALL PCLOCK(TJE,TJS)                                              US
      IF(IJE.GT.ITIME) GO TO 909                                        US
  747 CONTINUE                                                          US
  << MORE MAIN FOLLOWS, BUT NOT LISTED >>
/*
```

```
RALPH JOB STATISTICS --     231 CARDS READ --        231 LINES PRINTED --
                          0.00 MINUTES CPU TIME     0.00 MINUTES WAIT TIME
```

```
//LISTIT JOB 'C743,314','FILBY LIST PGMS'                                JOB
/*  SERVICE LIST
C   U S G S   S E I S M I C   P R O C E S S I N G   P A C K A G E  (F      CF
C   PROCESSES 1971 FIELD DATA RECORDED WITH TELEDYNE MODEL 27710
C         SAMPLE DD CARD FOR TAPE INPUT: ASSUME TAPEDD='FT20F001'
C     //GO.FT20F001 DD DSN=SEIS,UNIT=0C1,VOL=SER=UXXXX,DISP=(OLD,KEEP),
C     //   LABEL=(1,BLP,,IN),DCB=(RECFM=U,BLKSIZE=16000,DEN=1,TRTCH=C)
C        REPLACE UXXXX BY COMP. CENTER TAPE I.D., LIKE U1577
C                                                                        CF
       INTEGER BUF*2(4000),LVERS(18)
       REAL*4 CX(20,1000),BRHO(10),BSIG(10),PLOC(10)                     CF
       REAL*8 DATE,TIME,LINEID,TAPEDD,USER                               CF
       COMMON /BLK2/Y(1000),LX,NJUMP,NLEAK,RHO,CRHO,NDOTS                CF
       COMMON /BLK3/X(1000)                                              CF
       NAMELIST /JOB/NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,RHO,CF
      1NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,NOTRJ,NJOB,TAPEDD,USER,NPAGESC F
      2,RLINCH/ADAPT/FHIGH,NZC ,FHIGH2,FLOW2,LENBP,IFLGAP,LCH,DWARM,WSTRTC F
      3,ZTAU,XTAU/BIFLT/NPTS,PLOC,BSIG,BRHO/JOBIN/NREEL,LINEID,RANGE,WVELC F
      4,SRATE,RECEND,NDEC,ITIME,RHO,NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,CF
      5NOTRJ,NJOB,TAPEDD,USER,NPAGES,RLINCH,FHIGH,NSTR,LENBP,IFLGAP,LCH, CF
      6DWARM,WSTRT,ZTAU,XTAU,FHIGH2,FLOW2,NPTS,PLOC,BSIG,BRHO,NZC       CF
       DATA NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,NBITS,NBITS2,CF
      1NOTR,NOTRJ,NJOB,TAPEDD,FHIGH,NSTR,FLOW2,FHIGH2,LENBP,IFLGAP,LCH, CF
      2DWARM,WSTRT,ZTAU,XTAU,NPTS,NPAGES,RLINCH,PLOC,BRHO,BSIG,MODADD,  CF
      3USER,NZC/999,'DEFERRED',400.,4875.,.004,1.53,1,540,5,4,2000,3,3, CF
      4'FT20F001',63.,1,  15.,63.,9,2,9,.2,.2,.12,10.,2,1,7.63,0.,1.536,CF
      58*0.,.5,.5,8*0.,.99,.99,8*0.0,9,'U.S.G.S.',0/                    CF
C                                                                       CF
C_____________LIST__OF__PRINCIPAL__VARIABLES_____________________________CF
C                                                                       CF
C  NAME (DEFAULT VALUE)          DEFINITION                             CF
C                                                                       CF
C       <PARAMETERS USED IN ALL JOBS>                                   CF
C  USER    (U.S.G.S.)      USER'S NAME (8 ALPHANUMERIC)                 CF
C  NREEL   (9999)          USGS REEL NUMBER (INTEGER)                   CF
C  LINEID  (DEFERRED)      USGS LINE NUMBER (8 ALPAHNUMERIC)            CF
C  RANGE   (400.)          SHOT-RECEIVER OFFSET (FT.)                   CF
C  WVEL    (4875.)         WATER VELOCITY (FT/SEC)                      CF
C  SRATE   (.004)          INPUT SAMPLE RATE (SEC)                      CF
C  RECEND  (1.53)          RECORD LENGTH TO PROCESS (SEC)               CF
C  NDEC    (1)             DEGREE OF DECIMATION (INTEGER)               CF
C  ITIME   (540)           INTERNAL TIME ESTIMATE (SEC.,INTEGER)        CF
C  RHO     (0.74)          DC REJECT FILTER COEFFICIENT                 CF
C  NBITS   (5)             NUMBER OF DOTS PER TRACE (INTEGER)           CF
C  NBITS2  (4)             SECONDARY TRACEWIDTH EVERY 'MODADD' TRACES (INT)CF
C  MODADD  (9)             MODULO TO ADD SECONDARY BITS (INTEGER)       CF
C  NOTR    (2000)          NUMBER OF TRACES TO BE PROCESSED (INTEGER)   CF
C  NOTRJ   (3)             PROCESSING FREQUENCY (1,2,OR 3)              CF
C  NSTR    (1)             NUMBER OF STARTING TRACE ON TAPE (INTEGER)   CF
C  NJOB    (3)             JOB SELECTION SWITCH (1,2,3,OR 4)            CF
C  TAPEDD  (FT20F001)      TAPE DATA DEFINITION TAG (8 ALPHANUMERIC)    CF
C  RLINCH  (7.63)          RECORD SECTION LENGTH IN INCHES (MAX.=15.25)  CF
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,2,3>                      CF
C  NLEAK   (10)            INTEGRATION PARAMETER FOR AGC (INTEGER)      CF
C  NJUMP   (24)            UPDATE FREQUENCY OF AGC GAIN TRACE (INTEGER) CF
C  LENBP   (9)             NUMBER OF COEFFICIENTS IN BANDPASS (INTEGER,ODD)
C  FHIGH   (63.)           ERROR HIGH CUTOFF (CYCLES/SEC)               CF
C  FLOW2   (15.)           POST-PROC. LOW CUTOFF (CYCLES/SEC)           CF
C  FHIGH2  (63.)           POST-PROC. HIGH CUTOFF (CYCLES/SEC)          CF
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,3>                        CF
```

```
C  NPTS    (2)            NO. POINTS ON BIFILT PROFILE (INTEGER)          CF
C  PLOC    (0.,1.53)      LOCATION OF PROFILE TIES (SEC)                  CF
C  BRHO    (.5,.5)        RHO PROFILE COEFFICIENTS                        CF
C  BSIG    (.99,.99)      SIGMA PROFILE COEFFICIENTS                      CF
C       <ADDITIONAL PARAMETERS USED IN JOBS 2,3>                          CF
C  NZC     (0)            NO. OF LEADING ZERO REFLECTION COEFF.(INT)      CF
C  IFLGAP  (2)            GAP BETWEEN ADAPTIVE FILTER COEFFICIENTS (INT)  CF
C  LCN     (9)            NO. OF REFLECTION COEFF. IN LADDER (INTEGER)    CF
C  DWARM   (.2)           DURATION OF STATIONARY WARM-UP CYCLE (SEC)      CF
C  WSTRT   (.2)           STARTING POSITION OF WARM-UP (SEC)              CF
C  ZTAU    (.12)          RELAXATION TIME TO 1/E (SEC)                    CF
C  XTAU    (10.)          RELAXATION DISTANCE TO 1/E (TRACES)             CF
C                                                                         CF
C              (ALL VARIABLES ARE REAL EXCEPT AS INDICATED)               CF
C                                                                         CF
C-------------------------------------------------------------------------CF
C                                                                         CF
C           PROCESSING SEQUENCES:                                         CF
C                                                                         CF
C       NJOB=1   /FIELD/   /DCDEC/   /BPASS/BIFILT/BPASS2/SEISGM/        CF
C       NJOB=2   /FIELD/   /DCDEC/   /BPASS/BAFL/BPASS2/SEISGM/          CF
C       NJOB=3   /FIELD/   /DCDEC/   /BPASS/BAFL/BIFILT/BPASS2/SEISGM/   CF
C       NJOB=4   /FIELD/   /DCDEC/SEISGM/                                 CF
C-------------------------------------------------------------------------CF
C           VERSION G                          DON C. RILEY DECEMBER 1972 CF
C                                                                         CF
      CALL MCLOCK(DATE,TIME,WEEKDA)                                       CF
      CALL PCLOCK(IJS)                                                    CF
      RHO=0.74                                                            CF
      NLEAK=10                                                            CF
      NJUMP=24                                                            CF
      WRITE(6,376)                                                        CF
      READ(5,JOBIN,ERR=977)                                               CF
    2 LENGTH=RECEND/SRATE                                                 CF
      IF(NSTR.LE.1) GO TO 4                                               CF
      M=NSTR-1                                                            CF
      DO 3 I=1,M                                                          CF
    3 CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                    CF
    4 LX=LENGTH/NDEC                                                      CF
      RLINCH=AMIN1(RLINCH,15.3)                                           CF
      IF(RLINCH.GT.7.65) NPAGES=2                                         CF
      SRATE2=SRATE*NDEC                                                   CF
      CRHO=(1.+RHO)/2.                                                    CF
      KTEST=24*NOTRJ                                                      CF
      WRITE(6,JOB)                                                        CF
      IF(NJOB.EQ.4)  GO TO 40                                             CF
      FLOW=-FHIGH                                                         CF
      IF(MOD(LENBP,2).EQ.0)  LENBP=LENBP+1                                CF
      CALL GFILT(FLOW,FHIGH,SRATE2,LENBP,FLOW2,FHIGH2,LX)                 CF
      IF(NJOB.EQ.1) GO TO 30                                              CF
      WRITE(6,ADAPT)                                                      CF
      NSTRT=WSTRT/SRATE2                                                  CF
      NWARM=DWARM/SRATE2                                                  CF
      ZTAT=ZTAU/SRATE2                                                    CF
      IF(NJOB.EQ.2) GO TO 40                                              CF
   30 WRITE(6,BIFLT)
      CALL BIFILT(LX,BSIG,BRHO,PLOC,NPTS,SRATE2)
   40 CALL TAXIS(LX,SRATE2,NPAGES,RLINCH)                                 CF
CON   CALL NMOSET(RANGE,WVEL,SPATE,LENGTH)                                CF
      ITIME=ITIME*100                                                     CF
```

```
      CALL FIO999(44,LVERS(1),1e,1A)                                    CF
C                                                                       CF
      GO TO (1727,1707,1747,1737),NJOB                                  CF
C                                                                       CF
C                     NJOB=2                                            CF
C                                                                       CF
 1707 CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  CF
      CALL DCDECT(BUF)
COUT  CALL AGC                                                          CF
      CALL BPASS                                                        CF
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,MSTRT,ZTAT,XTAU,NZC)             CF
      CALL BPASS2                                                       CF
      NDOTS=NBITS                                                       CF
      CALL SEISGM                                                       CF
      DO 707 I=NOTRJ,NOTP,NOTRJ                                         CF
      NDOTS=NBITS                                                       CF
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  CF
      IF(IERR.EQ.1) GO TO 988                                           CF
      CALL DCDECT(BUF)
COUT  CALL AGC                                                          CF
      CALL BPASS                                                        CF
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 CF
      IF(IERR.EQ.1) GO TO 988                                           CF
      CALL BAFLGO                                                       CF
      IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 CF
      CALL BPASS2                                                       CF
      CALL SEISGM                                                       CF
      IF(MOD(I,KTEST).NE.0) GO TO 707                                   CF
      CALL PCLOCK(IJE,IJS)                                              CF
      IF(IJE.GT.ITIME) GO TO 909                                        CF
  707 CONTINUE                                                          CF
      GO TO 999                                                         CF
C                                                                       CF
C                                                                       CF
C                     NJOB=1                                            CF
C                                                                       CF
 1727 DO 727 I=NOTRJ,NOTR,NOTRJ                                         CF
      NDOTS=NBITS                                                       CF
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              CF
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  CF
      IF(IERR.EQ.1) GO TO 988                                           CF
      CALL DCDECT(BUF)
COUT  CALL AGC                                                          CF
      CALL BPASS                                                        CF
      CALL BIGOT                                                        CF
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 CF
      IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 CF
      IF(IERR.EQ.1) GO TO 988                                           CF
      CALL BPASS2                                                       CF
      CALL SEISGM                                                       CF
      IF(MOD(I,KTEST).NE.0) GO TO 727                                   CF
      CALL PCLOCK(IJE,IJS)                                              CF
      IF(IJE.GT.ITIME) GO TO 909                                        CF
  727 CONTINUE                                                          CF
      GO TO 999                                                         CF
C                                                                       CF
C                     NJOB=4                                            CF
C                                                                       CF
 1737 DO 737 I=NOTRJ,NOTR,NOTPJ                                         CF
```

```
      NDOTS=NBITS                                                       CF
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              CF
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  CF
      IF(IERR.EQ.1) GO TO 988                                           CF
      CALL DCDECT(BUF)
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 CF
      IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 CF
      IF(IERR.EQ.1) GO TO 988                                           CF
      CALL SEISGM                                                       CF
      IF(MOD(I,KTEST).NE.0) GO TO 737                                   CF
      CALL PCLOCK(IJE,IJS)                                              CF
      IF(IJE.GT.ITIME) GO TO 999                                        CF
  737 CONTINUE                                                          CF
      GO TO 999                                                         CF
C                                                                       CF
C                   NJOB=3                                              CF
C                                                                       CF
 1747 CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  CF
      CALL DCDECT(BUF)
COUT  CALL AGC                                                          CF
      CALL BPASS                                                        CF
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)             CF
      CALL BIGOT                                                        CF
      CALL BPASS2                                                       CF
      NDOTS=NBITS                                                       CF
      CALL SEISGM                                                       CF
      DO 747 I=NOTRJ,NOTR,NOTRJ                                         CF
      NDOTS=NBITS                                                       CF
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              CF
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  CF
      IF(IERR.EQ.1) GO TO 988                                           CF
      CALL DCDECT(BUF)
COUT  CALL AGC                                                          CF
      CALL BPASS                                                        CF
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 CF
      IF(IERR.EQ.1) GO TO 988                                           CF
      CALL BAFLGO                                                       CF
      CALL BIGOT                                                        CF
      IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 CF
      CALL BPASS2                                                       CF
      CALL SEISGM                                                       CF
      IF(MOD(I,KTEST).NE.0) GO TO 747                                   CF
      CALL PCLOCK(IJE,IJS)                                              CF
      IF(IJE.GT.ITIME) GO TO 999                                        CF
  747 CONTINUE                                                          CF
  << MORE MAIN FOLLOWS, BUT NOT LISTED >>
/*
```

RALPH JOB STATISTICS -- 228 CARDS READ -- 221 LINES PRINTED --
0.00 MINUTES CPU TIME 0.00 MINUTES WAIT TIME

```
//LISTTT JOB '0740,514','FILCY LIST PGMS'                                JOB
/* SERVICE LIST
C  U S G S  S E I S M I C  P R O C E S S I N G  P A C K A G E  CN          CN
C  PROCESSES 1972 ANALOG TO DIGITAL TAPES PRODUCED ON THE CDC 1700        CN
C  MACHINE AT N.C.E.E.  DOUBLE MULTIPLEX 556 BPI TAPES                    CN
C        SAMPLE DD CARD FOR TAPE INPUT; ASSUME TAPEDD='FT20F001'          CN
C    //GO.FT20F001 DD DSN=SEIS,UNIT=001,VOL=SER=UXXXX,DISP=(OLD,KEEP),    CN
C    //  LABEL=(1,BLP,,IN),DCB=(RECFM=U,BLKSIZE=16000,DEN=1,TRTCH=C)      CN
C       REPLACE UXXXX BY COMP. CENTER TAPE I.D., LIKE U1577              CN
C                                                                        CN
      INTEGER BUF*2(4000),LVERS(10)                                      CN
      REAL*4 CX(20,1000),BRHO(10),BSIG(10),PLOC(10)                      CN
      REAL*8 DATE,TIME,LINEID,TAPEDD,USER                                CN
      COMMON /BLK2/Y(1000),LX,NJUMP,NLEAK,RHO,CRHO,NDOTS                 CN
      COMMON /BLK3/X(1000)                                               CN
      NAMELIST /JOB/NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,RHO, CN
     1NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,NOTRJ,NJOB,TAPEDD,USER,NPAGES CN
     2,RLINCH/ADAPT/FHIGH,NZC ,FHIGH2,FLOW2,LENBP,IFLGAP,LCN,DWARM,WSTPT CN
     3,ZTAU,XTAU/BIFLT/NPTS,PLOC,RSIG,BRHO/JOBIN/NREEL,LINEID,RANGE,WVEL CN
     4,SRATE,RECEND,NDEC,ITIME,RHO,NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR, CN
     5NOTRJ,NJOB,TAPEDD,USER,NPAGES,RLINCH,FHIGH,NSTR,LENBP,IFLGAP,LCN,  CN
     6DWARM,WSTRT,ZTAU,XTAU,FHIGH2,FLOW2,NPTS,PLOC,BSIG,BRHO,NZC         CN
      DATA NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,NBITS,NBITS2, CN
     1NOTR,NOTRJ,NJOB,TAPEDD,FHIGH,NSTR,FLOW2,FHIGH2,LENBP,IFLGAP,LCN,   CN
     2DWARM,WSTRT,ZTAU,XTAU,NPTS,NPAGES,RLINCH,PLOC,BRHO,BSIG,MODADD,    CN
     3USER,NZC/999,'DEFERRED',400.,4875.,.004,1.53,1,540,5,4,2000,2,3,   CN
     4'FT20F001',63.,1,  15.,63.,9,2,9,.2,.2,.12,10.,2,1,7.63,0.,1.536,  CN
     58*0.,.5,.5,8*0.,.09,.99,3*0.0,9,'U.S.G.S.',0/                      CN
C                                                                        CN
C_____________LIST__OF__PRINCIPAL__VARIABLES_____________________________CN
C                                                                        CN
C  NAME (DEFAULT VALUE)          DEFINITION                              CN
C                                                                        CN
C       <PARAMETERS USED IN ALL JOBS>                                    CN
C  USER    (U.S.G.S.)      USER'S NAME (8 ALPHANUMERIC)                  CN
C  NREEL   (9999)          USGS REEL NUMBER (INTEGER)                    CN
C  LINEID  (DEFERRED)      USGS LINE NUMBER (8 ALPAHNUMERIC)             CN
C  RANGE   (400.)          SHOT-RECEIVER OFFSET (FT.)                    CN
C  WVEL    (4875.)         WATER VELOCITY (FT/SEC)                       CN
C  SRATE   (.004)          INPUT SAMPLE RATE (SEC)                       CN
C  RECEND  (1.53)          RECORD LENGTH TO PROCESS (SEC)                CN
C  NDEC    (1)             DEGREE OF DECIMATION (INTEGER)    **NOT USED**CN
C  ITIME   (540)           INTERNAL TIME ESTIMATE (SEC.,INTEGER)         CN
C  RHO     (0.74)          DC REJECT FILTER COEFFICIENT  **NOT USED***   CN
C  NBITS   (5)             NUMBER OF DOTS PER TRACE (INTEGER)            CN
C  NBITS2  (4)             SECONDARY TRACEWIDTH EVERY 'MODADD' TRACES (INT) CN
C  MODADD  (9)             MODULO TO ADD SECONDARY BITS (INTEGER)        CN
C  NOTR    (2000)          NUMBER OF TRACES TO BE PROCESSED (INTEGER)    CN
C  NOTRJ   (2)             PROCESSING FREQUENCY (1 OR 2)                 CN
C  NSTR    (1)             NUMBER OF STARTING TRACE ON TAPE (INTEGER)    CN
C  NJOB    (3)             JOB SELECTION SWITCH (1,2,3,OR 4)             CN
C  TAPEDD  (FT20F001)      TAPE DATA DEFINITION TAG (8 ALPHANUMERIC)     CN
C  RLINCH  (7.63)          RECORD SECTION LENGTH IN INCHES (MAX.=15.25)  CN
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,2,3>                       CN
C  NLEAK   (10)            INTEGRATION PARAMETER FOR AGC (INTEGER)       CN
C  NJUMP   (24)            UPDATE FREQUENCY OF AGC GAIN TRACE (INTEGER)  CN
C  LENBP   (9)             NUMBER OF COEFFICIENTS IN BANDPASS (INTEGER,ODD) CN
C  FHIGH   (63.)           PRIOR HIGH CUTOFF (CYCLES/SEC)                CN
C  FLOW2   (15.)           POST-PROC. LOW CUTOFF (CYCLES/SEC)            CN
C  FHIGH2  (63.)           POST-PROC. HIGH CUTOFF (CYCLES/SEC)           CN
```

```
C        <ADDITIONAL PARAMETERS USED IN JOBS 1,3>
C   NPTS     (2)              NO. POINTS ON BIFILT PROFILE (INTEGER)
C   PLOC     (0.,1.53)        LOCATION OF PROFILE TIES (SEC)
C   BRHO     (.5,.5)          RHO PROFILE COEFFICIENTS
C   BSIG     (.99,.99)        SIGMA PROFILE COEFFICIENTS
C        <ADDITIONAL PARAMETERS USED IN JOBS 2,3>
C   NZC      (0)              NO. OF LEADING ZERO REFLECTION COEFF.(INT)
C   IFLGAP   (2)              GAP BETWEEN ADAPTIVE FILTER COEFFICIENTS (INT)
C   ICN      (9)              NO. OF REFLECTION COEFF. IN LADDER (INTEGER)
C   DWARM    (.2)             DURATION OF STATIONARY WARM-UP CYCLE (SEC)
C   WSTRT    (.2)             STARTING POSITION OF WARM-UP (SEC)
C   ZTAU     (.12)            RELAXATION TIME TO 1/E (SEC)
C   XTAU     (10.)            RELAXATION DISTANCE TO 1/E (TRACES)
C
C              (ALL VARIABLES ARE REAL EXCEPT AS INDICATED)
C
C-------------------------------------------------------------------
C
C            PROCESSING SEQUENCES:
C
C        NJOB=1    /FIELD/    /DCDEC/    /BPASS/BIFILT/BPASS2/SEISGM/
C        NJOB=2    /FIELD/    /DCDEC/    /BPASS/BAFL/BPASS2/SEISGM/
C        NJOB=3    /FIELD/    /DCDEC/    /BPASS/BAFL/BIFILT/BPASS2/SEISGM/
C        NJOB=4    /FIELD/    /DCDEC/SEISGM/
C-------------------------------------------------------------------
C            VERSION G                          DON C. RILEY DECEMBER 1972
C
      CALL MCLOCK(DATE,TIME,WEEKDA)
      CALL PCLOCK(IJS)
      RHO=0.74
      NLEAK=10
      NJUMP=24
      WRITE(6,876)
      READ(5,JOBIN,ERR=977)
    2 LENGTH=RECEND/SRATE
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      IF(NSTR.LE.1) GO TO 4
      M=NSTR-1
      DO 3 I=1,M
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
    3 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
    4 LX=LENGTH/NDEC
      NOTRJ=MIND(NOTRJ,2)
      RLINCH=AMIN1(RLINCH,15.3)
      IF(RLINCH.GT.7.65) NPAGES=2
      SRATE2=SRATE*NDEC
C     CRHO=(1.+PHI)/2.          ** NOT USED **
      KTEST=24*NOTRJ
      WRITE(6,JOB)
      IF(NJOB.EQ.4)  GO TO 40
      FLOW=-FHIGH
      IF(MOD(LENBP,2).EQ.0)  LENBP=LENBP+1
      CALL GFILT(FLOW,FHIGH,SRATE2,LENBP,FLOW2,FHIGH2,LX)
      IF(NJOB.EQ.1) GO TO 30
      WRITE(6,ADAPT)
      NSTRT=WSTRT/SRATE2
      NWARM=DWARM/SRATE2
      ZTAT=ZTAU/SRATE2
      IF(NJOB.EQ.2) GO TO 40
   30 WRITE(6,BIFLT)
```

```
      CALL BIFILT(LX,BSIG,BRHO,PLOC,NPTS,SRATE2)
   40 CALL TAXIS(LX,SRATE2,NPAGES,RLINCH)
CON   CALL NMOSET(RANGE,WVEL,SRATE,LENGTH)
      ITIME=ITIME*100
      CALL FID999(44,LVERS(1),18,18)
C
      GO TO (1727,1707,1747,1737),NJOB
C
C                  NJOB=2
C
 1707 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      CALL DCDECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)
      CALL BPASS2
      NDOTS=NBITS
      CALL SEISGM
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      DO 707 I=NOTRJ,NOTR,NOTRJ
      NDOTS=NBITS
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL DCDECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL BAFLGO
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      CALL BPASS2
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL SEISGM
      IF(MOD(I,KTEST).NE.0) GO TO 707
      CALL PCLOCK(IJE,IJS)
      IF(IJE.GT.ITIME) GO TO 999
  707 CONTINUE
      GO TO 999
C
C
C                  NJOB=1
C
 1727 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      DO 727 I=NOTRJ,NOTR,NOTRJ
      NDOTS=NBITS
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 989
      CALL DCDECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      CALL BIGOT
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL BPASS2
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL SEISGM
```

```
      IF(MOD(I,KTEST).NE.0) GO TO 727
      CALL PCLOCK(IJE,IJS)
      IF(IJE.GT.ITIME) GO TO 909
  727 CONTINUE
      GO TO 999
C
C                   NJOB=4
C
 1737 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      DO 737 I=NOTRJ,NOTR,NOTRJ
      NDOTS=NBITS
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL DCDECN(BUF,LX,Y)
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL SEISGM
      IF(MOD(I,KTEST).NE.0) GO TO 737
      CALL PCLOCK(IJE,IJS)
      IF(IJE.GT.ITIME) GO TO 909
  737 CONTINUE
      GO TO 999
C
C                   NJOB=3
C
 1747 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      CALL DCDECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)
      CALL BIGOT
      CALL BPASS2
      NDOTS=NBITS
      CALL SEISGM
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      DO 747 I=NOTRJ,NOTR,NOTRJ
      NDOTS=NBITS
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL DCDECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL BAFLGO
      CALL BIGOT
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      CALL BPASS2
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL SEISGM
      IF(MOD(I,KTEST).NE.0) GO TO 747
      CALL PCLOCK(IJE,IJS)
      IF(IJE.GT.ITIME) GO TO 909
  747 CONTINUE
  << MORE MAIN FOLLOWS, BUT NOT LISTED >>
```

PRIMARY SUBROUTINES

Time and Space Adaptive Deconvolution.--The basic enhancement tool of the processing package is data adaptive deconvolution as described in the section under "Theoretical Considerations". The subroutine called "BAFL" is a fortran implementation of this method. Basically the procession of the data when using this feature is controlled by four parameters: LCN, IFLGAP, ZTAU, and XTAU. LCN is the integer number of reflection coefficients or stages in the adaptive ladder. The number of equivalent filter coefficients in the resulting prediction error filtering operation is thus equal to LCN + 1. As the number of reflection coefficients (or equivalent filter length) increases the quality of the filtering also increases. However, the cost (execution time) also increases with LCN. A filter length corresponding to LCN = 9 or 10 has been found to do a good job of filtering with reasonable execution speeds. The integer parameter IFLGAP essentially allows decimating the filter operator in time. This has the effect of deconvoluting only the lower fraction fn/IFLGAP of the spectrum of the data (fn is the folding frequency). Thus, if we had data sampled at 2 msec where the folding frequency is 250 hz ($fn = 1/2\Delta t$) and we knew that there was no useful information in the band 125→250 hz, then setting IFLGAP = 2 would deconvolve the useful band 0→250/2 hz. Of course, the data would have to be low pass filtered with the cutoff at 125 hz before deconvolution to avoid aliasing effects. If instead, we set IFLGAP = 3, then the band 0→83.3 hz would be whitened or deconvolved.

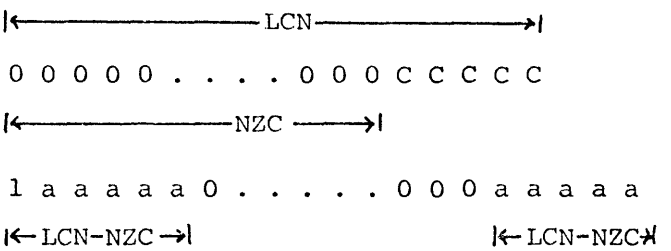
Sample Rate (m/sec)	IFLGAP	Deconvolved Portion (hz)
2	1	0 - 250
	2	0 - 125
	3	0 - 83
	4	0 - 63
4	1	0 - 125
	2	0 - 63
	3	0 - 42

The parameters ZTAU and XTAU represent the relaxation time of the adaptive processor in time (sec) and space (trans), respectively. These parameters are set by the user based on the degree of variation in the data that may be expected in time and space. The choice of these relaxation times is mainly a matter of experience but some guidelines may be in the following table.

	In Space XTAU =	In Time ZTAU =
Rapid Variation	5.	.08
Moderate Variation	10.	.12
Small Variation	15.	.16

The parameter NZC is the number of leading stages or reflection coefficients that are held to zero. The effect of this parameter on the resulting deconvolved spectrum is potentially of fundamental importance in both adaptive and stationary deconvolution of seismic data. Frequently data is encountered

where the reverberation period is moderate to long. Often the approach is to use a gapped prediction-error operator of the form $(1,0,0,\ldots,a_1,a_2,\ldots a_N)$ to predict greater than unit span while restricting the degrees of freedom. Clearly, such a procedure is not minimum-phase. If however, we set the first N reflection coefficients to zero the effect on the resultant filtering is that we are predicting greater than unit distance while retaining the minimum-phase characteristic. This has the physical interpretation of implying that there are no reflectors in the early section of the model as in the case of a water layer. The relationship between the prediction-error operator $(1,a_1,a_2,\ldots,a_{LCN+1})$ and the gapped reflection coefficent series $(C_1,C_2,\ldots,C_{LCN})$ is as follows:



For traditional unit prediction, set NZC=0.

The two remaining parameters are DWARM and WSTRT which control the adaption warm up routine necessary to initialize the backward state variable vector in the filter ladder. These parameters are used only on the first trace of the section and the resulting processing of subsequent traces is extremely insensitive to the initialization routine. It is recommended that the default parameters be used for these two parameters.

```
EL 20.1 (AUG 71)                        OS/360  FORTRAN H

     COMPILER OPTIONS - NAME= USGSC,OPT=02,LINECNT=58,SIZE=0000K,
                        SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,NOXREF
S    002            SUBROUTINE BAFL(LOOT,CX,IFLGAP,LCN,NWARM,ISTRT,ZTAU,XTAU,NZERO)
                C
                C
                C              ---------------------------
                C--------------S U B R O U T I N E   B A F L----------------
                C              ---------------------------
                C
                C       THE BURG ADAPTIVE FILTER LADDER: AN ADAPTIVE OR TIME-VARYING
                C       FIXED-LEAD PREDICTION ERROR PROCESSOR.  ADJUSTMENT OF EACH
                C       REFLECTION COEFFICIENT IS MADE EVERY JUMP STATE ATTEMPTING
                C       TO MINIMIZE THE STAGE OUTPUT POWER.
                C       INPUTS:
                C         X(1)...X(LX)=INITIAL DATA
                C         LCN= LAST NON-ZERO REFLECTION COEFFICIENT
                C         IFLGAP=  NUMBER OF GAPS BETWEEN FILTER COEFFICIENTS
                C              SETTING IFLGAP=1 DOES NOT GAP THE F.C. AND
                C              TRIES TO OPERATE ON THE ENTIRE SPECTRUM FROM
                C              0 TO W WHERE W IS THE FOLDING FREQUENCY.
                C              SETTING IFLGAP=2 OPERATES ON THE PORTION OF
                C              THE SPECTRUM FROM 0 TO W/2 , IFLGAP=3 FROM
                C              0 TO W/3 ETC. THUS ALLOWING SPECIFICATION OF
                C              WHAT PART OF THE SPECTRUM TO DECONVOLVE.
                C
                C         NWARM=DURATION OF STATIONARY C ESTIMATION CYCLE
                C         ISTRT=START OF STATIONARY GATE
                C         ZTAU=TEMPORAL RELAXATION TIME TO 1/E
                C         XTAU=SPATIAL RELAXATION DISTANCE TO 1/E
                C       OUTPUTS:
                C         X(1)...X(LOOT)=FORWARD ERROR PREDICTION TRACE
                C       OTHER VARIABLES:
                C         F(1)...F(LCN)=FORWARD STATE VECTOR
                C         B(1)...B(LCN)=BACKWARD STATE VECTOR
                C         C(1)...C(LCN)=REFLECTION COEFFICIENTS AT EACH STATE
                C         CX=REFLECTION COEFF. INTEGRATED IN SPACE & TIME
                C         DEN(1)...DEN(LCN)=  STAGE AUTOPOWER
                C         NUM(1)...NUM(LCN)= STAGE CROSSPOWER
                C
                C         CALLING 'BAFL' FIRST SETS UP THE LOOPING AND PASSING
                C         ARRAYS FOR THE PARTICULAR PROBLEM AS SPECIFIED BY LCN &
                C         IFLGAP.  THEN IT COMPUTES A SHORT (LENGTH=NWARM) ESTIMATE
                C         OF THE REFLECTION COEFFICIENT SERIES IN ORDER TO START THE
                C         ADAPTION OUT WITH SOME REASONABLE NUMBERS. THEN IT LOADS UP
                C         THE CX ARRAY WITH THE INITIAL VALUES AND PASSED INTO ENTRY
                C         'BAFLGO'.
                C
                C         THE USUAL ENTRY IS 'BAFLGO' WHICH FIRST INITIALIZES THE
                C         BACKWARD ARRAY THEN PASSES TO THE MAIN ALGORITHM.
                C         THE CX SERIES IS UPDATED EVERY IFLGAP DATA POINTS AND IN
                C         THE INTERMEDIATE STEPS THE OUTPUT ARE INTERPOLATED OR
                C         PROCESSED AS THOUGH THE R.C. WERE STATIONARY.
                C
                C
                C
                C
SN 0003                 REAL NUM(50),DEN(50),B(50),F(50),C(50),EM(1000),EP(1000),
```

```
     1  CX(ECN,LOOT),A(50)
SN 0004      COMMON /BLK3/X(1000)
   0005      DATA A,B,C/150*0.0/
SN 0006      FTEST=1.00
SN 0007      LCNP1=LCN+1
SN 0008      LCNP2=LCN+2
SN 0009      IFGM1=IFLGAP-1
SN 0010      LBSP=LOOT-IFLGAP
SN 0011      NZERP1=NZERO+1
SN 0012      NEND=LCN-NZERP1
SN 0013      DO 60 K=LBSP,LOOT
SN 0014   60 EP(K)=0.
C BEGIN STATIONARY WARM-UP
SN 0015      DO 10 I=1,NWARM
SN 0016      EM(I)=X(I*IFLGAP+ISTRT)
SN 0017   10 EP(I)=X(I*IFLGAP+ISTRT)
C
SN 0018      DO 11 J=2,LCNP1
SN 0019      DEN(J)=0.
SN 0020      NUM(J)=0.
SN 0021      DO 12 I=J,NWARM
SN 0022      DEN(J)=DEN(J)+EP(I)*EP(I)+EM(I-J+1)*EM(I-J+1)
SN 0025   12 NUM(J)=NUM(J)+EP(I)*EM(I-J+1)
SN 0024      DIV=NWARM-J+1
SN 0025      NUM(J)=NUM(J)/DIV
SN 0026      DEN(J)=DEN(J)/DIV
SN 0027      C(J)=-2.*NUM(J)/DEN(J)
SN 0028      DO 11 I=J,NWARM
SN 0029      EPI=EP(I)
SN 0030      EP(I)=EPI+C(J)*EM(I-J+1)
SN 0031      KS=IFLGAP*LCN+1
SN 0032   11 EM(I-J+1)=EM(I-J+1)+C(J)*EPI
SN 0033      DO 8 J=  1,LCN
SN 0034      DO 8 K=1,KS
SN 0035    8 CX(J,K)=C(J+1)
SN 0036      DO 33 J=1,LCN
SN 0037      DEN(J)=DEN(J+1)
SN 0038   33 NUM(J)=NUM(J+1)
C END WARM-UP CYCLE
C SET RELAXATION TIMES
SN 0039      DLX=EXP(-1./XTAU)
SN 0040      DL=EXP(-1./ZTAU)
SN 0041      DRX=1.-DLX
SN 0042      DR=1.-DL
C
C-----------------USUAL--ENTRY-------------------------------------------
SN 0043      ENTRY BAFLGG
C
C            INITIALIZE BACKWARD VECTOR
SN 0044      B(1)=X(KS-IFLGAP)
SN 0045      A(1)=1.0
SN 0046      DO 1000 J=2,LCN
             B(J)=X(KS-J*IFLGAP)
             A(J)=CX(J,KS)
SN 0049      DO 1000 I=2,J
SN 0050      A(I)=A(I)+CX(J,KS)*A(J-I+1)
```

```
0051       1000 B(J)=B(J)+A(I)*X(KS+(I-J-1)*IFLGAP)
                C
                C--------BEGIN--MAIN--LOOP-----------------------------------------
                C
0052                 DO 5000 K=KS,LOOT,IFLGAP
0053                 Z=X(K)
0054                 DO 1010 J=1,NZERP1
0055       1010 F(J)=Z
0056                 DO 2000 J=NZERP1,LCN
0057                 DEW(J)=(F(J)**2+B(J)**2)*DR+DEN(J)*DL
0058                 NUM(J)=F(J)*B(J)*DR+NUM(J)*DL
0059                 IF(FTEST.LE.1.1) CX(J,K)=-2.*NUM(J)/DEN(J)
0061                 CX(J,K)=-2.*DRX*NUM(J)/DEN(J)+CX(J,K)*DLX
0062       2000 F(J+1)=F(J)+CX(J,K)*B(J)
0063                 X(K)=F(LCNP1)
0064                 DO 3000 JR=1,NEND
0065                 J=LCN-JR
0066       3000 B(J+1)=B(J)+CX(J,K)*F(J)
0067                 IF(NZERO.EQ.0) GO TO 5000
0069                 DO 4000 JR=1,NZERO
0070                 J=NZERP1-JR
0071       4000 B(J+1)=B(J)
0072       5000 B(1)=Z
                C
                C--------END--OF--MAIN--LOOP-------------------------------------------
                C
                C        NOW GO BACK AND FILL IN THE GAPS....
 073                 IF(IFLGAP.EQ.1) GO TO 9050
0075                 DO 9000 L=1,IFGM1
0076                 KSTART=KS+L
0077                 DO 9000 K=KSTART,LOOT,IFLGAP
0078                 KC=K-L
0079                 Z=X(K)
0080                 DO 6000 J=1,NZERP1
0081       6000 F(J)=Z
0082                 DO 7000 J=NZERP1,LCN
0083       7000 F(J+1)=F(J)+CX(J,KC)*B(J)
0084                 X(K)=F(LCNP1)
0085                 DO 8000 JR=1,NEND
0086                 J=LCN-JR
0087       8000 B(J+1)=B(J)+CX(J,KC)*F(J)
0088                 IF(NZERO.EQ.0) GO TO 9000
0090                 DO 8050 JR=1,NZERO
0091                 J=NZERP1-JR
0092       8050 B(J+1)=B(J)
0093       9000 B(1)=Z
                C
0094       9050 FTEST=FTEST+1.0
0095                 RETURN
0096                 END
```

Pseudo-Wavenumber Filtering.--In some cases it may be desirable to filter in the horizontal wavenumber K_x domain. This is often of most use in separating flat-lying sea floor multiples from dipping events on the basis of dip alone. In this case, we would want to filter out the near horizontal events with some type of low-cut K_x filtering operation. Since a horizontal wave has a horizontal wavenumber (K_x) of zero, filtering with a low-cut K_x filter is analogous to removing dc to very low frequencies in the time domain. At the other extreme, we may desire to enhance lateral continuity or correlation by stacking or summing to some degree along the x or horizontal coordinate. This would be equivalent to a high-cut K_x filtering operation. To design and implement a general spatial operator to filter in the w-K_x plane would be necessarily expensive. Another approach is to design a recursive type filter operating on past spatial points exclusively. While this type of filter is not precisely a wavenumber filter, it is capable of doing the high-cut and low-cut operations mentioned above with a minimal amount of computational expense and use in the program.

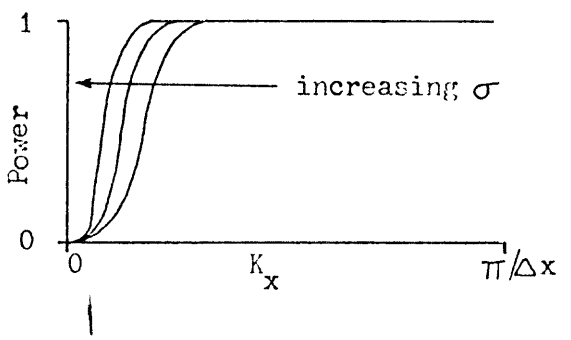
If we let $r = e^{iK_x\Delta X}$ define the trace delay operator, where K_x is the wavenumber in the x-direction and ΔX is the spacing between successive traces, this is analogous to the well-known Z transform $Z = e^{iw\Delta t}$, which is the unit-delay operator in time. A spatial low-cut filter takes the recursive form

$$\frac{1+\sigma}{2} \quad \frac{1-r}{1-\sigma r}$$

where $0 \leq \sigma < 1$. It may be verified that the power spectrum of this filter is

$$\frac{(1+2\sigma+\sigma^2)\ \mathrm{SIN}^2\ \frac{K_x\Delta X}{2}}{1+\sigma^2-2\sigma\cos K_x\Delta X}$$

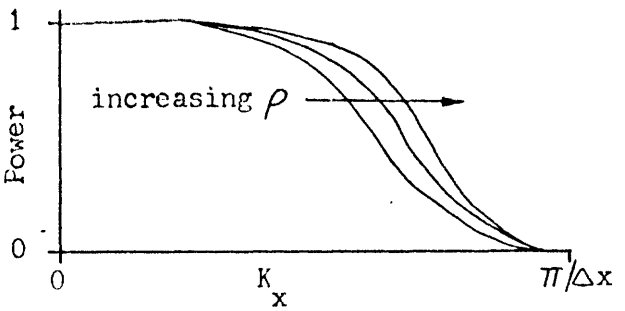
which has the form



Similarly, a high-cut filter may be derived as $(1-\rho)\ \frac{1}{(1-\rho r)}$ where $0 \leq \rho < 1$

which has the power spectrum $\frac{1-2\rho+\rho^2}{1-2\rho\cos K_x\Delta X+\rho^2}$

This takes the following form as a function of K_x

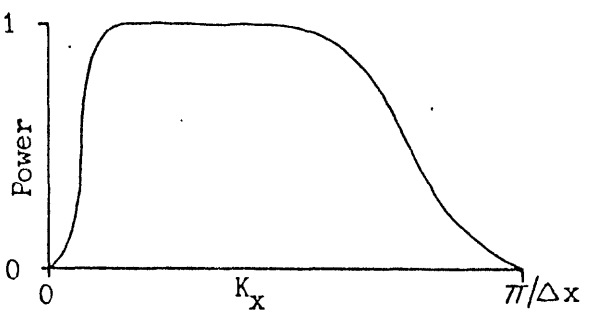


Combining these two filters we obtain a type of band-pass control for K_x filtering.

$$\frac{1+\sigma}{2}\,\frac{1-r}{1-\sigma r} \quad X \quad 1-\rho \quad \frac{1}{1-\rho r}$$

Low-cut removes near-horizontal events	High-cut stacks near-horizontal events

And in general has the form:



Note that when $\sigma = 1$ the filter is purely high-cut and acts to enhance the lateral correlation by integrating past traces by the weighting function $e^{-\rho r}$. When $\rho = 0$ the filtering is purely low-cut and acts to remove the near-horizontal energy. The wavenumber and the dip of the events are related by the long-wavelength approximation, viz. $\sin(\text{dip}) = K_x C/w$ where C is the seismic velocity and w is the frequency. Ideally we would like to determine the σ and ρ parameters knowing the high and low K_x cutoffs. Unfortunately, we need to know the frequency, velocity, and the vertical exaggeration of the display. Another approach is to simply relate the numerical values of the two filter parameters to the empirical results of using the filter. In the subroutine Bifilt which implements the above filter the parameters are specified as RHO for and SIG for σ. If we describe the removal of horizontal energy and stacking of horizontal energy as Hard, Medium, Light, or None, then the following table is meant as a guide for setting the parameters for the subroutine.

Removal of Near-Horizontal Events		Stacking in X							
		None		Light		Medium		Hard	
		RHO	SIG	RHO	SIG	RHO	SIG	RHO	SIG
	None	.00	.99	.36	.99	.53	.99	.64	.99
	Light	.00	.92	.36	.92	.53	.92	.64	.92
	Medium	.00	.81	.36	.81	.53	.81	.64	.81
	Hard	.00	.74	.36	.74	.53	.74	.64	.74

NOTE: The RHO and SIG parameters are referenced through the input parameter list as BRHO and BSIG, respectively.

```
:L 20.1 (AUG 71)                                    OS/360  FORTRAN H

      COMPILER OPTIONS - NAME= USGSC,OPT=02,LINECNT=58,SIZE=0000K,
                         SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,NOXREF
S   002           SUBROUTINE BIFILT(ND,SIG,RHO,PLOC,NPTS,SRATE2)
                C
                C                 BILINEAR KX FILTER
                C                 USEFUL FOR EXTINGUISHING HORIZONTAL PRIMARIES
                C                 AND MULTIPLES INTERFERING WITH DIPPING REFLECTORS
                C                 AND/OR FOR STACKING IN  X  TO ENHANCE LATERAL
                C                 CORRELATION IN THE PRESENCE OF ADDITIVE NOISE.
                C                 ALGORITHM OPERATES IN TIME VARYING MODE BETWEEN
                C                 PURE  KX  LOWCUT (RHO=0) AND PURE  KX  HIGHCUT
                C                 (SIG=1) AS SET IN THE RHO/SIG PROFILES.
                C
                C            PARAMETER LIST:
                C            IN      INPUT TRACE
                C            OUT     OUTPUT TRACE
                C            ND      LENGTH OF INPUT/OUTOUT TRACE
                C            SIG     SIGMA PARAMETER PROFILE
                C            RHO     RHO PARAMETER PROFILE
                C           PLOC     LOCATION POINTS OF PARAMETER CHANGES IN TIME
                C            NPTS    NO. OF SIG/RHO/LOC POINTS  MAXIMUM=10
                C
                C                 RHO                    SIGMA
                C  PLOC1----------X-------   --X--------------
                C       |          . RHO1       |   .
                C       |        .              |    .
                C  PLOC2|       XRHO2          |     XSIG2
                C       |        .              |       .
                C  PLOC3|       XRHO3          |       XSIG3
                C       |      .                |         .
                C       |    .                  |           .
                C  PLOC4|   XRHO4               |           XSIG4
                C       |                       |
                C
S   0003              REAL*4 SIG(1),RHO(1),PLOC(1),D(1000),E(1000),F(1000)
S   0004              REAL*4 IN,OUTH(1000),OUTHH(1000),INH(1000)
S   0005              INTEGER*4 LOC(10)
S   0006              COMMON /BLK3/IN(1000)
S   0007              LOC(1)=1
S   0008              DO 5 K=2,NPTS
S   0009              LOC(K)=PLOC(K)/SRATE2
S   0010              IF(LOC(K-1).GT.LOC(K)) GO TO 97
S   0012            5 CONTINUE
S   0013              LOC(NPTS)=ND
S   0014              NPM1=NPTS-1
S   0015              WRITE(6,800) RHO(1),SIG(1)
S   0016          800 FORMAT(1HO,6X,'PARAMETER PROFILE'//1H ,3X,'DEPTH',
                     1 4X,'RHO',5X,'SIGMA'//1H ,6X,'1',2(3X,F6.3))
S   0017              DO 10 IW=1,NPM1
S   0018              WRITE(6,805) LOC(IW+1),RHO(IW+1),SIG(IW+1)
S   0019          805 FORMAT(1H ,3X,I4,2(3X,F6.3))
S   0020              FMR=(RHO(IW+1)-RHO(IW))/(LOC(IW+1)-LOC(IW))
S   0021              FMS=(SIG(IW+1)-SIG(IW))/(LOC(IW+1)-LOC(IW))
    0022              NEND=LOC(IW+1) - 1
    0023              NSTRT=LOC(IW)
S   0024              IF(NSTRT.GT.NEND) GO TO 97
S   0025              DO 10 K=NSTRT,NEND
```

```
0027            R=RHO(IW) + FMR*(R-NSTRT)
0028            S=SIG(IW) + FMS*(K-NSTRT)
0029            D(K)=0.5*(1.+S)*(1.-R)
0030            E(K)=S+R
0031         10 F(K)=S*R
0032            E(ND)=SIG(NPTS)+RHO(NPTS)
0033            F(ND)=SIG(NPTS)*RHO(NPTS)
0034            D(ND)=0.5*(1.+SIG(NPTS))*(1.-RHO(NPTS))
0035            DO 20 K=1,ND
0036            OUTH(K)= 0.0
0037            INH(K)= 0.0
0038         20 OUTHH(K)=0.0
0039            RETURN
          C
0040            ENTRY BIGOT
0041            DO 70 K=1,ND
0042            TEMP=D(K)*(IN(K)-INH(K))+E(K)*OUTH(K)-F(K)*OUTHH(K)
0043            OUTHH(K)=OUTH(K)
0044            OUTH(K)=TEMP
0045            INH(K)=IN(K)
0046         70 IN(K)=TEMP
0047            RETURN
0048         97 WRITE(6,900)
0049        900 FORMAT(1HO,'***PARM.RHO/SIG ERROR***ABNEOJ***')
0050            STOP
0051            END
```

SUPPORTING SUBROUTINES

SEISGM - Variable Area Seismic Section Plotter.--Although most digital processing of seismic data can be done on nearly any available computer, effective and efficient display of such data does require special capability. The current device used is an electrostatic matrix plotter which writes a line of dots .01 inches in diameter .0138 inches apart. "SEISGM" computes an estimate of the rms value of the trace and scales the plotted trace to clip at this value.

The trace to be plotted is passed to the subroutine through the real array DATA in common block BLK2. The trace is NT samples long and is to be displayed with a maximum width of NBITS dots. The display is variable area in that dots fill in the positive side of the trace and leaves the other blank.

Before any calls to SEISGM the entry to TAXIS must first be called. This routine sets up the scaling array and initiates the writing of the time axis on the plot. Tick marks on the axis and time lines on the subsequent plot are 100 ms apart. The information required by TAXIS is SRATE the sample rate in seconds, NPAGES the number of pages to compose the section on, and RLINCH, the length of the record section in inches.

As the data is processed each finished trace is plotted via calls to SEISGM and the plot image is stored on a high-speed drum.

When all the processing has been completed, the plot of the seismic section is written to the electrostatic plotter by calling routine SDUMP. SDUMP requires no parameters.

The specific device used is a Versatec electrostatic plotter. The primary access to this unit is through calls to 'WRITER(LBUF,NBITS)' which writes a string of 'NBITS' (up to 70 x 8 bits) logical bits 'LBOF' to the plotter. Subroutine 'VLINE(LITERAL,NLIT)' is used to write a literal EBCDIC byte string for annotation purposes. This subroutine is derived from an original variable area plotting program written by Prof. Jon Claerbout of Stanford University.

```
L 20.1 (AUG 71)                          OS/360  FORTRAN H

      COMPILER OPTIONS - NAME= USOSC,OPT=02,LINECNT=58,SIZE=0000K,
                         SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,NOXREF
   002              SUBROUTINE SEISGM
                    C
                    C       SUBROUTINE 'SEISGM'   VARIABLE AREA SEISMIC SECTION PLOTTER.
                    C       THROUGH MULTIPLE CALLS TO SEISGM ADJACENT TRACES ARE WRITTEN
                    C       TO THE VERSATEC FILLING THE FULL-PAGE WIDTH (960 BITS).
                    C       DATA: INPUT DATA SERIES
                    C       NT = LENGTH OF INPUT DATA
                    C       NBITS= WIDTH OF VARIABLE AREA TRACE IN VERSATEC BITS
                    C       NPAGES: WHEN NPAGES=1 THE RECORD SECTION FILLS ONE PAGE
                    C               OF THE VERSATEC (7.536 IN.) FOR ANY LENGTH RECORDS.
                    C               WHEN NPAGES=2 A SPECIAL OPTION IS INVOKED ALLOWING
                    C               SOMEWHAT MORE GENERAL DISPLAY AS FOLLOWS.....
                    C               'RLINCH' INDICATES THE DESIRED RECORD SECTION LENGTH
                    C               IN INCHES. THUS THE RECORDS OF LENGTH NT ARE SCALED
                    C               INTO A VARIABLE OBJECT SPACE THAT MAY TAKE UP MORE
                    C               THAN ONE PAGE-WIDTH.  THIS IS ACCOMPLISHED BY COMPOSIN
                    C               THE PLOT INTO A LONGER BUFFER THAN THE VERSATEC CAN
                    C               HANDLE(WHICH IS 70 BYTES) AND DUMPING THIS PLOTTED
                    C               DATA ONTO A 2301 DRUM (FT UNIT 1) .  THEN WE REWIND AN
                    C               FETCH AND WRITE THE FIRST 70 BYTES ON ONE PAGE; THEN
                    C               GO BACK AND FETCH AND WRITE THE NEXT 70 BYTES OR
                    C               LESS.
                    C
                    C       'TAXIS' MUST BE CALLED BEFORE 1ST 'SEISGM' CALL SINCE
                    C       A REFERENCE TABLE MUST BE GENERATED.
                    C
   003              INTEGER*4 IREF(1120)
   004              LOGICAL*4 LMZ,LZ,LASK,LINE(35,10),LOAD(18),LOAD2(35),LBOT*1(72)
   005              LOGICAL*4 LTAX(35),LIGHT(35),DARK(35),LMASK(32)
   006              COMMON /BLK2/DATA(1000),NT,N5,N6,R5,R6,NBITS
   007              EQUIVALENCE (LBOT(1),LOAD2(18))
   008              DATA LMZ,LZ,LIGHT,DARK/Z80000000,36*Z00000000,35*ZFFFFFFFF/
   009              DATA LMASK/Z80000000,Z40000000,Z20000000,Z10000000,Z08000000,
                    1Z04000000,Z02000000,Z01000000,Z00800000,Z00400000,Z00200000,
                    2Z00100000,Z00080000,Z00040000,Z00020000,Z00010000,Z00008000,
                    3Z00004000,Z00002000,Z00001000,Z00000800,Z00000400,Z00000200,
                    4Z00000100,Z00000080,Z00000040,Z00000020,Z00000010,Z00000008,
                    5Z00000004,Z00000002,Z00000001/
   010              NMAG=MIN0(10,NBITS)
   011              SUM=0.
   012              DO 121 K=1,NT,4
   013              T=DATA(K)
   014          121 SUM=SUM+ABS(T)
   015              ERMS=4.*SUM/NT
                    C       WRITE(6,900) ERMS
                    C 900 FORMAT(1H+,100X,F11.1)
   016              SCALE=0.5*NMAG/ERMS
   017              IBIAS=(NMAG+1)/2
   018              DO 20 I=1,JBYTES
   019              DO 20 J=1,NMAG
                    C    SETTING = LTAX WRITES TIME MARKS ACROSS ENTIRE SECTION,LZ DOESN'T
                    C   20 LINE(I,J) = LZ
   020              20 LINE(I,J)=LTAX(I)
   021              DO 40 IBIT=1,32
```

```
0022            LASK=LMASK(IBIT)
0023            IBYTE = 1
0024            DO 40 IL=IBIT,JBITS,32
0025            IMAG=IBIAS+DATA(IREF(IL))*SCALE
0026            IF(IMAG.LE.0) GO TO 40
0028            IMAG=MIN0(NMAG,IMAG)
0029            DO 30 I=1,IMAG
0030         30 LINE(IBYTE,I) = LINE(IBYTE,I).OR.LASK
0031         40 IBYTE = IBYTE + 1
0032            DO 50 I=1,NMAG
0033         50 WRITE(1)  (LINE(K,I),K=1,JBYTES)
0034            LKNT = LKNT + NMAG
0035            RETURN
            C
            C        ENTRY 'TAXIS' WRITES A TIME SCALE TO THE VERSATEC.
            C        SRATE IS THE SAMPLING RATE.  TICK MARKS ARE AT 100 MILS.
            C
0036            ENTRY TAXIS(NT,SRATE,NPAGES,RLINCH)
0037            JBYTES=18
0038            JBITS=560
0039            IF(NPAGES.LE.1) GO TO 55
0041            JBYTES=35
0042            JBITS=MIN1(1120.,RLINCH*73.25)
0045         55 SCALE=(JBITS-1)/(NT*SRATE)
0044            ITMAX=SRATE*NT*10. + 1.
0045            TYME=0.
0046            DO 60 I=1,JBYTES
   47         60 LTAX(I)=LZ
0048            DO 70 I=1,ITMAX
0049            IR=SCALE*TYME + 1.0
0050            IBYTE=(IR-1)/32  + 1
0051            IBIT=IR - (IBYTE-1)*32
0052            LASK=LMASK(IBIT)
0053            LTAX(IBYTE)=LTAX(IBYTE).OR.LASK
0054         70 TYME=TYME + 0.1
0055            SCALE=(NT-1.)/(JBITS-1.)
0056            DO 90 IL=1,JBITS
0057         90 IREF(IL)=SCALE*(IL-1) + 1.49999
0058            DO 83 I=1,8
0059         82 WRITE(1) LIGHT
0060            WRITE(1) DARK
0061            DO 84 I=1,10
0062         84 WRITE(1) LTAX
0063            WRITE(1) DARK
0064            DO 81 I=1,10
0065         81 WRITE(1) LIGHT
0066            LKNT=30
            C
0067            RETURN
            C
            C        ENTRY SDUMP DUMPS THE DRUM TO THE VERSATEC
            C
   068          ENTRY SDUMP
   069          DO 85 I=1,330
0070         85 WRITE(1) LIGHT
0071            REWIND 1
```

```
0072            NL=LKNT + 378
0073            NFIN=NL
0074            IF(NPAGES.GT.1) NFIN=NL-340
0076            DO 82 I=1,NFIN
0077            READ(1) LOAD
0078         82 CALL WRITER(LOAD,70)
0079            IF(NPAGES.LE.1) GO TO 87
0081            REWIND 1
0082            DO 86 I=1,NL
0083            READ(1) LOAD2
0084         86 CALL WRITER(LBOT(3),70)
0085         87 CONTINUE
0086            RETURN
0087            END
```

AGC - Digital Automatic Gain Control.--Subroutine AGC rescales the power on a trace to have a constant level. It computes an estimate of the rough rms value of the trace as a function of time. The diffusion equation is applied to the rough rms to obtain a smoothed rms gain trace. The input data is then divided by the smoothed gain to yield a constant power output trace with no phase distortion. The input data is passed through TRACE in common block BLK2 and is returned in the same array. N is the length of the input/output array. NJUMP is the design frequency, i.e., for NJUMP=24, the gain trace is recomputed every 24 traces. NLEAK controls the degree of smoothing of the rough rms trace. A value of NLEAK of about 10 to 20 gives moderately fast gain recovery. This program was written by Prof. Jon Claerbout of Stanford University.

(NJUMP in main PGM = JUMPX in the subroutine.)

```
      SUBROUTINE AGC
      DIMENSION ARMS(1000)
      COMMON RMS(1000),SRMS(1000)
      COMMON /BLK2/TRACE(1000),N,JUMPX,NLEAK
      DATA IX/0/
      IF(MOD(IX,JUMPX).NE.0) GO TO 50
      B=(1.+2.*NLEAK*NLEAK)*1024.
      A=-NLEAK*NLEAK*1024.
      DO 10 I=1,N
      Q=TRACE(I)
   10 RMS(I)=ABS(Q)
      CALL TRI(A,B,A,N,SRMS,RMS,SRMS,RMS)
      IF(IX.NE.0) GO TO 30
      DO 20 I=1,N
   20 ARMS(I)=SRMS(I)
   30 DO 40 I=1,N
   40 ARMS(I)=.8*ARMS(I)+.2*SRMS(I)
C     WRITE(6,77)(ARMS(I),I=1,500)
C77  FORMAT(' GAIN CONTROL'/(10F12.8))
   50 DO 60 I=1,N
   60 TRACE(I)=TRACE(I)/ARMS(I)
      IX=IX+1
      RETURN
      END

      SUBROUTINE TRI(A,B,C,N,T,D,E,F)
      DIMENSION T(N),D(N),F(N),E(N)
      N1=N-1
      E(1)=1.0
      F(1)=0.
      DO 10 I=2,N1
      DEN=B+C*E(I-1)
      E(I)=-A/DEN
   10 F(I)=(D(I)-C*F(I-1))/DEN
      T(N)=F(N1)/(1.0-E(N1))
      DO 20 J=1,N1
      I=N-J
   20 T(I)=E(I)*T(I+1)+F(I)
      RETURN
      END
```

BPASS - Bandpass filter (convolution type).--Calling subroutine GFILT generates the filter coefficients for two different bandpass filters. SRATE is the input sample rate in seconds. LX is the length out the input/output arrays (X,Y). The low and high cutoff points in Hz is specified by the arguments FLOW and FHIGH for the first filter and FLOW2 and FHIGH2 for the second filter. Entry point BPASS filters the input trace Y and outputs the filtered trace X (using FLOW, FHIGH filter). Entry point BPASS2 filters the input trace X and outputs the filtered trace Y (using the FLOW2, FHIGH2 cutoff filter).

BPASS: Y(IN) X(OUT) filter: FLOW, FHIGH
BPASS2: X(IN) Y(OUT) filter: FLOW2, FHIGH2

```
      SUBROUTINE GFILT(FLOW,FHIGH,SRATE,LFILT,FLOW2,FHIGH2,LX)
      REAL*4 FILT(20),FILT2(20),X(1,1000),Y(1,1000)
      COMMON /BLK2/Y
      COMMON /BLK3/X
      BW=FHIGH-FLOW
      AS=BW*SRATE*3.1415927
      AC=SRATE*2.*3.1415927*(FHIGH-BW/2.)
      M=LFILT/2
      MID=M+1
      DO 10 I=1,M
      FILT(MID-I)=COS(AC*I)*SIN(AS*I)/(AS*I)
   10 FILT(MID+I)=FILT(MID-I)
      FILT(MID)=1.0
      BW=FHIGH2-FLOW2
      AS=BW*SRATE*3.1415927
      AC=SRATE*2.*3.1415927*(FHIGH2-BW/2.)
      DO 15 I=1,M
      FILT2(MID-I)=COS(AC*I)*SIN(AS*I)/(AS*I)
   15 FILT2(MID+I)=FILT2(MID-I)
      FILT2(MID)=1.0
      NEND=LX-LFILT+1
      RETURN
      ENTRY BPASS
C     Y(IN) X(OUT)
      DO 20 I=1,LX
   20 X(I,1)=0.
      DO 30 I=1,NEND
      DO 30 J=1,LFILT
   30 X(I,J)=X(I,J)+Y(I+M,1)*FILT(J)
      RETURN
      ENTRY BPASS2
C     X(IN) Y(OUT)
      DO 40 I=1,LX
   40 Y(I,1)=0.
      DO 50 I=1,NEND
      DO 50 J=1,LFILT
   50 Y(I,J)=Y(I,J)+X(I+M,1)*FILT2(J)
      RETURN
      END
```

Decoding subroutines.--The primary function of the decoding subroutines is to convert the input binary data from the tape into floating-point representation. In addition, a low-cut filter is applied to cancel out the very long period (less than 10 hz) noise associated with the propeller and other ship noise. This is the simple and cheap feedback filter $\frac{1+\rho}{2}\ \frac{1-Z}{1-\rho Z}$

where Z is the unit delay operator. The parameter is determined from the desired cutoff frequency f_c (hz) and the sampling rate Δt (sec) in the expression: ρ = secant $(2\pi f_c \Delta t)$ - tan $(2\pi f_c \Delta t)$ as the -3 db cutoff point in the filter response. The default value of the list parameter RHO is .74 corresponding to a cutoff around 15 hz.

DCDEC decodes the reformatted 9-track 800 bpi reels. This is the simplest routine since the original data are written as 16 bit two's complement words.

```
      SUBROUTINE DCDEC(IN,NDEC)
      INTEGER*2 IN(NDEC,1)
      COMMON /BLK2/OUT(800),LOUT,N1,N2,RHO,CRHO
      OUT(1)=IN(NDEC,1)+100
      DO 1000 K=2,LOUT
      KM1=K-1
 1000 OUT(K)=CRHO*(IN(NDEC,K)-IN(NDEC,KM1))+RHO*OUT(KM1)
      RETURN
      END
```

DCDECT decodes the Teledyne 27110 7-track 1971 field reels. The Teledyne recorder format requires a significant amount of bit shifting in order to arrive at a representation acceptable to the 360.

```
      SUBROUTINE DCDECT (BUFR)
C         ********** SUBROUTINE DCDECT **********
C         THIS ROUTINE DOES TWO THINGS: FIRST IT CHANGES THE SIGN
C         CONVENTION OF THE TELEDYNE 27110 TO IBM 360 COMPATIBLE AND
C         SHIFTS OUT THE LOW ORDER G(8) GAIN BIT ; THEN IT ALSO GOES
C         AHEAD AND DOES SOME LOW-PASS FILTERING AS SPECIFIED BY RHO IN
C         THE FILTER : (1+RHO)/2 * (1 - Z)/(1 - RHO*Z)
C
      INTEGER*2 BUFR(4000)
      COMMON/BLK2/OUT(1000),LX,N1,N2,RHO,CRHO
      DATA NSHIFT,NBIG,NSHF2/Z00010000,Z8000C000,Z0002000C/
      INEW=(BUFR(13)*NSHIFT+NBIG)/NSHF2
      OUT(1)=INEW
      IBYTE=16
      DO 333 K=2,LX
      IOLD=INEW
      INEW=(BUFR(IBYTE)*NSHIFT+NBIG)/NSHF2
      OUT(K)=CRHO*(INEW-IOLD)+RHO*OUT(K-1)
  333 IBYTE=IBYTE+3
      RETURN
      END
```

DCDECN decodes the NCER analog/digital tapes of the 1972 season. It was necessary to code the routine in 360 Assembler due to the complexity of the unpacking routine. This particular decoding subroutine does no low-cut filtering on the data.

```
//ASEMBL JOB (C740,314,1.0),'RILEY'
//HUMP EXEC ASMGC,PARM='DECK,NOLOAD'
//ASM.SYSIN DD *
GO        START 0
          ENTRY DCDECN
          USING *,15
DCDECN    SAVE  (14,12),,*
*********************************************************************
*
*   SUBROUTINE DCDECN:  FORTRAN CALLABLE
*      CALL DCDECN(BUF,LEN,OUT)
*
*      WHERE  BUF IS THE INPUT BUFFER INTEGER*2 THAT
*                 COMES OUT OF 'NOERRD' OR 'RDING' AND
*                 IS THE PACKED (CONVERTED) NCER 7 TRK
*                 TWO CHANNEL DATA.
*             LEN IS THE DESIRED LENGTH OF THE OUTPUT SERIES
*             OUT IS THE FLOATING POINT OUTPUT ARRAY
*    THIS ROUTINE COMPUTES THE SUM OF THE TWO CHANNELS
*     AND CONVERTS TO FLOATING-POINT AND PUTS IT IN OUT
*
*********************************************************************
          L     7,8(0,1)   ADDR. OF OUT
          LM    10,11,0(1)
          L     11,0(0,11)
          SEL   11,1
LOOP1     LH    8,0(,10)
          SLL   8,8
          IC    8,2(,10)
          SRDA  8,12
          SRA   9,20
*         AR    8,9     CHANNEL 1 + 2 SUM
          LD    4,CON80
          STD   4,CON90
          AR    8,9
          BM    NEG
          ST    8,CON94
          AD    4,CON90
          B     DONE
NEG       LPR   0,8
          ST    0,CON94
          SD    4,CON90
DONE      STE   4,0(,7)     STORE SAMPLE
          LA    7,4(0,7)
```

```
         LH     9,4(,10)
         SLL    9,16
         IC     8,3(,10)
         SLDL   8,4
         SRA    9,20
         SLL    8,20
         SRA    8,20
         LD     4,CON80
         STD    4,CON90
         AR     8,9
         BM     NEG2
         ST     8,CON94
         AD     4,CON90
         B      DONE2
NEG2     LPR    0,8
         ST     0,CON94
         SD     4,CON90
DONE2    STE    4,0(,7)   STORE SAMPLE
         LA     10,6(0,10)
         LA     7,4(0,7)
         BCT    11,LOOP1
         LE     4,ZERO
         STE    4,0(,7)
         RETURN (14,12),T
         DS     0D
CON80    DC     X'4E000000'
CON84    DC     X'00000080'
CON90    DS     F
CON94    DS     F
ZERO     DC     E'0'
         END
/*
BESD       -    AGO                *DCDECN  A       A
BTXT       8    AGOO<FDCDECN &%&<QO&HQ,& QOO HO AH - I  HC -B+  < & MY OH
BTXT   8   8    A- D& IG 0+& 0M  O&GOO&H& 0M, 0&O O AOO&H&-DI& &C -C(  D
BTXT   0   8    A & MI  M   MY OH- 0& IG OO& 0M  O&GOO-&H& 0M, 0&O 0 A--F
BTXT   Y        AAOO&FOO-& OOO O O%&<K"&<G=
BTXT   H   H    A+
BTXT   Q   D    A
BEND                                    ASMG 20SEP71  11:11:22 11 FEB 73
```

INPUT PARAMETERS FOR PROCESSING CONTROL

Description with default values

PARAMETER (DEFAULT VALUE) BRIEF DESCRIPTION

Underlined parameters should always be set by user according to job specifications. Where more than one default value is given the order is: package C, CF, CN. A'NU' default indicates that this parameter is not used in that package.

USER ('U.S.G.S.') 8 Alphanumeric characters specifying the person running the job. This will appear on the header identification of the output plot.

NREEL (9999) An integer number specifying the U.S.G.S. reel number appearing on the tape.

LINEID ('DEFERRED') 8 Alphanumeric characters specifying the profile line identification.

RANGE (400.) Real variable indicating the shot to receiver offset spacing in feet.

WVEL (4875.) Real variable specifying the water velocity in feet/second.

SRATE (.002),(.004), (.004) Sample rate in seconds at which the data was recorded or which currently exists on the data tape.

RECEND (1.53) The time in seconds to which the data will be processed, i.e. to only process the first 3.0 sec. of the data specify RECEND=3.0.

NDEC (2), (1), (NU) An integer specifying the degree of decimation desired. For example, with NDEC=2 the input sample rate, say 2 msec., will be decimated to 4 msec. as the effective sample rate during processing.

ITIME (540) Integer number of seconds that the program is allowed to run. This is an internal timer which reads the CPU clock and does not include I/O wait. This should be set at about 75% of the total job estimated time on the job card. It insures against loss of plot.

RHO (0.74), (.74), (NU) The parameter that controls the low-side cutoff of the DC reject filter. A more complete description appears in the text.

NBITS (5) An integer indicating the number of small electro-static 'dots' used to comprise the trace width on the output section. This is an important parameter that greatly affects the appearance of the processed data. It controls the amount of amplitude information and the vertical exaggeration of the plot. There are about 73.5 dots per inch along both the time and space axes.

NBITS2 (4) An integer similar to NBITS representing the secondary bit or trace width. This parameter is used in conjunction with MODADD to allow more flexibility in specifying the length of the output section.

MODADD (9) An integer value indicating when the secondary trace width NBITS2 is to be used. The trace will be NBITS2 dots wide every MODADD traces, otherwise it will be NBITS dots wide. For example, setting NBITS=4, and NBITS2=5, MODADD=30, then the trace width will be normally 4 dots, except every 30th trace when it will be 5 dots.

NOTR (2000) Integer number of traces to be processed, including skipped traces.

NOTRJ (3), (3), (2) Integer number either 1, 2, or 3 indicating the processing frequency. For NOTRJ=1 every trace will be processed; NOTRJ=2 every other trace will be processed.

NSTR (1) Integer number representing the starting trace on the tape with which processing is to start. For example, assume a tape has 1980 traces on it and it is desired to process only the last half of the tape, then setting NSTR=1980/2=990 and NOTR= 990 the program will start with the 990th sequential record or trace on the tape and end 990 traces beyond. To process the entire tape, set NSTR=1 (default) and NOTR=1980.

NJOB (3) An integer 1, 2, 3, or 4 which selects the processing sequence desired for the job. Within each of the several programs there are four options (for the sequences that are available see the individual program description) however, in each NJOB=4 is the playback only routine, i.e. no processing only plots the raw data.

TAPEDD ('FT20F001') 8 Alphanumeric characters of the form FTnnF001 which is the tape data definition tag 'DD' which points to the tape to be processed. nn is any integer 10 to 99 (except 44) which must agree with the job control statement. This permits multiprocessing of several tapes (profiles) within the same job. Use of this parameter will be illustrated in the job examples.

RLINCH (7.63) Real variable specifying the length, in inches, of the output section (time axis) that is desired. (current maximum is 15.25 inches).

The following parameters are used in processing sequences NJOB= 1, 2, and 3.

NLEAK (10), (NU), (NU) An integer intergration parameter used in the digital automatic gain control routine (AGC). This specifies the response time of the control adjustments. The default value is for rapid gain adjustments and has proved adequate for most profiles.

NJUMP (24), (NU), (NU) An integer setting the update frequency for the AGC program. It has been found that the gain control trace need not be recomputed for each trace. For example, setting NJUMP=24 the gain trace is updated every 24 traces.

LENBP (9) Integer number of coefficients in the bandpass filters. The higher the number of coefficients the higher the quality of the filter in terms of cutoff sharpness and side lobes. Filter lengths of 9 to 13 have been found to be adequate for most uses.

FHIGH (63.) Real variable indicating the desired high side cutoff in cycles/second of the filter applied before other processing.

FLOW2 (15.) Real variable of the low side cutoff of the noise suppression bandpass filter that is applied after processing and just before display.

FHIGH2 (63.) High side cutoff of the pre-display filter in cycles/second.

The following Parameters are used in processing sequences NJOB=1 and 3

NPTS (2) Integer number of points on the 'parameter 'profile' used by routine 'Bifilt'. (see Bifilt description).

PLOC (0., 1.53) A real array indicating the location of the parameter profile 'ties' for Bifilt (see description).

BRHO (.5, .5) A real array indicating the rho coefficients corresponding to the PLOC tie points. The values for the coefficients are determined according to the degree of K_x wavenumber attenuation desired. See the Bifilt routine description for explanation.

BSIG (.99, .99) Real array specifying the sigma parameter profile for the Bifilt routine. This variable is complementary to BRHO and is described in detail under the routine description.

The following parameters are used in the C series programs exclusively and only in job sequences NJOB=2 and 3.

NZC (1), (0), (0) Integer number of leading zero reflection coefficients. In the adaptive deconvolution program the filtering is characterized by equivalent reflection coefficients in an ideal layered media model.

IFLGAP (2) Integer gap interval between adaptive filter coefficients. This has the effect of decimating the filter to only operate on part of the spectrum. For example, setting IFLGAP=i the spectrum will be whitened from 0 to f_n where f is the folding frequency. With IFLGAP=2 0 to fn/2; IFLGAP=3 0 to fn/3.

LCN (9) Integer number of reflection coefficients in the adaptive prediction error operator. Maximum allowed is 20.

DWARM (.2) Real variable representing the duration of the warm up cycle performed in the first trace to initialize the adaptive filter.

WSTRT (.2) Real variable indicating the starting position in seconds of the warm up cycle.

ZTAU (.12) Real variable for the relaxation time in seconds of the time adaption rate. This is the time to 1/e relaxation of past time statistics.

XTAU (10.) Real variable indicating the relaxation distance to 1/e along the spatial coordinate of the time-space weighting function.

Coding input parameters

1. All coding begins in card column 2 and may be continued to the end of the card.

2. The start of the parameter list must begin with &JØBIN and the end of the list is marked by &END.

3. Parameters are specified by parmname=data for integer and real (decimal) variables and by parmname='char' for alphanumeric character data. Parameters are separated by commas with no intervening blanks.

4. Parameters not coded will assume their default values as designated on the parameter description table.

5. The order that the parameters appear is arbitrary,

Examples

Card Col. 2

```
&JOBIN NREEL=114,LINEID=' 144-145',RECEND=2.5,ITIME=600,RHO=.7,
NBITS=3,MODADD=9999,NOTRJ=3,NJOB=3,USER='DC RILEY',RLINCH=11.82,FHIGH=63.,
NZC=0,FHIGH2=70.,FLOW2=16.,LENBP=11,IFLGAP=2,LCN=10,ZTAU=.20,XTAU=5.0,
NSTR=2,NPTS=7,PLOC=0.,.2,.3,.5,1.,2.,2.5,BSIG=.8,.9,.999,.999,.999,.999
BRHO=.05,.1,.16,.18,.26,.36,.4,WSTRT=0.3,NOTR=1925,&END
```

Card Col. 2

```
&JOBIN NREEL=213,LINEID=' 174-175',RANGE=300.,RECEND=3.0,USER='DC RILEY',
NPAGES=2,RLINCH=15.,ITIME=600,NOTR=1992,FLOW=6.,FHIGH=65.,FLOW2=16.,FHIGH2=70.,
LCN=3,NPTS=5,PLOC=0.,.4,1.,2.,3.,BSIG=.8,.98,.99,.99,.99,BRHO=.1,.2,.4,.5,.6,
ZTAU=.15,XTAU=6.,&END
```

Card Col. 2

```
&JOBIN NREEL=044,LINEID=' 223-224',RECEND=2.5,ITIME=600,RHO=.7,
NBITS=3,MODADD=9999,NOTRJ=2,NJOB=3,USER='DC RILEY',RLINCH=11.82,FHIGH=63.,
NZC=0,FHIGH2=70.,FLOW2=16.,LENBP=11,IFLGAP=2,LCN=10,ZTAU=.20,XTAU=6.0,
NSTR=2,NPTS=7,PLOC=0.,.2,.3,.5,1.,2.,2.5,BSIG=.8,.9,.999,.999,.999,.999,.999,
BRHO=.05,.1,.16,.18,.26,.36,.4,WSTRT=0.3,NOTR=0300,&END
```

JOB control (JCL) - Stanford Computation Center (SCC)

1. Convention of statement description

   a. Lower-case letters indicate that the information is to be supplied by the user.

   b. Underlining indicates that the information is optional.

   c. All coding starts in card column 1 and ends before card column 72 except as noted.

   d. Blanks are to be included or excluded as given in the examples since they act as delimiters.

   e. The distinction between zeros and letter Ø is important and will be denoted by 0 and Ø respectively.

2. JØB statement

   Form:

   //aaaaaaaa JØB (bbbb,ccc,ddd.d,eeee), 'name'

   where aaaaaaaa is the jobname, 1-8 alphanumerics.
   bbbb is the account number
   ccc is the bin number
   ddd.d is the job time estimate in minutes
   eeee is the line estimate in thousands of lines.
   name is the user's name, up to 20 characters allowed

3. KEY statement

   Form:

   /* KEY kkk

   where kkk is the account keyword as set by the user.

4. SERVICE statement

   Form:

   /* SERVICE CLASS=class

   where class if B for using the daytime batch partition
   or Ø for the overnite partition

5. SETUP statement

The setup statement is used to place a job in hold status and tell the operator which tape to mount on which tape drive before releasing the job for execution.

Form:

```
/* SETUP T,'  message  '
```

where message may be for example:
'MOUNT TAPE scc (READ) ØN OCa, name, acc, jobname'
where scc is the comp. center tape id like U1577
OCa is tape drive, use OC1 for 7 track tapes and OC3 for 9 track tapes.
name is user's name
acc is account number
jobname is the jobname given on the JØB card

6. Other statements

Several other statements are necessary for the successful execution of the processing job such as the EXEC, JØBLIB, and DD statements. They will be introduced through examples rather than in their general form since we will be concerned with only a few of the many possible forms.

TAPE DATA DEFINITION (DD)

The function of the DD statements is to inform the computer as to where a data set (input) resides or to where a data set (output) is to be placed. It also provides information about the data set. For example, for a tape data set we need to inform the operating system the name of the data set (DSN=), the tape reel number (VØL=SER=), the disposition of the data set (DISP=), what drive to mount the tape on (UNIT=), label information (LABEL=), recording density (DEN=), and the recording format (RECFM= and TRTCH=). These requirements may seem annoying at first, but as we shall see, it is this type of control structure which will allow us to process a wide variety of data recorded by several different machines on the 360.

Example (1) We wish to process a tape recorded on the 27110 Teledyne digital recorder (1971 field tapes). Density=556 bpi, 7 track, Tapedd= FT20F001 (default), SCC identification is tape no. U1577. The appropriate DD statement is:

```
//FT20F001 DD DSN=SEIS,VØL=SER=U1577,DISP=(ØLD,KEEP),UNIT=OC1,
// LABEL=(1,BLP,,IN),UNIT=OC1,DCB=(RECFM=U,BLKSIZE=8000,DEN=1,TRTCH=C)
```

Example (2) We wish to process 1972 analog/digital tapes created on the N.C.E.R. CDC 1700. Density=556 bpi., 7 track, Tapedd=FT35F001, SCC id = U1823. The appropriate DD statement is:

```
//FT35F001 DD DSN=SEIS,VØL=SER=U1823,DISP=(ØLD,KEEP),UNIT=OC1,
// LABEL=(1,BLP,,IN),DCB=(RECFM=U,BLKSIZE=8000,DEN=1,TRTCH=C)
```

Example (3) We wish to process a reformatted 1971 tape produced by Teledyne. Density=800 bpi, tapedd = FT20F001, SCC ID = U1980. The DD is:

```
//FT20F001 DD DSN=SEIS,VØL=SER=U1980,DISP=(ØLD,KEEP),UNIT=OC3,
// LABEL=(2,BLP,,IN),DCB=(RECFM=U,BLKSIZE=8000,EROPT=SKP)
```

JØB streams

There are three primary JOB streams (sequences of cards in the deck) that are necessary for using the Seismic Processing Package. The first is concerned with compiling the SØURCE fortran programs into something called a LØAD MØDULE. The SØURCE program is written in a high-level programming language called fortran which makes it easy for the programmer to communicate his ideas with the computer. This in turn is CØMPILED or translated by the computer into a form with which it can directly deal with, namely machine instructions or machine code. These machine instructions together with some control text is what comprises the LØAD MØDULE. We shall not be concerned at all with what the load module looks like, or even with creating new modules (excepting when minor modifications to the SØURCE program are necessary).

CØMPILING SØURCE DECKS AND CREATING LØAD MØDULES

```
      JØB CARD GØES HERE
//     KEY CARD
//FIRST EXEC FØRTHCL,PARM.FØRT='LINECNT=58,ØPT=2,NAME=USGSC'
//FØRT.SYSIN DD *
           |
           |    FØRTRAN
           |         SOURCE
           |              DECK
           |
/*
//LKED.SYSLMØD DD DSN=C740.USGS(USGSC),DISP=(NEW,KEEP),
//   UNIT=2314,VØL=SER=SYS07,SPACE=(TRK,(14,1,1),RLSE)
//LKED.SYSLIB DD DSN=SYS1.FØRTLIS,DISP=SHR
//            DD DSN=SYS2.SSPLIB,DISP=SHR
//            DD DSN=SYS2.SUBLIB1,DISP=SHR
//            DD DSN=SYS2.VERSLIB,DISP=SHR
/*
//LKED.SYSIN DD *
           |
           |    ØBJECT
           |         DECKS  (IF ANY)
           |
/*
```

Moving LØAD MØDULES from disk to tape and tape to disk

Since LØAD MØDULES are fairly large data sets we may not wish to keep a copy of all our different LØAD MØDULES on disk all the time to avoid excessive disk storage charges. We may wish to create a tape containing all our programs instead. Then as we desire to process one type of tape (we have 3 types at present) we would place that LØAD MØDULE on the disk and process several jobs using that program. (recall that the operating system can only FETCH a program from a disk file).

1. Copying a LOAD MODULE from disk to tape

   Assume that we already have created a LØAD MØDULE named C740.USGS on SYS07 (as in the SØURCE example). We wish to copy this onto a tape no. U1577.

```
    JØB CARD GØES HERE
//   KEY CARD
/* SERVICE EXEC=IDLE
/* SETUP T,'MØUNT U1577 (WRITE ENABLE) ØN 9TRK,RILEY,DISKMØVE,C740'
//MØVE EXEC DSGET
//GØ.SYSIN DD *
 CØPY PDS=C740.USGS,FRØM=2314=SYS07,TØ=2400=(DSSAVE,3)
/*
```

This job will copy the data set named C740.USGS residing on SYS07 to the 3rd file on a standard label tape (label=DSSAVE) no. U1577.

To add an additional LØAD MØDULE to this tape we would execute.

```
// JØB card goes here
/* KEY card
/* SERVICE EXEC=I
/* SETUP T,'MØUNT U1577 (WRITE ENABLE) ØN 9TRK,RILEY, jobname,C740'
//MØVE EXEC DSGET
//GØ,SYSIN DD *
/*CØPY PDS=C740.USGS,FRØM=2314=SYS08,TØ=2400=(DSSSAVE,4)
```

This job would copy the file at C740.USGS residing on disk SYS08 and place it on the 4th file of the standard labeled tape (label=DSSAVE) no. U1577.

Moving the LØAD MØDULE on tape back onto the disk in preparation for x processing a tape

Let's assume that we have previously created a tape with all our load modules on it. Let's also assume that we have several tapes to process that were recorded on the Teledyne 27110 (1971 field tapes) We need to use Package CF. If this is located at the 1st file of our library tape we can move it back onto the disk (say, SYS07) by the following job:

```
//JØB card goes here
/* KEY key card
/* SETUP T,'MØUNT TAPE U1577 (READ) ØN 9TRK,RILEY,jobname,C740'
//PUTØN EXEC DSGET
//GØ.SYSIN DD *
//CØPY PDS=C740.USGS,FRØM=2400=(DSSAVE,1),TO=2314=SYS07
```

Executing LØAD MØDULES residing on a disk file

Assume that we already have a load module named C740.USGS on disk volume SYS07. To execute this program is a simple task and we only need to inform the operating system where it is via the JØBLIB statement.

//JOBLIB statement is place immediately preceeding the execute (EXEC) statement in the job stream.

1. Example (1)

   The following example illustrates the typical job stream for executing the Seismic Processing Package when the LØAD MØDULE has been placed on a disk storage device (in this example it is on disk pack SYS07 and is named C740.USGS). The numbers appearing to the left of each statement is not part of the statement. Note that all coding of the cards begins in card column 1 except for the input parameter cards which begin in card col. 2.

```
d ┌──CC1
  ▼
1 //USGSR213 JØB (C740,314,5.0,9),'USGS.RILEY.R213'
2 /* KEY ABC
3 /* SERVICE CLASS=Ø
4 /* SETUP T,'PLS MØUNT TAPE 01520 (READ) ØN 0C3,RILEY,C740,USGSR213,TNX'
5 //JØBLIB DD DSN=C740.USGS,VØL=SER=SYS07,UNIT=2314,DISP=(ØLD,PASS)
6 //SEISGØ EXEC PGM=USGSC
7 //FT01F001 DD UNIT=2314,VØL=SER=SYS03,SPACE=(CYL,(3,2)),
8 //   DCB=(RECFM=VSB,BLKSIZE=7204,LRECL=144)
9 //FT05F001 DD DDNAME=INPUT
0 //FT06F001 DD SYSØUT=A
1 //FT07F001 DD SYSØUT=B
2 //DISPLAY DD UNIT=004
3 //FT20F001 DD DSN=CHUKCHI,VØL=SER=01520,DISP=(ØLD,KEEP),UNIT=003,
4 //      LABEL=(2,BLP,,IN),DCB=(RECFM=U,BLKSIZE=~~32000~~ 8000,ERØPT=SKP)
5 //INPUT DD *
6  &JØBIN NREEL=215,LINEID=' 174-175',RANGE=300.,RECEND=3.0,USER='DC RILEY',
7  NPAGES=2,RLINCH=15.,ITIME=600,NØTR=1992,FLØW=6.,FHIGH=66.,FLØW2=16.,FHIGH2=70
8  LCN=8,NPTS=5,PLØC=0.,.4,1.,2.,3.,BSIG=.8,.98,.99,.99,.99,BRHØ=.1,.2,.4,.5,.6,
9  ZTAU=.15,XTAU=6.,&END
0 /*
```

Discussion of the example

Card 1 This is the usual JØB card and it specifies in this case that the time limit on the job is 5.0 minutes, the output is to be placed in bin 314, and that the job is named USGSR213. (A good practice is to indicate in the jobname what tape is being processed, in this case we're doing reel 213).

Card 2 This is the usual key card indicating that the user has set the keyword ABC for account C740.

Card 3 This SERVICE card indicates that the job is to be run in the ØVERNITE partition ($7/minute). To run the job in the daytime BATCH partition either leave the service card out or specify CLASS=B. ($9/minute).

Card 4 The setup card informs the operator what tape(s) to mount for our job. Translated the message reads: Please mount tape number U1520 on tape drive OC3 (9-track drive), we're going to read from this tape only, my name is Riley, my account is C740, the jobname is USGSR213, thanks.

Card 5 The JØBLIB card is a message to the operating system that it should look in a particular place for the program we're going to call up. In this example the program is named C740.USGS (as usual) and is residing on disk pack SYSO7, and it's already there (ØLD), and PASS it on to the next job step.

Card 6 This is where we actually call our program named USGSC which is a member of C740.USGS (in fact it's the only member). Now the operating system passes control to our program and away we go. That is, this statement says: in this step called SEISGO we want to EXECecute our ProGraM called USGSC.

Card 7 The next 6 cards will always be the same for all our jobs. They point to system devices that will be needed for the job. Cards 7&8 point to a disk file where our output seismogram will be written until the processing is finished.

Card 9 Card 9 points to the input stream where the input parameters will be found.

Cards 10&11 Point to the line printer and card punch, resp.

Card 12 This references the Versatec electrostatic plotter upon which the output seismograms will be written.

Card 13 &14 These cards are the usual DD data definition statements. This points to the input tape which is to be processed. The tag FT20F001 is the default TAPEDD parameter. This statement indicates the following: The tape mounted is reel U1520 (same as in the setup statement). It is mounted on tape drive 0C3 (9 track). The LABEL and DCB parameters are those for the reformatted 1971 9 track tapes (see the discussion on the DD statement for the other two forms).

Card 15 This card always looks like this. It backward references the FT05F001 telling the system where the input parameters are. The input to the processing program(the parameter list) must immediately follow this card.

Card 16 -19 Following the INPUT card we write the desired parameters that control the processing of the tape. Note that all coding must start in card column 2.

Card 20 This is the last card in the deck and signals the operating system of the end of the job stream.

Other examples

```
     JOB CARD GOES HERE
//    KEY CARD
/* SERVICE CLASS=D
/* SETUP 1,'PLS MOUNT TAPE U1577 (READ) ON 0C1,RILEY,C740,USGSR114,TNX'
//JOBLIB DD DSN=C740.USGS,VOL=SER=SYS07,UNIT=2314,DISP=(OLD,PASS)
//SEISGO EXEC PGM=USGSC
//FT01F001 DD UNIT=2314,VOL=SER=SYS03,SPACE=(CYL,(3,2)),
//  DCB=(RECFM=VSB,BLKSIZE=7204,LRECL=144)
//FT05F001 DD DDNAME=INPUT
//FT06F001 DD SYSOUT=A
//FT07F001 DD SYSOUT=B
//DISPLAY DD UNIT=0C4
//FT20F001 DD DSN=SEIS,VOL=SER=U1577,DISP=(OLD,KEEP),UNIT=0C1,
//  LABEL=(1,BLP,,IN),DCB=(RECFM=U,BLKSIZE=8000,DEN=1,TRTCH=C)
//INPUT DD *
 &JOBIN NREEL=114,LINEID=' 144-145',RECEND=2.5,ITIME=600,RHO=.7,
 NBITS=3,MODADD=9999,NOTRJ=3,NJCB=3,USER='DC RILEY',RLINCH=11.82,FHIGH=63.,
 NZC=0,FHIGH2=70.,FLOW2=16.,LENBP=11,IFLGAP=2,LCN=10,ZTAU=.20,XTAC=5.0,
 NSTR=2,NPTS=7,PLOC=0.,.2,.3,.5,1.,2.,2.5,BSIG=.8,.9,.999,.999,.999,.999,.999
 BRHO=.05,.1,.16,.18,.26,.36,.4,WSTRT=0.3,NOTR=1925,&END
/*
```

Processing Tape No. U1577- assuming package CF is on SYS07 then we're processing a 1971 field tape produced on the Teledyne 27110 digital recorder.

Processing tape no. U1823 - assuming package CN is on SYS07 then we're processing a 1972 analog/digital tape produced on the N.C.E.R. machine.

```
     JOB CARD GOES HERE
//     KEY CARD
/* SERVICE CLASS=Q
/* SETUP T,'PLS MOUNT TAPE U1823 (READ) ON UC1,RILEY,C740,USGSR044,TNX'
//JOBLIB DD DSN=C740.USGS,VOL=SER=SYS07,UNIT=2314,DISP=(OLD,PASS)
//SEISGO EXEC PGM=USGSC
//FT01F001 DD UNIT=2314,VOL=SER=SYS03,SPACE=(CYL,(3,2)),
//   DCB=(RECFM=VSB,BLKSIZE=7204,LRECL=144)
//FTC5F001 DD DDNAME=INPUT
//FT06F001 DD SYSOUT=A
//FT07F001 DD SYSOUT=B
//DISPLAY DD UNIT=004
//FT20F001 DD DSN=SEIS,VOL=SER=01823,DISP=(OLD,KEEP),UNIT=UC1,
//   LABEL=(1,BLP,,IN),DCB=(RECFM=U,DEN=1,TRTCH=C,BLKSIZE=16000)
//INPUT DD *
 &JOBIN NREEL=044,LINEID=' 223-224',RECEND=2.5,ITIME=600,RHO=.7,
 NBITS=3,MODADD=9999,MQTRJ=2,NJOB=3,USER='DC RILEY',RLINCH=11.82,FHIGH=63.,
 NZC=0,FHIGH2=70.,FLOW2=16.,LENBP=11,IFLGAP=2,LCN=10,ZTAU=.20,XTAU=5.0,
 NSTR=2,NPTS=7,PLOC=0.,.2,.3,.5,1.,2.,2.5,BSIG=.8,.9,.999,.999,.999,.999,.999
 BRHO=.05,.1,.16,.18,.26,.36,.4,WSTRT=0.3,NOTR=0300,&END
/*
```

SUGGESTED PROCEDURES

1. Load the appropriate package from the library tape onto a disk pack as per sec. 1X-C. Use the Futility partition (CLASS=F) if possible, this saves $ for 1/0 bound jobs like this.

2. Determine the depth in time on the section to which processing will be useful (RECEND). For conformance with the Raytheon fax recorder set NDØTS=3, MØDADD=9999, and RLINCH=4.72XRECEND.

3. Determine the number of traces to process.; i.e. set NØTR=no. tr. to process - NSTR, where NSTR is the starting trace no. Set number of traces to jump, i.e. NØTRJ=1 does them all, NØTRJ=2 skips every other trace. NØTRJ=2 is recommended.

4. Set the reel no. (NREEL), line identification (LINEID), and your name in 1-8 characters (USER).

5. Estimate the time of the job in minutes and code in the job card. Set 75% of this estimate in seconds in ITIME.

6. Set the NJØB parameter to the desired processing sequence.

7. Reset any other parameters to the desired values or use the default values.

8. Consider testing your selected parameters on a few traces (about NOTR=60 or so). This is most economically done with a job time limit of 1 min. in the IDLE priority of BATCH. Make any necessary alterations based on the results and test again. Then submit for complete processing overnite.*

9. When finished for a period with the program on disk it may be scratched from the lobby terminal via the command:

```
SCR &C740.USGS ØN SYS07 ACC=C740
```

*sec. IX-C illustrates the usual job stream

```
//LISTIT JOB 'C740,314','LIST PACKAGE C '                                JO
/* SERVICE LIST
C  U S G S  S E I S M I C  P R O C E S S I N G  P A C K A G E  C         US
C                                                                       US
      INTEGER TRACE*2(3950),BUF,LVERS(18)                               US
      REAL*4 CX(20,1000),BRHO(10),BSIG(10),PLOC(10)                     US
      REAL*8 DATE,TIME,LINEID,TAPEDD,USER                               US
      COMMON /BLK1/BUF(2000)                                            US
      COMMON /BLK2/Y(1000),LX,NJUMP,NLEAK,RHO,CRHO,NDOTS                US
      COMMON /BLK3/X(1000)                                              US
      EQUIVALENCE (TRACE(1),BUF(26))                                    US
      NAMELIST /JOB/NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,RHO,
     1NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,NOTRJ,NJOB,TAPEDD,USER,NPAGES
     2,RLINCH/ADAPT/FHIGH,NZC ,FHIGH2,FLOW2,LENBP,IFLGAP,LCN,DWARM,WSTRT
     3,ZTAU,XTAU/BIFLT/NPTS,PLOC,BSIG,BRHO/JOBIN/NREEL,LINEID,RANGE,WVEL
     4,SRATE,RECEND,NDEC,ITIME,RHO,NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,
     5NOTRJ,NJOB,TAPEDD,USER,NPAGES,RLINCH,FHIGH,NSTR,LENBP,IFLGAP,LCN,
     6DWARM,WSTRT,ZTAU,XTAU,FHIGH2,FLOW2,NPTS,PLOC,BSIG,BRHO,NZC
      DATA NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,NBITS,NBITS2,
     1NOTR,NOTRJ,NJOB,TAPEDD,FHIGH,NSTR,FLOW2,FHIGH2,LENBP,IFLGAP,LCN,
     2DWARM,WSTRT,ZTAU,XTAU,NPTS,NPAGES,RLINCH,PLOC,BRHO,BSIG,MODADD,
     3USER,NZC/999,'DEFERRED',400.,4875.,.002,1.53,2,540,5,4,2000,3,3,
     4'FT20F001',63.,1,  15.,63.,9,2,9,.2,.2,.12,10.,2,1,7.63,0.,1.536,
     58*0.,.5,.5,8*0.,.99,.99,8*0.0,9,'U.S.G.S.',1/
C                                                                       US
C_____________LIST__OF__PRINCIPAL__VARIABLES_____________________________
C
C  NAME (DEFAULT VALUE)          DEFINITION
C
C       <PARAMETERS USED IN ALL JOBS>
C  USER    (U.S.G.S.)      USER'S NAME (8 ALPHANUMERIC)
C  NREEL   (9999)          USGS REEL NUMBER (INTEGER)
C  LINEID  (DEFERRED)      USGS LINE NUMBER (8 ALPAHNUMERIC)
C  RANGE   (400.)          SHOT-RECEIVER OFFSET (FT.)
C  WVEL    (4875.)         WATER VELOCITY (FT/SEC)
C  SRATE   (.002)          INPUT SAMPLE RATE (SEC)
C  RECEND  (1.53)          RECORD LENGTH TO PROCESS (SEC)
C  NDEC    (2)             DEGREE OF DECIMATION (INTEGER)
C  ITIME   (540)           INTERNAL TIME ESTIMATE (SEC.,INTEGER)
C  RHO     (0.74)          DC REJECT FILTER COEFFICIENT
C  NBITS   (5)             NUMBER OF DOTS PER TRACE (INTEGER)
C  NBITS2  (4)             SECONDARY TRACEWIDTH EVERY 'MODADD' TRACES (INT)
C  MODADD  (9)             MODULO TO ADD SECONDARY BITS (INTEGER)
C  NOTR    (2000)          NUMBER OF TRACES TO BE PROCESSED (INTEGER)
C  NOTRJ   (3)             PROCESSING FREQUENCY (1,2,OR 3)
C  NSTR    (1)             NUMBER OF STARTING TRACE ON TAPE (INTEGER)
C  NJOB    (3)             JOB SELECTION SWITCH (1,2,3,OR 4)
C  TAPEDD  (FT20F001)      TAPE DATA DEFINITION TAG (8 ALPHANUMERIC)
C  RLINCH  (7.63)          RECORD SECTION LENGTH IN INCHES (MAX.=15.25)
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,2,3>
C  NLEAK   (10)            INTEGRATION PARAMETER FOR AGC (INTEGER)
C  NJUMP   (24)            UPDATE FREQUENCY OF AGC GAIN TRACE (INTEGER)
C  LENBP   (9)             NUMBER OF COEFFICIENTS IN BANDPASS (INTEGER,ODD)
C  FHIGH   (63.)           PRIOR HIGH CUTOFF (CYCLES/SEC)
C  FLOW2   (15.)           POST-PROC. LOW CUTOFF (CYCLES/SEC)
C  FHIGH2  (63.)           POST-PROC. HIGH CUTOFF (CYCLES/SEC)
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,3>
C  NPTS    (2)             NO. POINTS ON BIFILT PROFILE (INTEGER)
C  PLOC    (0.,1.53)       LOCATION OF PROFILE TIES (SEC)
C  BRHO    (.5,.5)         RHO PROFILE COEFFICIENTS
```

```
C  BSIG    (.99,.99)      SIGMA PROFILE COEFFICIENTS
C       <ADDITIONAL PARAMETERS USED IN JOBS 2,3>
C  NZC     (1)            NO. OF LEADING ZERO REFLECTION COEFF.(INT)
C  IFLGAP  (2)            GAP BETWEEN ADAPTIVE FILTER COEFFICIENTS (INT)
C  LCN     (9)            NO. OF REFLECTION COEFF. IN LADDER (INTEGER)
C  DWARM   (.2)           DURATION OF STATIONARY WARM-UP CYCLE (SEC)
C  WSTRT   (.2)           STARTING POSITION OF WARM-UP (SEC)
C  ZTAU    (.12)          RELAXATION TIME TO 1/E (SEC)
C  XTAU    (10.)          RELAXATION DISTANCE TO 1/E (TRACES)
C
C              (ALL VARIABLES ARE REAL EXCEPT AS INDICATED)
C                                                                      US
C______________________________________________________________________UC
C                                                                      US
C            PROCESSING SEQUENCES:                                     US
C                                                                      US
C       NJOB=1    /FIELD/NMO/DCDEC/AGC/BPASS/BIFILT/BPASS2/SEISGM/     US
C       NJOB=2    /FIELD/NMO/DCDEC/AGC/BPASS/BAFL/BPASS2/SEISGM/       US
C       NJOB=3    /FIELD/NMO/DCDEC/AGC/BPASS/BAFL/BIFILT/BPASS2/SEISGM/ US
C       NJOB=4    /FIELD/NMO/DCDEC/SEISGM/                             US
C______________________________________________________________________US
C            VERSION Q                          DON C. RILEY AUGUST 1972
C                                                                      US
      CALL MCLOCK(DATE,TIME,WEEKDA)                                    US
      CALL PCLOCK(IJS)                                                 US
      RHO=0.74
      NLEAK=10
      NJUMP=24
      WRITE(6,876)                                                     US
      READ(5,JOBIN,ERR=977)
    2 LENGTH=RECEND/SRATE                                              US
      IF(NSTR.LE.1) GO TO 4
      M=NSTR-1
      DO 3 I=1,M
    3 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
    4 LX=LENGTH/NDEC                                                   US
      RLINCH=AMIN1(RLINCH,15.3)                                        C
      IF(RLINCH.GT.7.65) NPAGES=2                                      C
      SRATE2=SRATE*NDEC                                                US
      CRHO=(1.+RHO)/2.                                                 US
      KTEST=24*NOTRJ                                                   US
      WRITE(6,JOB)
      IF(NJOB.EQ.4)  GO TO 40                                          US
      FLOW=-FHIGH
      IF(MOD(LENBP,2).EQ.0)  LENBP=LENBP+1                             US
      CALL GFILT(FLOW,FHIGH,SRATE2,LENBP,FLOW2,FHIGH2,LX)              US
      IF(NJOB.EQ.1) GO TO 30                                           US
      WRITE(6,ADAPT)                                                   US
      NSTRT=WSTRT/SRATE2
      NWARM=DWARM/SRATE2
      ZTAT=ZTAU/SRATE2
      IF(NJOB.EQ.2) GO TO 40                                           US
   30 WRITE(6,BIFLT)
      CALL BIFILT(LX,BSIG,BRHO,PLOC,NPTS,SRATE2)
   40 CALL TAXIS(LX,SRATE2,NPAGES,RLINCH)                              US
      CALL NMOSEI(RANGE,WVEL,SRATE,LENGTH)                             US
      ITIME=ITIME*100
      CALL FI0999(44,LVERS(1),18,13)                                   US
C                                                                      US
      GO TO (1727,1707,1747,1737),NJOB                                 US
```

```
C                                                                       USC
C                   NJOB=2                                              USC
C                                                                       USC
 1707 CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  USC
      CALL NMO                                                          USC
      CALL DCDEC(TRACE,NDEC)                                            USC
      CALL AGC                                                          USC
      CALL BPASS                                                        USC
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)             USC
      CALL BPASS2                                                       USC
      NDOTS=NBITS                                                       USC
      CALL SEISGM                                                       USC
      DO 707 I=NOTRJ,NOTR,NOTRJ                                         USC
      NDOTS=NBITS                                                       USC
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              USC
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  USC
      IF(IERR.EQ.1) GO TO 988                                           USC
      CALL NMO                                                          USC
      CALL DCDEC(TRACE,NDEC)                                            USC
      CALL AGC                                                          USC
      CALL BPASS                                                        USC
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 USC
      IF(IERR.EQ.1) GO TO 988                                           USC
      CALL BAFLGO                                                       USC
      IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 USC
      CALL BPASS2                                                       USC
      CALL SEISGM                                                       USC
      IF(MOD(I,KTEST).NE.0) GO TO 707                                   USC
      CALL PCLOCK(IJE,IJS)                                              USC
      IF(IJE.GT.ITIME) GO TO 909                                        USC
  707 CONTINUE                                                          USC
      GO TO 999                                                         USC
C                                                                       USC
C                                                                       USC
C                   NJOB=1                                              USC
C                                                                       USC
 1727 DO 727 I=NOTRJ,NOTR,NOTRJ                                         USC
      NDOTS=NBITS                                                       USC
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              USC
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  USC
      IF(IERR.EQ.1) GO TO 988                                           USC
      CALL NMO                                                          USC
      CALL DCDEC(TRACE,NDEC)                                            USC
      CALL AGC                                                          USC
      CALL BPASS
      CALL BIGOT                                                        USC
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 USC
      IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 USC
      IF(IERR.EQ.1) GO TO 988                                           USC
      CALL BPASS2
      CALL SEISGM                                                       USC
      IF(MOD(I,KTEST).NE.0) GO TO 727                                   USC
      CALL PCLOCK(IJE,IJS)                                              USC
      IF(IJE.GT.ITIME) GO TO 909                                        USC
  727 CONTINUE                                                          USC
      GO TO 999
C                                                                       USC
C                   NJOB=4                                              USC
C                                                                       USC
 1737 DO 737 I=NOTRJ,NOTR,NOTRJ                                         USC
```

```
      NDOTS=NBITS                                                       USG
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              USG
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  USG
      IF(IERR.EQ.1) GO TO 988                                           USG
      CALL NMO                                                          USG
      CALL DCDEC(TRACE,NDEC)                                            USG
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 USG
      IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 USG
      IF(IERR.EQ.1) GO TO 988                                           USG
      CALL SEISGM                                                       USG
      IF(MOD(I,KTEST).NE.0) GO TO 737                                   USG
      CALL PCLOCK(IJE,IJS)                                              USG
      IF(IJE.GT.ITIME) GO TO 909                                        USG
  737 CONTINUE                                                          USG
      GO TO 999                                                         USG
C                                                                       USG
C                 NJOB=3                                                USG
C                                                                       USG
 1747 CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  USG
      CALL NMO                                                          USG
      CALL DCDEC(TRACE,NDEC)                                            USG
      CALL AGC                                                          USG
      CALL BPASS                                                        USG
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)             USG
      CALL BIGOT                                                        USG
      CALL BPASS2                                                       USG
      NDOTS=NBITS                                                       USG
      CALL SEISGM                                                       USG
      DO 747 I=NOTRJ,NOTR,NOTRJ                                         USG
      NDOTS=NBITS                                                       USG
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              USG
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  USG
      IF(IERR.EQ.1) GO TO 988                                           USG
      CALL NMO                                                          USG
      CALL DCDEC(TRACE,NDEC)                                            USG
      CALL AGC                                                          USG
      CALL BPASS                                                        USG
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 USG
      IF(IERR.EQ.1) GO TO 988                                           USG
      CALL BAFLGO                                                       USG
      CALL BIGOT                                                        USG
      IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                 USG
      CALL BPASS2                                                       USG
      CALL SEISGM                                                       USG
      IF(MOD(I,KTEST).NE.0) GO TO 747                                   USG
      CALL PCLOCK(IJE,IJS)                                              USG
      IF(IJE.GT.ITIME) GO TO 909                                        USG
  747 CONTINUE                                                          USG
C                                                                       USG
C                                                                       USG
C                                                                       USG
  999 CALL PCLOCK(IJE,IJS)                                              USG
      WRITE(6,900) IJE                                                  USG
      CALL XCLOCK(IJS)                                                  USG
      WRITE(44,622) USER
      DO 5 K=1,19
    5 CALL VLINE(LVERS,70)
      CALL VLINE(' ',1)                                                 USG
      CALL VLINE(' ',1)                                                 USG
C
```

```
C     WRITE PARM.JOB TO VERSATEC.....                                   USG
      WRITE(44,600)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,601) NREEL                                               USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,602) LINEID                                              USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,603)                                                     USG
      GO TO (50,60,70,45),NJOB                                          USG
   45 WRITE(44,621)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      GO TO 80                                                          USG
   50 WRITE(44,614)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,615)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      GO TO 80                                                          USG
   60 WRITE(44,604)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,605)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      GO TO 80                                                          USG
   70 WRITE(44,616)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,617)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
   80 WRITE(44,603)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,606) RANGE,WVEL,SRATE,NSTR                               USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,609) RLINCH,NLEAK,NJUMP,NBITS,MODADD                     USG
      CALL VLINE(LVERS,70)                                              USG
      IF(NJOB.EQ.4)  GO TO 120                                          USG
      IF(NJOB.EQ.1) GO TO 110                                           USG
      WRITE(44,607) FLOW2,FHIGH2,LENBP,IFLGAP,LCN,NZC                   USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,608) DWARM,WSTRT,ZTAU,XTAU,RHO                           USG
      CALL VLINE(LVERS,70)                                              USG
      IF(NJOB.EQ.2) GO TO 120                                           USG
  110 WRITE(44,618) NPTS,PLOC                                           USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,619) BSIG                                                USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,620) BRHO                                                USG
      CALL VLINE(LVERS,70)                                              USG
  120 WRITE(44,610) NOTR,NOTRJ,RECEND,NDEC,NPAGES                       USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,603)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,612)  IJE,DATE,TIME                                      USG
      CALL VLINE(LVERS,70)                                              USG
      WRITE(44,600)                                                     USG
      CALL VLINE(LVERS,70)                                              USG
C                                                                       USG
C                                                                       USG
      CALL VLINE(' ',1)                                                 USG
      CALL VLINE(' ',1)                                                 USG
      CALL VLINE('                T W O - W A Y  T R A V E L  T I M E',USG
     152)                                                               USG
      CALL VLINE(' ',1)                                                 USG
```

```
      CALL VLINE(' ',1)                                                 USG
      CALL XCLOCK(IJE,IJS)                                              USG
      WRITE(6,879) IJE                                                  USG
      CALL XCLOCK(IJS)                                                  USG
      CALL SDUMP                                                        USG
      CALL XCLOCK(IJE,IJS)                                              USG
      WRITE(44,613) IJE                                                 USG
      WRITE(6,613)  IJE                                                 USG
      CALL VLINE(LVERS,70)                                              USG
      CALL VLINE('                          END OF PLOT        BIN 314',49)USG
      READ(5,JOBIN,ERR=977,END=9999)                                    USG
      WRITE(6,877)                                                      USG
      CALL MCLOCK(DATE,TIME,WEEKDA)                                     USG
      CALL PCLOCK(IJS)                                                  USG
      CALL ENDTAP                                                       USG
      REWIND 1                                                          USG
      GO TO 2                                                           USG
 9999 STOP                                                              USG
  988 WRITE(6,880) I                                                    USG
      GO TO 999                                                         USG
  989 WRITE(6,878) I                                                    USG
      WRITE(44,878) I                                                   USG
      CALL VLINE(LVERS,70)                                              USG
      GO TO 999                                                         USG
  977 WRITE(6,881)
      GO TO 9999
C                                                                       USG
C        F O R M A T S                                                  USG
  600 FORMAT(70('*'))                                                   USG
  601 FORMAT('*',8X,'CHUKCHI SEA 1971 SEISMIC DATA  U.S.G.S. REEL NO.', USG
     1 I4,8X,'*')                                                       USG
  602 FORMAT('*',10X,'SINGLE-FOLD MARINE REFLECTION PROFILE NO.',A8,9X, USG
     1 '*')                                                             USG
  603 FORMAT('*',30X,'<<<<>>>>',30X,'*')                                USG
  604 FORMAT('*',15X,'-TIME & SPACE ADAPTIVE DECONVOLUTION-',16X,'*')   USG
  605 FORMAT('*',6X,'PROCESSING SEQUENCE: /FIELD/NMO/DCDEC/AGC/BAFLGO/SEJSG
     1ISGM/',6X,'*')                                                    USG
  606 FORMAT('*   PARM.JOB=',1H','RANGE=',F5.0,',WVEL=',F6.0,',SRATE=', JSG
     1 F6.3,',NSTR=',I4,',          *')                                USG
  607 FORMAT('*',13X,'FLOW2=',F3.0,',FHIGH2=',F3.0,',LENBP=',I2,',IFLGAPUSG
     1=',I1,',LCN=',I2,',NZC=',I1,',    *')                           USG
  608 FORMAT('*',13X,'DWARM=',F3.2,',WSTRT=',F3.2,',ZTAU=',F5.3,',XTAU='
     1,F5.1,',RHO=',F3.2,',',5X,'*')
  609 FORMAT('*',13X,'RLINCH=',F4.1,',NLEAK=',I3,',NJUMP=',I4,',NBITS=',JSG
     1 I2,',MODADD=',I2,',    *')                                      USG
  610 FORMAT('*',13X,'NOTR=',I4,',NOTRJ=',I1,',RECEND=',F4.2,',NDEC=',  USG
     1I1,',NPAGES=',I1,1H',9X,'*')                                      USG
  612 FORMAT ('*  360/67 CPU TIME= ',I5,' CSEC.  DATE: ',A8,2X,          USG
     1 A8,'  STANFORD  *')                                             USG
  613 FORMAT('   DRUM.DUMP.PLAYBACK.TIME = ',I7,' CSEC.')              USG
  614 FORMAT('*',23X,'-HORIZONTAL BIFILTER-',24X,'*')                   USG
  615 FORMAT ('*',6X,'PROCESSING SEQUENCE: /FIELD/NMO/DCDEC/AGC/BIFILT/SUSG
     1EISGM/',6X,'*')                                                   USG
  616 FORMAT('*',10X,'-TIME & SPACE ADAPTIVE DECONVOLUTION + BIFILTER-',USG
     110X,'*')                                                          USG
  617 FORMAT('*',5X,'PROCESSING SEQ: /FIELD/NMO/DCDEC/AGC/BAFLGO/BIFILT/ USG
     1SEISGM/',5X,'*')                                                  USG
  618 FORMAT('*',13X,'NPTS=',I2,',LCC=',10(F3.1,','),T70,'*')
  619 FORMAT('*',20X,'BSIC=',10(F3.2,','),T70,'*')                      USG
  620 FORMAT('*',20X,'BRHO=',10(F3.2,','),T70,'*')                      USG
```

```
  621 FORMAT('*',9X,'SEISMIC PLAYBACK ONLY SEQ: /FIELD/NMO/DCDEC/SEISGN/USG
     1',8X,'*')                                                          USG
  622 FORMAT('START OF PLOTJOB ',A8,', BIN 314')
  876 FORMAT(1H0,30X,10('*'),' BEGIN CHUKCHI JOB A ',10('*'))            USG
  877 FORMAT(1H1,30X,10('*'),' BEGIN CHUKCHI JOB B ',10('*'))            USG
  878 FORMAT(1H0,'EST. TIMER SELF-EXIT AT TR. ',I8)                      USG
  879 FORMAT(1H ,'W-BLOCK TIMER:',I8)                                    USG
  880 FORMAT(1H ,'***END OF FILE ENCOUNTERED AT TR. NO.',I6)             USG
  881 FORMAT(1H0,'***ERROR IN INPUT CARDS *** ABNEOJ ***')
  900 FORMAT(1H0,7X,'PARTITION TIMER IN CSEC. ',I8)                      USG
C                                                                        US2
      END                                                                USG
      SUBROUTINE BAFL(LOUT,CX,IFLGAP,LCN,NWARM,ISTRT,ZTAU,XTAU,NZERO)    USG
C                                                                        USG
C                    ----------------------------                        USG
C-------------------S U B R O U T I N E   B A F L-----------------       USG
C                    ----------------------------                        USG
C                                                                        USG
C        THE BURG ADAPTIVE FILTER LADDER: AN ADAPTIVE OR TIME-VARYING    USG
C        FIXED-LEAD PREDICTION ERROR PROCESSOR.  ADJUSTMENT OF EACH      USG
C        REFLECTION COEFFICIENT IS MADE EVERY JUMP STATE ATTEMPTING      USG
C        TO MINIMIZE THE STAGE OUTPUT POWER.                             USG
C        INPUTS:                                                         USG
C          X(1)...X(LX)=INITIAL DATA                                     USG
C          LCN= LAST NON-ZERO REFLECTION COEFFICIENT                     USG
C          IFLGAP=  NUMBER OF GAPS BETWEEN FILTER COEFFICIENTS           USG
C              SETTING IFLGAP=0 DOES NOT GAP THE F.C. AND                USG
C              TRIES TO OPERATE ON THE ENTIRE SPECTRUM FROM              USG
C              0 TO W WHERE W IS THE FOLDING FREQUENCY.                  USG
C              SETTING IFLGAP=1 OPERATES ON THE PORTION OF               USG
C              THE SPECTRUM FROM 0 TO W/2 , IFLGAP=2 FROM                USG
C              0 TO W/3 ETC. THUS ALLOWING SPECIFICATION OF              USG
C              WHAT PART OF THE SPECTRUM TO DECONVOLVE.                  USG
C                                                                        USG
C          NWARM=DURATION OF STATIONARY C ESTIMATION CYCLE               USG
C          ISTRT=START OF STATIONARY GATE                                USG
C          ZTAU=TEMPORAL RELAXATION TIME TO 1/E                          USG
C          XTAU=SPATIAL RELAXATION DISTANCE TO 1/E                       USG
C        OUTPUTS:                                                        USG
C          X(1)...X(LOUT)=FORWARD ERROR PREDICTION TRACE                 USG
C        OTHER VARIABLES:                                                USG
C          F(1)...F(LCN)=FORWARD STATE VECTOR                            USG
C          B(1)...B(LCN)=BACKWARD STATE VECTOR                           USG
C          C(1)...C(LCN)=REFLECTION COEFFICIENTS AT EACH STATE           USG
C          CX=REFLECTION COEFF. INTEGRATED IN SPACE & TIME               USG
C          DEN(1)...DEN(LCN)=  STAGE AUTOPOWER                           USG
C          NUM(1)...NUM(LCN)= STAGE CROSSPOWER                           USG
C                                                                        USG
C          CALLING 'BAFL' FIRST SETS UP THE LOOPING AND PASSING          USG
C          ARRAYS FOR THE PARTICULAR PROBLEM AS SPECIFIED BY LCN &       USG
C          IFLGAP.  THEN IT COMPUTES A SHORT (LENGTH=NWARM) ESTIMATE     USG
C          OF THE REFLECTION COEFFICIENT SERIES IN ORDER TO START THE     USG
C          ADAPTION OUT WITH SOME REASONABLE NUMBERS. THEN IT LOADS UP    USG
C          THE CX ARRAY WITH THE INITIAL VALUES AND PASSED INTO ENTRY     USG
C          'BAFLGO'.                                                     USG
C                                                                        USG
C          THE USUAL ENTRY IS 'BAFLGO' WHICH FIRST INITIALIZES THE       USG
C          BACKWARD ARRAY THEN PASSES TO THE MAIN ALGORITHM.             USG
C          THE CX SERIES IS UPDATED EVERY IFLGAP DATA POINTS AND IN      USG
C          THE INTERMEDIATE STEPS THE OUTPUT ARE INTERPOLATED OR         USG
```

```
C          PROCESSED AS THOUGH THE R.C. WERE STATIONARY.                    US
C                                                                           US
C                                                                           US
C                                                                           US
C                                                                           US
      REAL NUM(50),DEN(50),B(50),F(50),O(50),EM(1000),EP(1000),             US
     1 CX(LCN,LOUT),A(50)                                                   US
      COMMON /BLK3/X(1000)                                                  US
      DATA A,B,C/150*0.0/                                                   US
      FTEST=1.00                                                            US
      LCNP1=LCN+1                                                           US
      LCNP2=LCN+2                                                           US
      IFGM1=IFLGAP-1                                                        US
      LBSP=LOUT-IFLGAP                                                      US
      NZERP1=NZERO+1                                                        US
      NEND=LCN-NZERP1                                                       US
      DO 80 K=LBSP,LOUT                                                     US
   80 EP(K)=0.                                                              US
C BEGIN STATIONARY WARM-UP                                                  US
      DO 10 I=1,NWARM                                                       US
      EM(I)=X(I*IFLGAP+ISTRT)                                               US
   10 EP(I)=X(I*IFLGAP+ISTRT)                                               US
C                                                                           US
      DO 11 J=2,LCNP1                                                       US
      DEN(J)=0.                                                             US
      NUM(J)=0.                                                             US
      DO 12 I=J,NWARM                                                       US
      DEN(J)=DEN(J)+EP(I)*EP(I)+EM(I-J+1)*EM(I-J+1)                         US
   12 NUM(J)=NUM(J)+EP(I)*EM(I-J+1)                                         US
      DIV=NWARM-J+1                                                         US
      NUM(J)=NUM(J)/DIV                                                     US
      DEN(J)=DEN(J)/DIV                                                     US
      C(J)=-2.*NUM(J)/DEN(J)                                                US
      DO 11 I=J,NWARM                                                       US
      EPI=EP(I)                                                             US
      EP(I)=EPI+C(J)*EM(I-J+1)                                              US
      KS=IFLGAP*LCN+1                                                       US
   11 EM(I-J+1)=EM(I-J+1)+C(J)*EPI                                          US
      DO 8 J=  1,LCN                                                        US
      DO 8 K=1,KS                                                           US
    8 CX(J,K)=C(J+1)                                                        US
      DO 33 J=1,LCN                                                         US
      DEN(J)=DEN(J+1)                                                       US
   33 NUM(J)=NUM(J+1)                                                       US
C END WARM-UP CYCLE                                                         US
C SET RELAXATION TIMES                                                      US
      DLX=EXP(-1./XTAU)                                                     US
      DL=EXP(-1./ZTAU)                                                      US
      DRX=1.-DLX                                                            US
      DR=1.-DL                                                              US
C                                                                           US
C----------------USUAL--ENTRY-------------------------------------          US
      ENTRY BAFLGO                                                          US
C                                                                           US
C          INITIALIZE BACKWARD VECTOR                                       US
      B(1)=X(KS-IFLGAP)                                                     US
      A(1)=1.0                                                              US
      DO 1000 J=2,LCN                                                       US
      B(J)=X(KS-J*IFLGAP)                                                   US
      A(J)=CX(J,KS)                                                         US
```

```
      DO 1000 I=2,J                                                     USG
      A(I)=A(I)+CX(J,KS)*A(J-I+1)                                       USG
 1000 B(J)=B(J)+A(I)*X(KS+(I-J-1)*IFLGAP)                               USG
C                                                                       USG
C--------BEGIN--MAIN--LOOP-------------------------------------------  USG
C                                                                       USG
      DO 5000 K=KS,LOUT,IFLGAP
      Z=X(K)
      DO 1010 J=1,NZERP1
 1010 F(J)=Z
      DO 2000 J=NZERP1,LCN
      DEN(J)=(F(J)**2+B(J)**2)*DR+DEN(J)*DL
      NUM(J)=F(J)*B(J)*DR+NUM(J)*DL
      IF(FTEST.LE.1.1) CX(J,K)=-2.*NUM(J)/DEN(J)
      CX(J,K)=-2.*DRX*NUM(J)/DEN(J)+CX(J,K)*DLX
 2000 F(J+1)=F(J)+CX(J,K)*B(J)
      X(K)=F(LCNP1)
      DO 3000 JR=1,NEND
      J=LCN-JR
 3000 B(J+1)=B(J)+CX(J,K)*F(J)
      IF(NZERO.EQ.0) GO TO 5000
      DO 4000 JR=1,NZERO
      J=NZERP1-JR
 4000 B(J+1)=B(J)
 5000 B(1)=Z
C                                                                       USG
C--------END--OF--MAIN--LOOP-------------------------------------------  USG
C                                                                       USG
C       NOW GO BACK AND FILL IN THE GAPS....                            USG
      IF(IFLGAP.EQ.1) GO TO 9050
      DO 9000 L=1,IFGM1
      KSTART=KS+L
      DO 9000 K=KSTART,LOUT,IFLGAP
      KC=K-L
      Z=X(K)
      DO 6000 J=1,NZERP1
 6000 F(J)=Z
      DO 7000 J=NZERP1,LCN
 7000 F(J+1)=F(J)+CX(J,KC)*B(J)
      X(K)=F(LCNP1)
      DO 8000 JR=1,NEND
      J=LCN-JR
 8000 B(J+1)=B(J)+CX(J,KC)*F(J)
      IF(NZERO.EQ.0) GO TO 9000
      DO 8050 JR=1,NZERO
      J=NZERP1-JR
 8050 B(J+1)=B(J)
 9000 B(1)=Z
C                                                                       USG
 9050 FTEST=FTEST+1.0
      RETURN                                                            USG
      END                                                               USG
      SUBROUTINE SEISGM                                                 USG
C                                                                       USG
C        SUBROUTINE 'SEISGM'   VARIABLE AREA SEISMIC SECTION PLOTTER.   USG
C        THROUGH MULTIPLE CALLS TO SEISGM ADJACENT TRACES ARE WRITTEN   USG
C        TO THE VERSATEC FILLING THE FULL-PAGE WIDTH (560 BITS).        USG
C        DATA: INPUT DATA SERIES                                        USG
C        NT = LENGTH OF INPUT DATA                                      USG
C        NBITS= WIDTH OF VARIABLE AREA TRACE IN VERSATEC BITS           USG
```

```
C         NPAGES: WHEN NPAGES=1 THE RECORD SECTION FILLS ONE PAGE          USC
C                 OF THE VERSATEC (7.536 IN.) FOR ANY LENGTH RECORDS.      USC
C                 WHEN NPAGES=2 A SPECIAL OPTION IS INVOKED ALLOWING       USC
C                 SOMEWHAT MORE GENERAL DISPLAY AS FOLLOWS.....            USC
C                 'PLINCH' INDICATES THE DESIRED RECORD SECTION LENGTH     USC
C                 IN INCHES. THUS THE RECORDS OF LENGTH NT ARE SCALED      USC
C                 INTO A VARIABLE OBJECT SPACE THAT MAY TAKE UP MORE       USC
C                 THAN ONE PAGE-WIDTH.  THIS IS ACCOMPLISHED BY COMPOSINGUSC
C                 THE PLOT INTO A LONGER BUFFER THAN THE VERSATEC CAN      USC
C                 HANDLE(WHICH IS 70 BYTES) AND DUMPING THIS PLOTTED       USC
C                 DATA ONTO A 2301 DRUM (FT UNIT 1) .  THEN WE REWIND ANDUSC
C                 FETCH AND WRITE THE FIRST 70 BYTES ON ONE PAGE; THEN     USC
C                 GO BACK AND FETCH AND WRITE THE NEXT 70 BYTES OR         USC
C                 LESS.                                                    USC
C                                                                          USC
C     'TAXIS' MUST BE CALLED BEFORE 1ST 'SEISGM' CALL SINCE                USC
C     A REFERENCE TABLE MUST BE GENERATED.                                 USC
C                                                                          USC
      INTEGER*4 IREF(1120)                                                 USC
      LOGICAL*4 LMZ,LZ,LASK,LINE(35,10),LOAD(18),LOAD2(35),LBOT*1(72)      USC
      LOGICAL*4 LTAX(35),LIGHT(35),DARK(35),LMASK(32)                      USC
      COMMON /BLK2/DATA(1000),NT,N5,N6,R5,R6,NBITS                         USC
      EQUIVALENCE (LBOT(1),LOAD2(18))                                      USC
      DATA LMZ,LZ,LIGHT,DARK/Z80000000,36*Z00000000,35*ZFFFFFFFF/          USC
      DATA LMASK/Z80000000,Z40000000,Z20000000,Z10000000,Z08000000,        USC
     1Z04000000,Z02000000,Z01000000,Z00800000,Z00400000,Z00200000,         USC
     2Z00100000,Z00080000,Z00040000,Z00020000,Z00010000,Z00008000,         USC
     3Z00004000,Z00002000,Z00001000,Z00000800,Z00000400,Z00000200,         USC
     4Z00000100,Z00000080,Z00000040,Z00000020,Z00000010,Z00000008,         USC
     5Z00000004,Z00000002,Z00000001/                                       USC
      NMAG=MIN0(10,NBITS)                                                  USC
      SUM=0.                                                               USC
      DO 121 K=1,NT,4                                                      USC
      T=DATA(K)                                                            USC
  121 SUM=SUM+ABS(T)                                                       USC
      ERMS=4.*SUM/NT                                                       USC
C     WRITE(6,900) ERMS                                                    USC
C 900 FORMAT(1H+,100X,F11.1)                                               USC
      SCALE=0.5*NMAG/ERMS                                                  USC
      IBIAS=(NMAG+1)/2                                                     USC
      DO 20 I=1,JBYTES                                                     USC
      DO 20 J=1,NMAG                                                       USC
C   SETTING = LTAX WRITES TIME MARKS ACROSS ENTIRE SECTION,LZ DOESN'T      USC
C  20 LINE(I,J) = LZ                                                       USC
   20 LINE(I,J)=LTAX(I)                                                    USC
      DO 40 IBIT=1,32                                                      USC
      LASK=LMASK(IBIT)                                                     USC
      IBYTE = 1                                                            USC
      DO 40 IL=IBIT,JBITS,32                                               USC
      IMAG=IBIAS+DATA(IREF(IL))*SCALE                                      USC
      IF(IMAG.LE.0) GO TO 40                                               USC
      IMAG=MIN0(NMAG,IMAG)                                                 USC
      DO 30 I=1,IMAG                                                       USC
   30 LINE(IBYTE,I) = LINE(IBYTE,I).OR.LASK                                US
   40 IBYTE = IBYTE + 1                                                    US
      DO 50 I=1,NMAG                                                       US
   50 WRITE(1)  (LINE(K,I),K=1,JBYTES)                                     US
      LKNT = LKNT + NMAG                                                   US
      RETURN                                                               US
C                                                                          US
```

```
C         ENTRY 'TAXIS' WRITES A TIME SCALE TO THE VERSATEC.
C         SRATE IS THE SAMPLING RATE.  TICK MARKS ARE AT 100 MILS.
C
      ENTRY TAXIS(NT,SRATE,NPAGES,RLINCH)
      JBYTES=18
      JBITS=560
      IF(NPAGES.LE.1) GO TO 55
      JBYTES=35
      JBITS=MIN1(1120.,RLINCH*78.25)
   55 SCALE=(JBITS-1)/(NT*SRATE)
      ITMAX=SRATE*NT*10. + 1.
      TYME=0.
      DO 60 I=1,JBYTES
   60 LTAX(I)=LZ
      DO 70 I=1,ITMAX
      IR=SCALE*TYME + 1.0
      IBYTE=(IR-1)/32  + 1
      IBIT=IR - (IBYTE-1)*32
      LASK=LMASK(IBIT)
      LTAX(IBYTE)=LTAX(IBYTE).OR.LASK
   70 TYME=TYME + 0.1
      SCALE=(NT-1.)/(JBITS-1.)
      DO 90 IL=1,JBITS
   90 IREF(IL)=SCALE*(IL-1) + 1.49999
      DO 83 I=1,8
   83 WRITE(1) LIGHT
      WRITE(1) DARK
      DO 84 I=1,10
   84 WRITE(1) LTAX
      WRITE(1) DARK
      DO 81 I=1,10
   81 WRITE(1) LIGHT
      LKNT=30
C
      RETURN
C
C         ENTRY SDUMP DUMPS THE DRUM TO THE VERSATEC
C
      ENTRY SDUMP
      DO 85 I=1,380
   85 WRITE(1) LIGHT
      REWIND 1
      NL=LKNT + 378
      NFIN=NL
      IF(NPAGES.GT.1) NFIN=NL-340
      DO 82 I=1,NFIN
      READ(1) LOAD
   82 CALL WRITER(LOAD,70)
      IF(NPAGES.LE.1) GO TO 87
      REWIND 1
      DO 86 I=1,NL
      READ(1) LOAD2
   86 CALL WRITER(LBOT(3),70)
   87 CONTINUE
      RETURN
      END
      SUBROUTINE BIFILT(ND,SIG,RHO,PLOC,NPTS,SRATE2)
C                 BILINER KX FILTER
C                 USEFUL FOR EXTINGUISHING HORIZONTAL PRIMARIES
C                 AND MULTIPLES INTERFERING WITH DIPPING REFLECTORS
```

```
C                  AND/OR FOR STACKING IN  X  TO ENHANCE LATERAL          US
C                  CORRELATION IN THE PRESENCE OF ADDITIVE NOISE.         US
C                  ALGORITHM OPERATES IN TIME VARYING MODE BETWEEN        US
C                  PURE  KX  LOWCUT (RHO=0) AND PURE  KX  HIGHCUT         US
C                  (SIG=1) AS SET IN THE RHO/SIG PROFILES.                US
C                                                                         US
C             PARAMETER LIST:                                             US
C             IN      INPUT TRACE                                         US
C             OUT     OUTPUT TRACE                                        US
C             ND      LENGTH OF INPUT/OUTOUT TRACE                        US
C             SIG     SIGMA PARAMETER PROFILE                             US
C             RHO     RHO PARAMETER PROFILE                               US
C             LOC     LOCATION POINTS OF PARAMETER CHANGES                US
C             NPTS    NO. OF SIG/RHO/LOC POINTS                           US
C                                                                         US
C               RHO                    SIGMA                            US
C    LOC1----------X-------   --X--------------                           US
C         |          . RHO1       |   .                                   US
C         |        .              |  .                                    US
C    LOC2|       XRHO2            |    XSIG2                              US
C         |       .               |      .                                US
C    LOC3|       XRHO3            |       XSIG3                           US
C         |      .                |        .                              US
C         |     .                 |         .                             US
C    LOC4|    XRHO4              |          XSIG4                        US
C         |                       |                                       US
C                                                                         US
      REAL*4 SIG(1),RHO(1),PLOC(1),D(1000),E(1000),F(1000)                US
      REAL*4 IN,OUTH(1000),OUTHR(1000),INH(1000)                          US
      INTEGER*4 LOC(10)                                                   US
      COMMON /BLK3/IN(1000)                                               US
      LOC(1)=1
      DO 5 K=2,NPTS
      LOC(K)=PLOC(K)/SRATE2
      IF(LOC(K-1).GT.LOC(K)) GO TO 97
    5 CONTINUE
      LOC(NPTS)=ND                                                        US
      NPM1=NPTS-1                                                         US
      WRITE(6,800) RHO(1),SIG(1)                                          US
  800 FORMAT(1HO,6X,'PARAMETER PROFILE'//1H ,3X,'DEPTH',                  US
     1 4X,'RHO',5X,'SIGMA'//1H ,6X,'1',2(3X,F6.3))                        US
      DO 10 IW=1,NPM1                                                     US
      WRITE(6,805) LOC(IW+1),RHO(IW+1),SIG(IW+1)                          US
  805 FORMAT(1H ,3X,I4,2(3X,F6.3))                                        US
      RMR=(RHO(IW+1)-RHO(IW))/(LOC(IW+1)-LOC(IW))                         US
      RMS=(SIG(IW+1)-SIG(IW))/(LOC(IW+1)-LOC(IW))                         US
      NEND=LOC(IW+1) - 1                                                  US
      NSTRT=LOC(IW)                                                       US
      IF(NSTRT.GT.NEND) GO TO 97                                          US
      DO 10 K=NSTRT,NEND                                                  US
      R=RHO(IW) + RMR*(K-NSTRT)                                           US
      S=SIG(IW) + RMS*(K-NSTRT)                                           US
      D(K)=0.5*(1.+S)*(1.-R)                                              US
      E(K)=S+R                                                            US
   10 E(K)=S*R                                                            US
      E(ND)=SIG(NPTS)+RHO(NPTS)                                           US
      F(ND)=SIG(NPTS)*RHO(NPTS)                                           US
      D(ND)=0.5*(1.+SIG(NPTS))*(1.-RHO(NPTS))                             US
      DO 20 K=1,ND                                                        US
      OUTH(K)= 0.0                                                        US
```

```
      INH(K)= 0.0                                                       USC
   20 OUTHH(K)=0.0                                                      USC
      RETURN                                                            USC
C                                                                       USC
      ENTRY BIGOT                                                       USC
      DO 70 K=1,ND                                                      USC
      TEMP=D(K)*(IN(K)-INH(K))+E(K)*OUTH(K)-F(K)*OUTHH(K)               USC
      OUTHH(K)=OUTH(K)                                                  USC
      OUTH(K)=TEMP                                                      USC
      INH(K)=IN(K)                                                      USC
   70 IN(K)=TEMP                                                        USC
      RETURN                                                            USC
   97 WRITE(6,900)                                                      USC
  900 FORMAT(1HO,'***PARM.RHO/SIG ERROR***ABNEOJ***')                   USC
      STOP                                                              USC
      END                                                               USC
      SUBROUTINE AGC                                                    USC
      DIMENSION ARMS(1000)                                              USC
      COMMON RMS(1000),SRMS(1000)                                       USC
      COMMON /BLK2/TRACE(1000),N,JUMPX,NLEAK                            USC
      DATA IX/0/                                                        USC
      IF(MOD(IX,JUMPX).NE.0) GO TO 50                                   USC
      B=(1.+2.*NLEAK*NLEAK)*1024.                                       USC
      A=-NLEAK*NLEAK*1024.                                              USC
      DO 10 I=1,N                                                       USC
      Q=TRACE(I)                                                        USC
   10 RMS(I)=ABS(Q)                                                     USC
      CALL TRI(A,B,A,N,SRMS,RMS,SRMS,RMS)                               USC
      IF(IX.NE.0) GO TO 30                                              USC
      DO 20 I=1,N                                                       USC
   20 ARMS(I)=SRMS(I)                                                   USC
   30 DO 40 I=1,N                                                       USC
   40 ARMS(I)=.8*ARMS(I)+.2*SRMS(I)                                     USC
C     WRITE(6,77)(ARMS(I),I=1,500)                                      USC
C77  FORMAT(' GAIN CONTROL'/(10F12.8))                                  USC
   50 DO 60 I=1,N                                                       USC
   60 TRACE(I)=TRACE(I)/ARMS(I)                                         USC
      IX=IX+1                                                           USC
      RETURN                                                            USC
      END                                                               USC
      SUBROUTINE TRI(A,B,C,N,T,D,E,F)                                   USC
      DIMENSION T(N),D(N),E(N),F(N)                                     USC
      N1=N-1                                                            USC
      E(1)=1.0                                                          USC
      F(1)=0.                                                           USC
      DO 10 I=2,N1                                                      USC
      DEN=B+C*E(I-1)                                                    USC
      E(I)=-A/DEN                                                       USC
   10 F(I)=(D(I)-C*F(I-1))/DEN                                          USC
      T(N)=F(N1)/(1.0-E(N1))                                            USC
      DO 20 J=1,N1                                                      USC
      I=N-J                                                             USC
   20 T(I)=E(I)*T(I+1)+F(I)                                             USC
      RETURN                                                            USC
      END                                                               USC
      SUBROUTINE NMOSET(RX,C,DT,N)
      INTEGER*4 ITD(2000),IDATA(3950),BUF                               USC
      COMMON /BLK1/BUF(2000)                                            USC
      EQUIVALENCE (IDATA(1),BUF(26))                                    USC
      TD(ID)=SQRT(ID*ID + (.5*RX/(C*DT))**2)                            USC
```

```
      TMAX=TD(N)                                                        USC
      DO 10 ID=1,N                                                      USC
   10 ITD(ID)=1+(N-1)*(TD(ID)/TMAX)                                     USC
      RETURN                                                            USC
      ENTRY NMD                                                         USC
      DO 20 ID=1,N                                                      USC
   20 IDATA(ID)=IDATA(ITD(ID))                                          USC
      RETURN                                                            USC
      END                                                               USC
      SUBROUTINE DCDEC(IN,NDEC)                                         USC
      INTEGER*2 IN(NDEC,1)                                              USC
      COMMON /BLK2/OUT(1000),LOUT,N1,N2,RHO,CRHO                        USC
      OUT(1)=IN(NDEC,1)+100                                             USC
      DO 1000 K=2,LOUT                                                  USC
      KM1=K-1                                                           USC
 1000 OUT(K)=CRHO*(IN(NDEC,K)-IN(NDEC,KM1))+RHO*OUT(KM1)                USC
      RETURN                                                            USC
      END                                                               USC
      SUBROUTINE GFILT(FLOW,FHIGH,SRATE,LFILT,FLOW2,FHIGH2,LX)          USC
      REAL*4 FILT(20),FILT2(20),X(1,1000),Y(1,1000)                     USC
      COMMON /BLK2/Y                                                    USC
      COMMON /BLK3/X                                                    USC
      BW=FHIGH-FLOW                                                     USC
      AS=BW*SRATE*3.1415927                                             USC
      AC=SRATE*2.*3.1415927*(FHIGH-BW/2.)                               USC
      M=LFILT/2                                                         USC
      MID=M+1                                                           USC
      DO 10 I=1,M                                                       USC
      FILT(MID-I)=COS(AC*I)*SIN(AS*I)/(AS*I)                            USC
   10 FILT(MID+I)=FILT(MID-I)                                           USC
      FILT(MID)=1.0                                                     USC
      BW=FHIGH2-FLOW2                                                   USC
      AS=BW*SRATE*3.1415927                                             USC
      AC=SRATE*2.*3.1415927*(FHIGH2-BW/2.)                              USC
      DO 15 I=1,M                                                       USC
      FILT2(MID-I)=COS(AC*I)*SIN(AS*I)/(AS*I)                           USC
   15 FILT2(MID+I)=FILT2(MID-I)                                         USC
      FILT2(MID)=1.0                                                    USC
      NEND=LX-LFILT+1                                                   USC
      RETURN                                                            USC
      ENTRY BPASS                                                       USC
C  Y(IN) X(OUT)                                                         USC
      DO 20 I=1,LX                                                      USC
   20 X(I,1)=0.                                                         USC
      DO 30 I=1,NEND                                                    USC
      DO 30 J=1,LFILT                                                   USC
   30 X(I,J)=X(I,J)+Y(I+M,1)*FILT(J)                                    USC
      RETURN                                                            USC
      ENTRY BPASS2                                                      USC
C  X(IN) Y(OUT)                                                         USC
      DO 40 I=1,LX                                                      USC
   40 Y(I,1)=0.                                                         USC
      DO 50 I=1,NEND                                                    USC
      DO 50 J=1,LFILT                                                   USC
   50 Y(I,J)=Y(I,J)+X(I+M,1)*FILT2(J)                                   USC
      RETURN                                                            USC
      END                                                               USC
```

```
//LISTIT JOB 'C740,314','LIST PACKAGE CF'                                J
/* SERVICE LIST
//ZAP EXEC FORTHCL,PARM.FORT='OPT=2,NAME=USGSC,LINECNT=60'
//FORT.SYSIN DD *
C  U S G S  S E I S M I C  P R O C E S S I N G  P A C K A G E  CF        C
C  PROCESSES 1971 FIELD DATA RECORDED WITH TELEDYNE MODEL 27710
C        SAMPLE DD CARD FOR TAPE INPUT; ASSUME TAPEDD='FT20F001'
C     //GO.FT2CF001 DD DSN=SEIS,UNIT=0C1,VOL=SER=UXXXX,DISP=(OLD,KEEP),
C     //  LABEL=(1,BLP,,IN),DCB=(RECFM=U,BLKSIZE=16000,DEN=1,TRTCH=C)
C        REPLACE UXXXX BY COMP. CENTER TAPE I.D., LIKE U1577
C                                                                       C
      INTEGER BUF*2(4000),LVERS(18)
      REAL*4 CX(20,1000),BRHO(10),BSIG(10),PLOC(10)                    C
      REAL*8 DATE,TIME,LINEID,TAPEDD,USER                              C
      COMMON /BLK2/Y(1000),LX,NJUMP,NLEAK,RHO,CRHO,NDOTS               C
      COMMON /BLK3/X(1000)                                             C
      NAMELIST /JOB/NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,RHO,C
     1NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,NOTRJ,NJOB,TAPEDD,USER,NPAGESC
     2,RLINCH/ADAPT/FHIGH,NZC ,FHIGH2,FLOW2,LENBP,IFLGAP,LCN,DWARM,WSTRTC
     3,ZTAU,XTAU/BIFLT/NPTS,PLOC,BSIG,BRHO/JOBIN/NREEL,LINEID,RANGE,WVELC
     4,SRATE,RECEND,NDEC,ITIME,RHO,NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,C
     5NOTRJ,NJOB,TAPEDD,USER,NPAGES,RLINCH,FHIGH,NSTR,LENBP,IFLGAP,LCN, C
     6DWARM,WSTRT,ZTAU,XTAU,FHIGH2,FLOW2,NPTS,PLOC,BSIG,BRHO,NZC       C
      DATA NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,NBITS,NBITS2,C
     1NOTR,NOTRJ,NJOB,TAPEDD,FHIGH,NSTR,FLOW2,FHIGH2,LENBP,IFLGAP,LCN, C
     2DWARM,WSTRT,ZTAU,XTAU,NPTS,NPAGES,RLINCH,PLOC,BRHO,BSIG,MODADD,   C
     3USER,NZC/999,'DEFERRED',400.,4875.,.004,1.53,1,540,5,4,2000,3,3,  C
     4'FT20F001',63.,1,  15.,63.,9,2,9,.2,.2,.12,10.,2,1,7.63,0.,1.536, C
     58*0.,.5,.5,8*0.,.99,.99,8*0.0,9,'U.S.G.S.',0/                    C
C                                                                       C
C_____________LIST__OF__PRINCIPAL__VARIABLES_____________________________C
C                                                                       C
C  NAME (DEFAULT VALUE)          DEFINITION                             C
C                                                                       C
C       <PARAMETERS USED IN ALL JOBS>                                   C
C  USER   (U.S.G.S.)     USER'S NAME (8 ALPHANUMERIC)                   C
C  NREEL  (9999)         USGS REEL NUMBER (INTEGER)                     C
C  LINEID (DEFERRED)     USGS LINE NUMBER (8 ALPAHNUMERIC)              C
C  RANGE  (400.)         SHOT-RECEIVER OFFSET (FT.)                     C
C  WVEL   (4875.)        WATER VELOCITY (FT/SEC)                        C
C  SRATE  (.004)         INPUT SAMPLE RATE (SEC)                        C
C  RECEND (1.53)         RECORD LENGTH TO PROCESS (SEC)                 C
C  NDEC   (1)            DEGREE OF DECIMATION (INTEGER)                 C
C  ITIME  (540)          INTERNAL TIME ESTIMATE (SEC.,INTEGER)          C
C  RHO    (0.74)         DC REJECT FILTER COEFFICIENT                   C
C  NBITS  (5)            NUMBER OF DOTS PER TRACE (INTEGER)             C
C  NBITS2 (4)            SECONDARY TRACEWIDTH EVERY 'MODADD' TRACES (INT)C
C  MODADD (9)            MODULO TO ADD SECONDARY BITS (INTEGER)         C
C  NOTR   (2000)         NUMBER OF TRACES TO BE PROCESSED (INTEGER)     C
C  NOTRJ  (3)            PROCESSING FREQUENCY (1,2,OR 3)                C
C  NSTR   (1)            NUMBER OF STARTING TRACE ON TAPE (INTEGER)     C
C  NJOB   (3)            JOB SELECTION SWITCH (1,2,3,OR 4)              C
C  TAPEDD (FT20F001)     TAPE DATA DEFINITION TAG (8 ALPHANUMERIC)      C
C  RLINCH (7.63)         RECORD SECTION LENGTH IN INCHES (MAX.=15.25)   C
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,2,3>                      C
C  NLEAK  (10)           INTEGRATION PARAMETER FOR AGC (INTEGER)        C
C  NJUMP  (24)           UPDATE FREQUENCY OF AGC GAIN TRACE (INTEGER)   C
C  LENBP  (9)            NUMBER OF COEFFICIENTS IN BANDPASS (INTEGER,ODD)C
C  FHIGH  (63.)          PRIOR HIGH CUTOFF (CYCLES/SEC)                 C
C  FLOW2  (15.)          POST-PROC. LOW CUTOFF (CYCLES/SEC)             C
```

```
C  FHIGH2 (63.)          POST-PROC. HIGH CUTOFF (CYCLES/SEC)                  CF
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,3>                              CF
C  NPTS    (2)          NO. POINTS ON BIFILT PROFILE (INTEGER)               CF
C  PLOC    (0.,1.53)    LOCATION OF PROFILE TIES (SEC)                       CF
C  BRHO    (.5,.5)      RHO PROFILE COEFFICIENTS                             CF
C  BSIG    (.99,.99)    SIGMA PROFILE COEFFICIENTS                           CF
C       <ADDITIONAL PARAMETERS USED IN JOBS 2,3>                              CF
C  NZC     (0)          NO. OF LEADING ZERO REFLECTION COEFF.(INT)           CF
C  IFLGAP  (2)          GAP BETWEEN ADAPTIVE FILTER COEFFICIENTS (INT)       CF
C  LCN     (9)          NO. OF REFLECTION COEFF. IN LADDER (INTEGER)         CF
C  DWARM   (.2)         DURATION OF STATIONARY WARM-UP CYCLE (SEC)           CF
C  WSTRT   (.2)         STARTING POSITION OF WARM-UP (SEC)                   CF
C  ZTAU    (.12)        RELAXATION TIME TO 1/E (SEC)                         CF
C  XTAU    (10.)        RELAXATION DISTANCE TO 1/E (TRACES)                  CF
C                                                                            CF
C              (ALL VARIABLES ARE REAL EXCEPT AS INDICATED)                  CF
C                                                                            CF
C-------------------------------------------------------------------------- CF
C                                                                            CF
C           PROCESSING SEQUENCES:                                            CF
C                                                                            CF
C       NJOB=1   /FIELD/   /DCDEC/   /BPASS/BIFILT/BPASS2/SEISGM/            CF
C       NJOB=2   /FIELD/   /DCDEC/   /BPASS/BAFL/BPASS2/SEISGM/              CF
C       NJOB=3   /FIELD/   /DCDEC/   /BPASS/BAFL/BIFILT/BPASS2/SEISGM/       CF
C       NJOB=4   /FIELD/   /DCDEC/SEISGM/                                    CF
C-------------------------------------------------------------------------- CF
C           VERSION G                          DON C. RILEY DECEMBER 1972    CF
C                                                                            CF
      CALL MCLOCK(DATE,TIME,WEEKDA)                                          CF
      CALL PCLOCK(IJS)                                                       CF
      RHO=0.74                                                               CF
      NLEAK=10                                                               CF
      NJUMP=24                                                               CF
      WRITE(6,876)                                                           CF
      READ(5,JOBIN,ERR=977)                                                  CF
    2 LENGTH=RECEND/SRATE                                                    CF
      IF(NSTR.LE.1) GO TO 4                                                  CF
      M=NSTR-1                                                               CF
      DO 3 I=1,M                                                             CF
    3 CALL NCERRD(BUF,LEN,IERR,TAPEDD)                                       CF
    4 LX=LENGTH/NDEC                                                         CF
      RLINCH=AMIN1(RLINCH,15.3)                                              CF
      IF(RLINCH.GT.7.65) NPAGES=2                                            CF
      SRATE2=SRATE*NDEC                                                      CF
      CRHO=(1.+RHO)/2.                                                       CF
      KTEST=24*NOTRJ                                                         CF
      WRITE(6,JOB)                                                           CF
      IF(NJOB.EQ.4)  GO TO 40                                                CF
      FLOW=-FHIGH                                                            CI
      IF(MOD(LENBP,2).EQ.0)  LENBP=LENBP+1                                   CI
      CALL GFILT(FLOW,FHIGH,SRATE2,LENBP,FLOW2,FHIGH2,LX)                    CI
      IF(NJOB.EQ.1) GO TO 30                                                 CI
      WRITE(6,ADAPT)                                                         CI
      NSTRT=WSTRT/SRATE2                                                     CI
      NWARM=DWARM/SRATE2                                                     CI
      ZTAT=ZTAU/SRATE2                                                       CI
      IF(NJOB.EQ.2) GO TO 40                                                 C
   30 WRITE(6,BIFLT)                                                         CI
      CALL BIFILT(LX,BSIG,BRHO,PLOC,NPTS,SRATE2)                             C
   40 CALL TAXIS(LX,SRATE2,NPAGES,RLINCH)                                    C
```

```
CON    CALL NMOSET(RANGE,WVEL,SRATE,LENGTH)                                  C
       ITIME=ITIME*100                                                       C
       CALL FI0999(44,LVERS(1),18,18)                                        C
C                                                                            C
       GO TO (1727,1707,1747,1737),NJOB                                      C
C                                                                            C
C                    NJOB=2                                                  C
C                                                                            C
 1707  CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                      C
       CALL DCDEFT(BUF)
COUT   CALL AGC                                                              C
       CALL BPASS                                                            C
       CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)                 C
       CALL BPASS2                                                           C
       NDOTS=NBITS                                                           C
       CALL SEISGM                                                           C
       DO 707 I=NOTRJ,NOTR,NOTRJ                                             C
       NDOTS=NBITS                                                           C
       IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                                  C
       CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                      C
       IF(IERR.EQ.1) GO TO 988                                               C
       CALL DCDECT(BUF)
COUT   CALL AGC                                                              C
       CALL BPASS                                                            C
       IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                     C
       IF(IERR.EQ.1) GO TO 988                                               C
       CALL BAFLGO                                                           C
       IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                     C
       CALL BPASS2                                                           C
       CALL SEISGM                                                           C
       IF(MOD(I,KTEST).NE.0) GO TO 707                                       C
       CALL PCLOCK(IJE,IJS)                                                  C
       IF(IJE.GT.ITIME) GO TO 909                                            C
  707  CONTINUE                                                              C
       GO TO 999                                                             C
C                                                                            C
C                                                                            C
C                    NJOB=1                                                  C
C                                                                            C
 1727  DO 727 I=NOTRJ,NOTR,NOTRJ                                             C
       NDOTS=NBITS                                                           C
       IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                                  C
       CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                      C
       IF(IERR.EQ.1) GO TO 988                                               C
       CALL DCDECT(BUF)
COUT   CALL AGC                                                              C
       CALL BPASS                                                            C
       CALL BIGOT                                                            C
       IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                     C
       IF(NOTRJ.GE.3) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)                     C
       IF(IERR.EQ.1) GO TO 988                                               C
       CALL BPASS2                                                           C
       CALL SEISGM                                                           C
       IF(MOD(I,KTEST).NE.0) GO TO 727                                       C
       CALL PCLOCK(IJE,IJS)                                                  C
       IF(IJE.GT.ITIME) GO TO 909                                            C
  727  CONTINUE                                                              C
       GO TO 999                                                             C
C                                                                            C
C                    NJOB=4                                                  C
```

```
C                                                                       C
 1737 DO 737 I=NOTRJ,NOTR,NOTRJ                                         C
      NDOTS=NBITS                                                       C
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              C
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  C
      IF(IERR.EQ.1) GO TO 988                                           C
      CALL DCDEFT(BUF)
      IF(NOTRJ.GE.2) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                C
      IF(NOTRJ.GE.3) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                C
      IF(IERR.EQ.1) GO TO 988                                           C
      CALL SETSGM                                                       C
      IF(MOD(I,KTEST).NE.0) GO TO 737                                   C
      CALL PCLOCK(IJE,IJS)                                              C
      IF(IJE.GT.ITIME) GO TO 909                                        C
  737 CONTINUE                                                          C
      GO TO 999                                                         C
C                                                                       C
C                   NJOB=3                                              C
C                                                                       C
 1747 CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  C
      CALL DCDEFT(BUF)
COUT  CALL AGC                                                          C
      CALL BPASS                                                        C
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)             C
      CALL BIGOT                                                        C
      CALL BPASS2                                                       C
      NDOTS=NBITS                                                       C
      CALL SETSGM                                                       C
      DO 747 I=NOTRJ,NOTR,NOTRJ                                         C
      NDOTS=NBITS                                                       C
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2                              C
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)                                  C
      IF(IERR.EQ.1) GO TO 988                                           C
      CALL DCDEFT(BUF)
COUT  CALL AGC                                                          C
      CALL BPASS                                                        C
      IF(NOTRJ.GE.2) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                C
      IF(IERR.EQ.1) GO TO 988                                           C
      CALL BAFLGO                                                       C
      CALL BIGOT                                                        C
      IF(NOTRJ.GE.3) CALL NOERRD(   BUF,LEN,IERR,TAPEDD)                C
      CALL BPASS2                                                       C
      CALL SETSGM                                                       C
      IF(MOD(I,KTEST).NE.0) GO TO 747                                   C
      CALL PCLOCK(IJE,IJS)                                              C
      IF(IJE.GT.ITIME) GO TO 909                                        C
  747 CONTINUE                                                          C
C                                                                       C
C                                                                       C
C                                                                       C
  999 CALL PCLOCK(IJE,IJS)                                              C
      WRITE(6,900) IJE                                                  C
      CALL XCLOCK(IJS)                                                  C
      WRITE(44,622) USER                                                C
      DO 5 K=1,19                                                       C
    5 CALL VLINE(LVERS,70)                                              C
      CALL VLINE(' ',1)                                                 C
      CALL VLINE(' ',1)                                                 C
C                                                                       C
C     WRITE PARM.JOB TO VERSATEC.....                                   C
```

```
      WRITE(44,600)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,601) NREEL                                               CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,602) LINEID                                              CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,603)                                                     CP
      GO TO (50,60,70,45),NJOB                                          CP
   45 WRITE(44,621)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      GO TO 80                                                          CP
   50 WRITE(44,614)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,615)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      GO TO 80                                                          CP
   60 WRITE(44,604)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,605)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      GO TO 80                                                          CP
   70 WRITE(44,616)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,617)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
   80 WRITE(44,603)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,606) RANGE,WVEL,SRATE,NSTR                               CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,609) RLINCH,NLEAK,NJUMP,NBITS,MODADD                     CP
      CALL VLINE(LVERS,70)                                              CP
      IF(NJOB.EQ.4)  GO TO 120                                          CP
      IF(NJOB.EQ.1) GO TO 110                                           CP
      WRITE(44,607) FLOW2,FHIGH2,LENBP,IFLGAP,LCN,NZC                   CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,608) DWARM,WSTRT,ZTAU,XTAU,RHO                           CP
      CALL VLINE(LVERS,70)                                              CP
      IF(NJOB.EQ.2) GO TO 120                                           CP
  110 WRITE(44,618) NPTS,PLOC                                           CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,619) BSIG                                                CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,620) BRHO                                                CP
      CALL VLINE(LVERS,70)                                              CP
  120 WRITE(44,610) NOTR,NOTRJ,RECENC,NDEC,NPAGES                       CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,603)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,612)  IJE,DATE,TIME                                      CP
      CALL VLINE(LVERS,70)                                              CP
      WRITE(44,600)                                                     CP
      CALL VLINE(LVERS,70)                                              CP
C                                                                       CP
C                                                                       CP
      CALL VLINE(' ',1)                                                 CP
      CALL VLINE(' ',1)                                                 CP
      CALL VLINE('                    T W O - W A Y  T R A V E L  T I M E',CP
     1 152)                                                             CP
      CALL VLINE(' ',1)                                                 CP
      CALL VLINE(' ',1)                                                 CP
```

```
      CALL XCLOCK(IJE,IJS)                                                  C
      WRITE(6,879) IJE                                                      C
      CALL XCLOCK(IJS)                                                      C
      CALL SDUMP                                                            C
      CALL XCLOCK(IJE,IJS)                                                  C
      WRITE(44,613) IJE                                                     C
      WRITE(6,613)  IJE                                                     C
      CALL VLINE(LVERS,70)                                                  C
      CALL VLINE('                          END OF PLOT         BIN 314',49)C
      READ(5,JOBIN,ERR=977,END=9999)                                        C
      WRITE(6,877)                                                          C
      CALL MCLOCK(DATE,TIME,WEEKDA)                                         C
      CALL PCLOCK(IJS)                                                      C
      CALL ENDTAP                                                           C
      REWIND 1                                                              C
      GO TO 2                                                               C
 9999 STOP                                                                  C
  988 WRITE(6,880) I                                                        C
      GO TO 999                                                             C
  909 WRITE(6,878) I                                                        C
      WRITE(44,878) I                                                       C
      CALL VLINE(LVERS,70)                                                  C
      GO TO 999                                                             C
  977 WRITE(6,881)                                                          C
      GO TO 9999                                                            C
C                                                                           C
C        F O R M A T S                                                      C
  600 FORMAT(70('*'))                                                       C
  601 FORMAT('*',8X,'CHUKCHI SEA 1971 SEISMIC DATA  U.S.G.S. REEL NO.',     C
     1 I4,8X,'*')                                                           C
  602 FORMAT('*',10X,'SINGLE-FOLD MARINE REFLECTION PROFILE NO.',A8,9X,     C
     1 '*')                                                                 C
  603 FORMAT('*',30X,'<<<<>>>>',30X,'*')                                    C
  604 FORMAT('*',15X,'-TIME & SPACE ADAPTIVE DECONVOLUTION-',16X,'*')       C
  605 FORMAT('*',6X,'PROCESSING SEQUENCE: /FIELD/NMO/DCDEC/AGC/BAFLGO/SEC   C
     1ISGM/',6X,'*')                                                        C
  606 FORMAT('*   PARM.JOB=',1H','RANGE=',F5.0,',WVEL=',F6.0,',SRATE=',     C
     1 F6.3,',NSTR=',I4,',          *')                                    C
  607 FORMAT('*',13X,'FLOW2=',F3.0,',FHIGH2=',F3.0,',LENBP=',I2,',IFLGAPC   C
     1=',I1,',LCN=',I2,',NZC=',I1,',    *')                              C
  608 FORMAT('*',13X,'DWARM=',F3.2,',WSTRT=',F3.2,',ZTAU=',F5.3,',XTAU='C   C
     1,F5.1,',RHO=',F3.2,',',5X,'*')                                        C
  609 FORMAT('*',13X,'RLINCH=',F4.1,',NLEAK=',I3,',NJUMP=',I4,',NBITS=',C   C
     1 I2,',MODADD=',I2,',   *')                                           C
  610 FORMAT('*',13X,'NOTR=',I4,',NOTRJ=',I1,',RECEND=',F4.2,',NDEC=',     C
     1I1,',NPAGES=',I1,1H',9X,'*')                                          C
  612 FORMAT ('*  360/67 CPU TIME= ',I5,' CSEC.  DATE: ' A8,2X,             C
     1 A8,'  STANFORD  *')                                                  C
  613 FORMAT('   DRUM.DUMP.PLAYBACK.TIME = ',I7,' CSEC.')                  C
  614 FORMAT('*',23X,'-HORIZONTAL BIFILTER-',24X,'*')                       C
  615 FORMAT ('*',6X,'PROCESSING SEQUENCE: /FIELD/NMO/DCDEC/AGC/BIFILT/SC   C
     1EISGM/',6X,'*')                                                       C
  616 FORMAT('*',10X,'-TIME & SPACE ADAPTIVE DECONVOLUTION + BIFILTER-',C   C
     110X,'*')                                                              C
  617 FORMAT('*',5X,'PROCESSING SEC: /FIELD/NMO/DCDEC/AGC/BAFLGO/BIFILT/C   C
     1SEISGM/',5X,'*')                                                      C
  618 FORMAT('*',13X,'NPTS=',I2,',LOC=',10(F3.1,','),T70,'*')               C
  619 FORMAT('*',20X,'BSIG=',10(F3.2,','),T70,'*')                          C
  620 FORMAT('*',20X,'BPHC=',10(F3.2,','),T70,'*')                          C
  621 FORMAT('*',9X,'SEISMIC PLAYBACK ONLY SEQ: /FIELD/NMO/DCDEC/SEISGM/C   C
```

```
     1',8X,'*')                                                          C
  622 FORMAT('START OF PLOTJOB ',A8,', BIN 314')                         C
  876 FORMAT(1H0,30X,10('*'),' BEGIN CHUKCHI JOB A ',10('*'))            C
  877 FORMAT(1H1,30X,10('*'),' BEGIN CHUKCHI JOB B ',10('*'))            C
  878 FORMAT(1H0,'EST. TIMER SELF-EXIT AT TR. ',I8)                      C
  879 FORMAT(1H ,'W-BLOCK TIMER:',I8)                                    C
  880 FORMAT(1H ,'***END OF FILE ENCOUNTERED AT TR. NO.',I6)             C
  881 FORMAT(1H0,'***ERROR IN INPUT CARDS *** ABNEOJ ***')               C
  900 FORMAT(1H0,7X,'PARTITION TIMER IN CSEC. ',I8)                      C
C                                                                       C
      END                                                               C
      SUBROUTINE BAFL(LOUT,CX,IFLGAP,LCN,NWARM,ISTRT,ZTAU,XTAU,NZERO)   C
C                                                                       C
C                                                                       C
C-------------S U B R O U T I N E   B A F L---------------             C
C                                                                       C
C                                                                       C
C       THE BURG ADAPTIVE FILTER LADDER: AN ADAPTIVE OR TIME-VARYING    C
C       FIXED-LEAD PREDICTION ERROR PROCESSOR.  ADJUSTMENT OF EACH      C
C       REFLECTION COEFFICIENT IS MADE EVERY JUMP STATE ATTEMPTING      C
C       TO MINIMIZE THE STAGE OUTPUT POWER.                             C
C       INPUTS:                                                         C
C         X(1)...X(LX)=INITIAL DATA                                     C
C         LCN= LAST NON-ZERO REFLECTION COEFFICIENT                     C
C         IFLGAP=  NUMBER OF GAPS BETWEEN FILTER COEFFICIENTS           C
C             SETTING IFLGAP=0 DOES NOT GAP THE F.C. AND                C
C             TRIES TO OPERATE ON THE ENTIRE SPECTRUM FROM              C
C             0 TO W WHERE W IS THE FOLDING FREQUENCY.                  C
C             SETTING IFLGAP=1 OPERATES ON THE PORTION OF               C
C             THE SPECTRUM FROM 0 TO W/2 , IFLGAP=2 FROM                C
C             0 TO W/3 ETC. THUS ALLOWING SPECIFICATION OF              C
C             WHAT PART OF THE SPECTRUM TO DECONVOLVE.                  C
C                                                                       C
C         NWARM=DURATION OF STATIONARY C ESTIMATION CYCLE               C
C         ISTRT=START OF STATIONARY GATE                                C
C         ZTAU=TEMPORAL RELAXATION TIME TO 1/E                          C
C         XTAU=SPATIAL RELAXATION DISTANCE TO 1/E                       C
C       OUTPUTS:                                                        C
C         X(1)...X(LOUT)=FORWARD ERROR PREDICTION TRACE                 C
C       OTHER VARIABLES:                                                C
C         F(1)...F(LCN)=FORWARD STATE VECTOR                            C
C         B(1)...B(LCN)=BACKWARD STATE VECTOR                           C
C         C(1)...C(LCN)=REFLECTION COEFFICIENTS AT EACH STATE           C
C         CX=REFLECTION COEFF. INTEGRATED IN SPACE & TIME               C
C         DEN(1)...DEN(LCN)=  STAGE AUTOPOWER                           C
C         NUM(1)...NUM(LCN)= STAGE CROSSPOWER                           C
C                                                                       C
C         CALLING 'BAFL' FIRST SETS UP THE LOOPING AND PASSING          C
C         ARRAYS FOR THE PARTICULAR PROBLEM AS SPECIFIED BY LCN &       C
C         IFLGAP.  THEN IT COMPUTES A SHORT (LENGTH=NWARM) ESTIMATE     C
C         OF THE REFLECTION COEFFICIENT SERIES IN ORDER TO START THE     C
C         ADAPTION OUT WITH SOME REASONABLE NUMBERS. THEN IT LOADS UP    C
C         THE CX ARRAY WITH THE INITIAL VALUES AND PASSED INTO ENTRY     C
C         'BAFLGO'.                                                     C
C                                                                       C
C         THE USUAL ENTRY IS 'BAFLGO' WHICH FIRST INITIALIZES THE       C
C         BACKWARD ARRAY THEN PASSES TO THE MAIN ALGORITHM.             C
C         THE CX SERIES IS UPDATED EVERY IFLGAP DATA POINTS AND IN      C
C         THE INTERMEDIATE STEPS THE OUTPUT ARE INTERPOLATED OR         C
C         PROCESSED AS THOUGH THE R.C. WERE STATIONARY.                 C
```

```
C                                                                       CF
C                                                                       CF
C                                                                       CF
C                                                                       CF
      REAL NUM(50),DEN(50),B(50),F(50),C(50),EM(1000),EP(1000),         CF
     1 CX(LCN,LOUT),A(50)                                               CF
      COMMON /BLK3/X(1000)                                              CF
      DATA A,B,C/150*0.0/                                               CF
      FTEST=1.00                                                        CF
      LCNP1=LCN+1                                                       CF
      LCNP2=LCN+2                                                       CF
      IFGM1=IFLGAP-1                                                    CF
      LBSP=LOUT-IFLGAP                                                  CF
      NZERP1=NZERO+I                                                    CF
      NEND=LCN-NZERP1                                                   CF
      DO 80 K=LBSP,LOUT                                                 CF
   80 EP(K)=0.                                                          CF
C BEGIN STATIONARY WARM-UP                                              CF
      DO 10 I=1,NWARM                                                   CF
      EM(I)=X(I*IFLGAP+ISTRT)                                           CF
   10 EP(I)=X(I*IFLGAP+ISTRT)                                           CF
C                                                                       CF
      DO 11 J=2,LCNP1                                                   CF
      DEN(J)=0.                                                         CF
      NUM(J)=0.                                                         CF
      DO 12 I=J,NWARM                                                   CF
      DEN(J)=DEN(J)+EP(I)*EP(I)+EM(I-J+1)*EM(I-J+1)                     CF
   12 NUM(J)=NUM(J)+EP(I)*EM(I-J+1)                                     CF
      DIV=NWARM-J+1                                                     CF
      NUM(J)=NUM(J)/DIV                                                 CF
      DEN(J)=DEN(J)/DIV                                                 CF
      C(J)=-2.*NUM(J)/DEN(J)                                            CF
      DO 11 I=J,NWARM                                                   CF
      EPI=EP(I)                                                         CF
      EP(I)=EPI+C(J)*EM(I-J+1)                                          CF
      KS=IFLGAP*LCN+1                                                   CF
   11 EM(I-J+1)=EM(I-J+1)+C(J)*EPI                                      CF
      DO 8 J=  1,LCN                                                    CF
      DO 8 K=1,KS                                                       CF
    8 CX(J,K)=C(J+1)                                                    CF
      DO 33 J=1,LCN                                                     CF
      DEN(J)=DEN(J+1)                                                   CF
   33 NUM(J)=NUM(J+1)                                                   CF
C END WARM-UP CYCLE                                                     CF
C SET RELAXATION TIMES                                                  CF
      CLX=EXP(-1./XTAU)                                                 CF
      CL=EXP(-1./ZTAU)                                                  CF
      CRX=1.-CLX                                                        CF
      CR=1.-CL                                                          CF
C                                                                       CF
C-----------------USUAL--ENTRY------------------------------------------CF
      ENTRY BAFLGO                                                      CF
C                                                                       CF
C           INITIALIZE BACKWARD VECTOR                                  CF
      B(1)=X(KS-IFLGAP)                                                 CF
      A(1)=1.0                                                          CF
      DO 100C J=2,LCN                                                   CF
      B(J)=X(KS-J*IFLGAP)                                               CF
      A(J)=CX(J,KS)                                                     CF
      DO 1000 I=2,J                                                     CF
```

```
      A(I)=A(I)+CX(J,KS)*A(J-I+1)                                       CF
 1000 B(J)=B(J)+A(I)*X(KS+(I-J-1)*IFLGAP)                               CF
C                                                                       CF
C--------BEGIN--MAIN--LOOP-------------------------------------------   CF
C                                                                       CF
      DO 5000 K=KS,LOUT,IFLGAP                                          CF
      Z=X(K)                                                            CF
      DO 1010 J=1,NZERP1                                                CF
 1010 F(J)=Z                                                            CF
      DO 2000 J=NZERP1,LCN                                              CF
      DEN(J)=(F(J)**2+B(J)**2)*DR+DEN(J)*DL                             CF
      NUM(J)=F(J)*B(J)*DR+NUM(J)*DL                                     CF
      IF(FTEST.LE.1.1) CX(J,K)=-2.*NUM(J)/DEN(J)                        CF
      CX(J,K)=-2.*DRX*NUM(J)/DEN(J)+CX(J,K)*DLX                         CF
 2000 F(J+1)=F(J)+CX(J,K)*B(J)                                          CF
      X(K)=F(LCNP1)                                                     CF
      DO 3000 JR=1,NEND                                                 CF
      J=LCN-JR                                                          CF
 3000 B(J+1)=B(J)+CX(J,K)*F(J)                                          CF
      IF(NZERO.EQ.0) GO TO 5000                                         CF
      DO 4000 JR=1,NZERO                                                CF
      J=NZERP1-JR                                                       CF
 4000 B(J+1)=B(J)                                                       CF
 5000 B(1)=Z                                                            CF
C                                                                       CF
C--------END--OF--MAIN--LOOP-------------------------------------------   CF
C                                                                       CF
C       NOW GO BACK AND FILL IN THE GAPS....                            CF
      IF(IFLGAP.EQ.1) GO TO 9050                                        CF
      DO 9000 L=1,IFGM1                                                 CF
      KSTART=KS+L                                                       CF
      DO 9000 K=KSTART,LOUT,IFLGAP                                      CF
      KC=K-L                                                            CF
      Z=X(K)                                                            CF
      DO 6000 J=1,NZERP1                                                CF
 6000 F(J)=Z                                                            CF
      DO 7000 J=NZERP1,LCN                                              CF
 7000 F(J+1)=F(J)+CX(J,KC)*B(J)                                         CF
      X(K)=F(LCNP1)                                                     CF
      DO 8000 JR=1,NEND                                                 CF
      J=LCN-JR                                                          CF
 8000 B(J+1)=B(J)+CX(J,KC)*F(J)                                         CF
      IF(NZERO.EQ.0) GO TO 9000                                         CF
      DO 8050 JR=1,NZERO                                                CF
      J=NZERP1-JR                                                       CF
 8050 B(J+1)=B(J)                                                       CF
 9000 B(1)=Z                                                            CF
C                                                                       CF
 9050 FTEST=FTEST+1.0                                                   CF
      RETURN                                                            CF
      END                                                               CF
      SUBROUTINE SEISGM                                                 CF
C                                                                       CF
C       SUBROUTINE 'SEISGM'   VARIABLE AREA SEISMIC SECTION PLOTTER.    CF
C       THROUGH MULTIPLE CALLS TO SEISGM ADJACENT TRACES ARE WRITTEN    CF
C       TO THE VERSATEC FILLING THE FULL-PAGE WIDTH (560 BITS).         CF
C       DATA: INPUT DATA SERIES                                         CF
C       NT = LENGTH OF INPUT DATA                                       CF
C       NBITS= WIDTH OF VARIABLE AREA TRACE IN VERSATEC BITS            CF
C       NPAGES: WHEN NPAGES=1 THE RECORD SECTION FILLS ONE PAGE         CF
```

```
C                 OF THE VERSATEC (7.536 IN.) FOR ANY LENGTH RECORDS.     CF
C                 WHEN NPAGES=2 A SPECIAL OPTION IS INVOKED ALLOWING      CF
C                 SOMEWHAT MORE GENERAL DISPLAY AS FOLLOWS.....           CF
C                 'RLINCH' INDICATES THE DESIRED RECORD SECTION LENGTH    CF
C                 IN INCHES. THUS THE RECORDS OF LENGTH NT ARE SCALED     CF
C                 INTO A VARIABLE OBJECT SPACE THAT MAY TAKE UP MORE      CF
C                 THAN ONE PAGE-WIDTH.  THIS IS ACCOMPLISHED BY COMPOSINGCF
C                 THE PLOT INTO A LONGER BUFFER THAN THE VERSATEC CAN     CF
C                 HANDLE(WHICH IS 70 BYTES) AND DUMPING THIS PLOTTED      CF
C                 DATA ONTO A 2301 DRUM (FT UNIT 1) .  THEN WE REWIND ANDCF
C                 FETCH AND WRITE THE FIRST 70 BYTES ON ONE PAGE; THEN    CF
C                 GO BACK AND FETCH AND WRITE THE NEXT 70 BYTES OR        CF
C                 LESS.                                                   CF
C                                                                         CF
C     'TAXIS' MUST BE CALLED BEFORE 1ST 'SEISGM' CALL SINCE               CF
C     A REFERENCE TABLE MUST BE GENERATED.                                CF
C                                                                         CF
      INTEGER*4 IREF(1120)                                                CF
      LOGICAL*4 LMZ,LZ,LASK,LINE(35,10),LOAD(18),LOAD2(35),LBOT*1(72)     CF
      LOGICAL*4 LTAX(35),LIGHT(35),DARK(35),LMASK(32)                     CF
      COMMON /BLK2/DATA(1000),NT,N5,N6,R5,R6,NBITS                        CF
      EQUIVALENCE (LBOT(1),LOAD2(18))                                     CF
      DATA LMZ,LZ,LIGHT,DARK/Z80000000,36*Z00000000,35*ZFFFFFFFF/         CF
      DATA LMASK/Z80000000,Z40000000,Z20000000,Z10000000,Z08000000,       CF
     1Z04000000,Z02000000,Z01000000,Z00800000,Z00400000,Z00200000,        CF
     2Z00100000,Z00080000,Z00040000,Z00020000,Z00010000,Z00008000,        CF
     3Z00004000,Z00002000,Z00001000,Z00000800,Z00000400,Z00000200,        CF
     4Z00000100,Z00000080,Z00000040,Z00000020,Z00000010,Z00000008,        CF
     5Z00000004,Z00000002,Z00000001/                                      CF
      NMAG=MIN0(10,NBITS)                                                 CF
      SUM=0.                                                              CF
      DO 121 K=1,NT,4                                                     CF
      T=DATA(K)                                                           CF
  121 SUM=SUM+ABS(T)                                                      CF
      ERMS=4.*SUM/NT                                                      CF
C     WRITE(6,900) ERMS                                                   CF
C 900 FORMAT(1H+,100X,F11.1)                                              CF
      SCALE=0.5*NMAG/ERMS                                                 CF
      IBIAS=(NMAG+1)/2                                                    CF
      DO 20 I=1,JBYTES                                                    CF
      DO 20 J=1,NMAG                                                      CF
C    SETTING = LTAX WRITES TIME MARKS ACROSS ENTIRE SECTION,LZ DOESN'T    CF
C   20 LINE(I,J) = LZ                                                     CF
   20 LINE(I,J)=LTAX(I)                                                   CF
      DO 40 IBIT=1,32                                                     CF
      LASK=LMASK(IBIT)                                                    CF
      IBYTE = 1                                                           CF
      DO 40 IL=IBIT,JBITS,32                                              CF
      IMAG=IBIAS+DATA(IREF(IL))*SCALE                                     CF
      IF(IMAG.LE.0) GO TO 40                                              CF
      IMAG=MIN0(NMAG,IMAG)                                                CF
      DO 30 I=1,IMAG                                                      CF
   30 LINE(IBYTE,I) = LINE(IBYTE,I).OR.LASK                               CF
   40 IBYTE = IBYTE + 1                                                   CF
      DO 50 I=1,NMAG                                                      CF
   50 WRITE(1)  (LINE(K,I),K=1,JBYTES)                                    CF
      LKNT = LKNT + NMAG                                                  CF
      RETURN                                                              CF
C                                                                         CF
C        ENTRY 'TAXIS' WRITES A TIME SCALE TO THE VERSATEC.               CF
```

```
C         SRATE IS THE SAMPLING RATE.  TICK MARKS ARE AT 100 MILS.
C
      ENTRY TAXIS(NT,SRATE,NPAGES,RLINCH)
      JBYTES=18
      JBITS=560
      IF(NPAGES.LE.1) GO TO 55
      JBYTES=35
      JBITS=MIN1(1120.,RLINCH*73.25)
   55 SCALE=(JBITS-1)/(NT*SRATE)
      ITMAX=SRATE*NT*10. + 1.
      TYME=0.
      DO 60 I=1,JBYTES
   60 LTAX(I)=LZ
      DO 70 I=1,ITMAX
      IR=SCALE*TYME + 1.0
      IBYTE=(IR-1)/32  + 1
      IBIT=IR - (IBYTE-1)*32
      LASK=LMASK(IBIT)
      LTAX(IBYTE)=LTAX(IBYTE).OR.LASK
   70 TYME=TYME + 0.1
      SCALE=(NT-1.)/(JBITS-1.)
      DO 90 IL=1,JBITS
   90 IREF(IL)=SCALE*(IL-1) + 1.49999
      DO 83 I=1,8
   83 WRITE(1) LIGHT
      WRITE(1) DARK
      DO 84 I=1,10
   84 WRITE(1) LTAX
      WRITE(1) DARK
      DO 81 I=1,10
   81 WRITE(1) LIGHT
      LKNT=30
C
      RETURN
C
C         ENTRY SDUMP DUMPS THE DRUM TO THE VERSATEC
C
      ENTRY SDUMP
      DO 85 I=1,380
   85 WRITE(1) LIGHT
      REWIND 1
      NL=LKNT + 378
      NFIN=NL
      IF(NPAGES.GT.1) NFIN=NL-340
      DO 82 I=1,NFIN
      READ(1) LOAD
   82 CALL WRITER(LOAD,70)
      IF(NPAGES.LE.1) GO TO 87
      REWIND 1
      DO 86 I=1,NL
      READ(1) LOAD2
   86 CALL WRITER(LBOT(3),70)
   87 CONTINUE
      RETURN
      END
      SUBROUTINE BIFILT(ND,SIG,RHO,PLOC,NPTS,SRATE2)
C                 BILINER KX FILTER
C                 USEFUL FOR EXTINGUISHING HORIZONTAL PRIMARIES
C                 AND MULTIPLES INTERFERING WITH DIPPING REFLECTORS
C                 AND/OR FOR STACKING IN  X  TO ENHANCE LATERAL
```

```
C                  CORRELATION IN THE PRESENCE OF ADDITIVE NOISE.
C                  ALGORITHM OPERATES IN TIME VARYING MODE BETWEEN
C                  PURE  KX  LOWCUT (RHO=0) AND PURE  KX  HIGHCUT
C                  (SIG=1) AS SET IN THE RHO/SIG PROFILES.
C
C              PARAMETER LIST:
C              IN      INPUT TRACE
C              OUT     OUTPUT TRACE
C              ND      LENGTH OF INPUT/OUTOUT TRACE
C              SIG     SIGMA PARAMETER PROFILE
C              RHO     RHO PARAMETER PROFILE
C              LOC     LOCATION POINTS OF PARAMETER CHANGES
C              NPTS    NO. OF SIG/RHO/LOC POINTS
C
C                RHO                    SIGMA
C     LOC1----------X-------   --X--------------
C         |          . RHO1       |   .
C         |         .             |    .
C     LOC2|        XRHO2          |     XSIG2
C         |         .             |       .
C     LOC3|        XRHO3          |       XSIG3
C         |       .               |         .
C         |     .                 |           .
C     LOC4|    XRHO4              |           XSIG4
C         |                       |
C
      REAL*4 SIG(1),RHO(1),PLOC(1),D(1000),E(1000),F(1000)
      REAL*4 IN,OUTH(1000),OUTHH(1000),INH(1000)
      INTEGER*4 LOC(10)
      COMMON /BLK3/IN(1000)
      LOC(1)=1
      DO 5 K=2,NPTS
      LOC(K)=PLOC(K)/SRATE2
      IF(LOC(K-1).GT.LOC(K)) GO TO 97
    5 CONTINUE
      LOC(NPTS)=ND
      NPM1=NPTS-1
      WRITE(6,800) RHO(1),SIG(1)
  800 FORMAT(1HO,6X,'PARAMETER PROFILE'//1H ,3X,'DEPTH',
     1 4X,'RHO',5X,'SIGMA'//1H ,6X,'1',2(3X,F6.3))
      DO 10 IW=1,NPM1
      WRITE(6,805) LOC(IW+1),RHO(IW+1),SIG(IW+1)
  805 FORMAT(1H ,3X,I4,2(3X,F6.3))
      FMR=(RHO(IW+1)-RHO(IW))/(LOC(IW+1)-LOC(IW))
      FMS=(SIG(IW+1)-SIG(IW))/(LOC(IW+1)-LOC(IW))
      NEND=LOC(IW+1) - 1
      NSTRT=LOC(IW)
      IF(NSTRT.GT.NEND) GO TO 97
      DO 10 K=NSTRT,NEND
      R=RHO(IW) + FMR*(K-NSTRT)
      S=SIG(IW) + FMS*(K-NSTRT)
      D(K)=0.5*(1.+S)*(1.-R)
      E(K)=S+R
   10 F(K)=S*R
      E(ND)=SIG(NPTS)+RHO(NPTS)
      F(ND)=SIG(NPTS)*RHO(NPTS)
      D(ND)=0.5*(1.+SIG(NPTS))*(1.-RHO(NPTS))
      DO 20 K=1,ND
      OUTH(K)= 0.0
      INH(K)= 0.0
```

```
   20 OUTHH(K)=0.0                                                      C
      RETURN                                                            C
C                                                                       C
      ENTRY BIGOT                                                       C
      DO 70 K=1,ND                                                      C
      TEMP=D(K)*(IN(K)-INH(K))+E(K)*OUTH(K)-F(K)*OUTHH(K)               C
      OUTHH(K)=OUTH(K)                                                  C
      OUTH(K)=TEMP                                                      C
      INH(K)=IN(K)                                                      C
   70 IN(K)=TEMP                                                        C
      RETURN                                                            C
   97 WRITE(6,900)                                                      C
  900 FORMAT(1HO,'***PARM.RHO/SIG ERROR***ABNFOJ***')                   C
      STOP                                                              C
      END                                                               C
      SUBROUTINE GFILT(FLOW,FHIGH,SRATE,LFILT,FLOW2,FHIGH2,LX)          C
      REAL*4 FILT(20),FILT2(20),X(1,1000),Y(1,1000)                     C
      COMMON /BLK2/Y                                                    C
      COMMON /BLK3/X                                                    C
      BW=FHIGH-FLOW                                                     C
      AS=BW*SRATE*3.1415927                                             C
      AC=SRATE*2.*3.1415927*(FHIGH-BW/2.)                               C
      M=LFILT/2                                                         C
      MID=M+1                                                           C
      DO 10 I=1,M                                                       C
      FILT(MID-I)=COS(AC*I)*SIN(AS*I)/(AS*I)                            C
   10 FILT(MID+I)=FILT(MID-I)                                           C
      FILT(MID)=1.0                                                     C
      BW=FHIGH2-FLOW2                                                   C
      AS=BW*SRATE*3.1415927                                             C
      AC=SRATE*2.*3.1415927*(FHIGH2-BW/2.)                              C
      DO 15 I=1,M                                                       C
      FILT2(MID-I)=COS(AC*I)*SIN(AS*I)/(AS*I)                           C
   15 FILT2(MID+I)=FILT2(MID-I)                                         C
      FILT2(MID)=1.0                                                    C
      NEND=LX-LFILT+1                                                   C
      RETURN                                                            C
      ENTRY BPASS                                                       C
C  Y(IN) X(OUT)                                                         C
      DO 20 I=1,LX                                                      C
   20 X(I,1)=0.                                                         C
      DO 30 I=1,NEND                                                    C
      DO 30 J=1,LFILT                                                   C
   30 X(I,J)=X(I,J)+Y(I+M,1)*FILT(J)                                    C
      RETURN                                                            C
      ENTRY BPASS2                                                      C
C  X(IN) Y(OUT)                                                         C
      DO 40 I=1,LX                                                      C
   40 Y(I,1)=0.                                                         C
      DO 50 I=1,NEND                                                    C
      DO 50 J=1,LFILT                                                   C
   50 Y(I,J)=Y(I,J)+X(I+M,1)*FILT2(J)                                   C
      RETURN                                                            C
      END                                                               C
/*
//LKED.SYSLMOD DD DSN=C740.USGS(USGSC),DISP=(NEW,KEEP),
//   UNIT=2314,VOL=SER=SYS07,SPACE=(TRK,(15,1,1),RLSE)
//LKED.SYSLIB DD DSN=SYS1.FORTLIB,DISP=SHR
//             DD DSN=SYS2.SSPLIB,DISP=SHR
//             DD DSN=SYS2.SUBLIB1,DISP=SHR
```

```
//              DD DSN=SYS2.VERSLIB,DISP=SHR
/*
//LKED.SYSIN DD *
BESD        &    ADCDECT                                                    DC
BTXT        Y    AGOO<G DCDECT&%&<Q(Q(O-& &D&& HG@                          DC
BESD        &    BBLK2     E      |4                                        DC
BTXT        *    A+             A    B    C    (    &                      DC
BTXT    H   H    A""""@                                                     DC
BTXT    Y   D    A                                                          DC
BTXT    0   D    A B                                                        DC
BTXT    8   D    A A                                                        DC
BTXT    0   8    AC-&UQ-&-AO FA& DQO&&<O-   O& QC+   -) &H&&&8Q &8Y-&Q--&&K DC
BTXT   AY   8    AG J&& &M -&&GOJ & & &M,-&&O--DQ ?UI   B& &@8-?4^O HAO -Q DC
BTXT   A-   8    A&8HP- * && && QE+   -) &H&&&8Q/$YY &Q- &&KSG J2&-&M   &&GODC
BTXT   AQ   8    AJ@&B& &M, &&JFQT$Z8S- @-?O:UOT-   # 9RO&@G&J6$"Q-& G=Q &DDC
BTXT   A&   >    AC- <Q& QQQQ%&*K"&<G=Q-& AC- A& B$V&-&U&O&YGO&H            DC
BTXT    -   <    A  AU    Y   A<                                            DC
BTXT    M   *    A    F         D              -    H                       DC
BRLD        -      B A<   H A A<   - A A<   U A A<   Y                       DC
BEND                          A=         DATE  73.027/17.40.22             DC
/*
/*
/*
```

```
 RALPH JOB STATISTICS --     804 CARDS READ --        804 LINES PRINTED --
                           0.00 MINUTES CPU TIME     0.00 MINUTES WAIT TIME
```

```
//LISTIT JOB 'C740,314','LIST PACKAGE CN'                                J
/* SERVICE LIST
//ZAP EXEC FORTHCL,PARM.FORT='OPT=2,NAME=USGSC,LINECNT=60'
//FORT.SYSIN DD *
C  U S G S  S E I S M I C  P R O C E S S I N G  P A C K A G E  CN
C  PROCESSES 1972 ANALOG TO DIGITAL TAPES PRODUCED ON THE CDC 1700
C  MACHINE AT N.C.E.R.  DOUBLE MULTIPLEX 556 BPI TAPES
C        SAMPLE DD CARD FOR TAPE INPUT; ASSUME TAPEDD='FT20F001'
C     //GO.FT2CF001 DD DSN=SEIS,UNIT=001,VOL=SER=UXXXX,DISP=(OLD,KEEP),
C     //  LABEL=(1,BLP,,IN),DCB=(RECFM=U,BLKSIZE=16000,DEN=1,TRTCH=C)
C        REPLACE UXXXX BY COMP. CENTER TAPE I.D., LIKE U1577
C
      INTEGER BUF*2(4000),LVERS(18)
      REAL*4 CX(20,1000),BRHO(10),BSIG(10),PLOC(10)
      REAL*8 DATE,TIME,LINEID,TAPEDD,USER
      COMMON /BLK2/Y(1000),LX,NJUMP,NLEAK,RHO,CRHO,NDOTS
      COMMON /BLK3/X(1000)
      NAMELIST /JOB/NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,RHO,
     1NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,NOTRJ,NJOB,TAPEDD,USER,NPAGES
     2,RLINCH/ADAPT/FHIGH,NZC ,FHIGH2,FLOW2,LENBP,IFLGAP,LCN,DWARM,WSTRT
     3,ZTAU,XTAU/BIFLT/NPTS,PLOC,BSIG,BRHO/JOBIN/NREEL,LINEID,RANGE,WVEL
     4,SRATE,RECEND,NDEC,ITIME,RHO,NLEAK,NJUMP,NBITS,NBITS2,MODADD,NOTR,
     5NOTRJ,NJOB,TAPEDD,USER,NPAGES,RLINCH,FHIGH,NSTR,LENBP,IFLGAP,LCN,
     6DWARM,WSTRT,ZTAU,XTAU,FHIGH2,FLOW2,NPTS,PLOC,BSIG,BRHO,NZC
      DATA NREEL,LINEID,RANGE,WVEL,SRATE,RECEND,NDEC,ITIME,NBITS,NBITS2,
     1NOTR,NOTRJ,NJOB,TAPEDD,FHIGH,NSTR,FLOW2,FHIGH2,LENBP,IFLGAP,LCN,
     2DWARM,WSTRT,ZTAU,XTAU,NPTS,NPAGES,RLINCH,PLOC,BRHO,BSIG,MODADD,
     3USER,NZC/999,'DEFERRED',400.,4875.,.004,1.53,1,540,5,4,2000,2,3,
     4'FT20F001',63.,1,  15.,63.,9,2,9,.2,.2,.12,10.,2,1,7.63,0.,1.536,
     58*0.,.5,.5,8*0.,.99,.99,8*0.0,9,'U.S.G.S.',0/
C
C_____________LIST__OF__PRINCIPAL__VARIABLES_____________________________
C
C  NAME (DEFAULT VALUE)          DEFINITION
C
C       <PARAMETERS USED IN ALL JOBS>
C  USER   (U.S.G.S.)     USER'S NAME (8 ALPHANUMERIC)
C  NREEL  (9999)         USGS REEL NUMBER (INTEGER)
C  LINEID (DEFERRED)     USGS LINE NUMBER (8 ALPAHNUMERIC)
C  RANGE  (400.)         SHOT-RECEIVER OFFSET (FT.)
C  WVEL   (4875.)        WATER VELOCITY (FT/SEC)
C  SRATE  (.004)         INPUT SAMPLE RATE (SEC)
C  RECEND (1.53)         RECORD LENGTH TO PROCESS (SEC)
C  NDEC   (1)            DEGREE OF DECIMATION (INTEGER)    **NOT USED**
C  ITIME  (540)          INTERNAL TIME ESTIMATE (SEC.,INTEGER)
C  RHO    (0.74)         DC REJECT FILTER COEFFICIENT  **NOT USED***
C  NBITS  (5)            NUMBER OF DOTS PER TRACE (INTEGER)
C  NBITS2 (4)            SECONDARY TRACEWIDTH EVERY 'MODADD' TRACES (INT)
C  MODADD (9)            MODULO TO ADD SECONDARY BITS (INTEGER)
C  NOTR   (2000)         NUMBER OF TRACES TO BE PROCESSED (INTEGER)
C  NOTRJ  (2)            PROCESSING FREQUENCY (1 OR 2)
C  NSTR   (1)            NUMBER OF STARTING TRACE ON TAPE (INTEGER)
C  NJOB   (3)            JOB SELECTION SWITCH (1,2,3,OR 4)
C  TAPEDD (FT20F001)     TAPE DATA DEFINITION TAG (8 ALPHANUMERIC)
C  RLINCH (7.68)         RECORD SECTION LENGTH IN INCHES (MAX.=15.25)
C       <ADDITIONAL PARAMETERS USED IN JOBS 1,2,3>
C  NLEAK  (10)           INTEGRATION PARAMETER FOR AGC (INTEGER)
C  NJUMP  (24)           UPDATE FREQUENCY OF AGC GAIN TRACE (INTEGER)
C  LENBP  (9)            NUMBER OF COEFFICIENTS IN BANDPASS (INTEGER,ODD)
C  FHIGH  (63.)          PRIOR HIGH CUTOFF (CYCLES/SEC)
```

```
C   FLOW2   (15.)          POST-PROC. LOW CUTOFF (CYCLES/SEC)
C   FHIGH2 (63.)          POST-PROC. HIGH CUTOFF (CYCLES/SEC)
C        <ADDITIONAL PARAMETERS USED IN JOBS 1,3>
C   NPTS    (2)           NO. POINTS ON BIFILT PROFILE (INTEGER)
C   PLCC    (0.,1.53)     LOCATION OF PROFILE TIES (SEC)
C   PRHO    (.5,.5)       RHO PROFILE COEFFICIENTS
C   PSIG    (.99,.99)     SIGMA PROFILE COEFFICIENTS
C        <ADDITIONAL PARAMETERS USED IN JOBS 2,3>
C   NZC     (0)           NO. OF LEADING ZERO REFLECTION COEFF.(INT)
C   IFLGAP  (2)           GAP BETWEEN ADAPTIVE FILTER COEFFICIENTS (INT)
C   LCN     (9)           NO. OF REFLECTION COEFF. IN LADDER (INTEGER)
C   DWARM   (.2)          DURATION OF STATIONARY WARM-UP CYCLE (SEC)
C   WSTRT   (.2)          STARTING POSITION OF WARM-UP (SEC)
C   ZTAU    (.12)         RELAXATION TIME TO 1/E (SEC)
C   XTAU    (10.)         RELAXATION DISTANCE TO 1/E (TRACES)
C
C              (ALL VARIABLES ARE REAL EXCEPT AS INDICATED)
C
C-------------------------------------------------------------------------
C
C            PROCESSING SEQUENCES:
C
C        NJOB=1    /FIELD/    /DCDEC/    /BPASS/BIFILT/BPASS2/SEISGM/
C        NJOB=2    /FIELD/    /DCDEC/    /BPASS/BAFL/BPASS2/SEISGM/
C        NJOB=3    /FIELD/    /DCDEC/    /BPASS/BAFL/BIFILT/BPASS2/SEISGM/
C        NJOB=4    /FIELD/    /DCDEC/SEISGM/
C-------------------------------------------------------------------------
C            VERSION G                          DON C. RILEY DECEMBER 1972
C
      CALL MCLOCK(DATE,TIME,WEEKDA)
      CALL PCLOCK(IJS)
      RHO=0.74
      NLEAK=10
      NJUMP=24
      WRITE(6,876)
      READ(5,JOBIN,ERR=977)
    2 LENGTH=RECEND/SRATE
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      IF(NSTR.LE.1) GO TO 4
      M=NSTR-1
      DO 3 I=1,M
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
    3 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
    4 LX=LENGTH/NDEC
      NOTRJ=MIN0(NOTRJ,2)
      RLINCH=AMIN1(RLINCH,15.3)
      IF(RLINCH.GT.7.65) NPAGES=2
      SRATE2=SRATE*NDEC
C     CRHO=(1.+RHO)/2.          ** NOT USED **
      KTEST=24*NOTRJ
      WRITE(6,JCB)
      IF(NJOB.EQ.4)  GO TO 40
      FLOW=-FHIGH
      IF(MOD(LENBP,2).EQ.0)  LENBP=LENBP+1
      CALL GFILT(FLOW,FHIGH,SRATE2,LENBP,FLOW2,FHIGH2,LX)
      IF(NJOB.EQ.1) GO TO 30
      WRITE(6,ADAPT)
      NSTRT=WSTRT/SRATE2
      NWARM=DWARM/SRATE2
      ZTAT=ZTAU/SRATE2
```

```
      IF(NJOB.EQ.2) GO TO 40
   30 WRITE(6,BIFLT)
      CALL BIFILT(LX,BSIG,BRHO,PLCC,NPTS,SRATE2)
   40 CALL TAXIS(LX,SRATE2,NPAGES,RLINCH)
CON   CALL NMCSET(RANGE,WVEL,SRATE,LENGTH)
      ITIME=ITIME*100
      CALL FIO999(44,LVERS(1),18,18)
C
      GO TO (1727,1707,1747,1737),NJOB
C
C                  NJOB=2
C
 1707 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL NCERRD(BUF(751),LEN,IERR,TAPEDC)
      CALL DCDECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      CALL BAFL(LX,CX,IFLGAP,LCN,NWARM,NSTRT,ZTAT,XTAU,NZC)
      CALL BPASS2
      NDOTS=NBITS
      CALL SEISGM
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      DO 707 I=NOTRJ,NOTR,NOTRJ
      NDOTS=NBITS
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NCERRD(BUF(751),LEN,IERR,TAPEDC)
      IF(IERR.EQ.1) GO TO 988
      CALL DCDECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL BAFLGO
      IF(NOTRJ.GE.2) CALL NCERRD(  BUF,LEN,IERR,TAPEDC)
      CALL BPASS2
      CALL NCERRD(BUF,LEN,IERR,TAPEDD)
      CALL SEISGM
      IF(MOD(I,KTEST).NE.0) GO TO 707
      CALL PCLOCK(IJE,IJS)
      IF(IJE.GT.ITIME) GO TO 909
  707 CONTINUE
      GO TO 999
C
C
C                  NJOB=1
C
 1727 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      DO 727 I=NOTRJ,NOTR,NOTRJ
      NDOTS=NBITS
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NCERRD(BUF(751),LEN,IERR,TAPEDC)
      IF(IERR.EQ.1) GO TO 988
      CALL DCDECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      CALL BIGOT
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL BPASS2
```

```
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL SEISGM
      IF(MOD(I,KTEST).NE.0) GO TO 727
      CALL PCLOCK(IJE,IJS)
      IF(IJE.GT.ITIME) GO TO 909
  727 CONTINUE
      GO TO 999
C
C                   NJOB=4
C
 1737 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      DO 737 I=NOTRJ,NOTR,NOTRJ
      NDOTS=NBITS
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL DODECN(BUF,LX,Y)
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL SEISGM
      IF(MOD(I,KTEST).NE.0) GO TO 737
      CALL PCLOCK(IJE,IJS)
      IF(IJE.GT.ITIME) GO TO 909
  737 CONTINUE
      GO TO 999
C
C                   NJOB=3
C
 1747 CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      CALL DODECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      CALL BAFL(LX,CX,IFLGAP,LGN,NWARM,NSTRT,ZTAT,XTAU,NZC)
      CALL BIGOT
      CALL BPASS2
      NDOTS=NBITS
      CALL SEISGM
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      DO 747 I=NOTRJ,NOTR,NOTRJ
      NDOTS=NBITS
      IF(MOD(I,MODADD).EQ.0)  NDOTS=NBITS2
      CALL NOERRD(BUF(751),LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL DODECN(BUF,LX,Y)
COUT  CALL AGC
      CALL BPASS
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      IF(IERR.EQ.1) GO TO 988
      CALL BAFLGO
      CALL BIGOT
      IF(NOTRJ.GE.2) CALL NOERRD(  BUF,LEN,IERR,TAPEDD)
      CALL BPASS2
      CALL NOERRD(BUF,LEN,IERR,TAPEDD)
      CALL SEISGM
      IF(MOD(I,KTEST).NE.0) GO TO 747
      CALL PCLOCK(IJE,IJS)
      IF(IJE.GT.ITIME) GO TO 909
```

```
  747 CONTINUE
C
C
C
  999 CALL PCLOCK(IJE,IJS)
      WRITE(6,900) IJE
      CALL XCLOCK(IJS)
      WRITE(44,622) USER
      DO 5 K=1,19
    5 CALL VLINE(LVERS,70)
      CALL VLINE(' ',1)
      CALL VLINE(' ',1)
C
C     WRITE PARM.JOB TO VERSATEC.....
      WRITE(44,600)
      CALL VLINE(LVERS,70)
      WRITE(44,601) NREEL
      CALL VLINE(LVERS,70)
      WRITE(44,602) LINEID
      CALL VLINE(LVERS,70)
      WRITE(44,603)
      GO TO (50,60,70,45),NJOB
   45 WRITE(44,621)
      CALL VLINE(LVERS,70)
      GO TO 80
   50 WRITE(44,614)
      CALL VLINE(LVERS,70)
      WRITE(44,615)
      CALL VLINE(LVERS,70)
      GO TO 80
   60 WRITE(44,604)
      CALL VLINE(LVERS,70)
      WRITE(44,605)
      CALL VLINE(LVERS,70)
      GO TO 80
   70 WRITE(44,616)
      CALL VLINE(LVERS,70)
      WRITE(44,617)
      CALL VLINE(LVERS,70)
   80 WRITE(44,603)
      CALL VLINE(LVERS,70)
      WRITE(44,606) RANGE,WVEL,SRATE,NSTR
      CALL VLINE(LVERS,70)
      WRITE(44,609) RLINCH,NLEAK,NJUMP,NBITS,MODADD
      CALL VLINE(LVERS,70)
      IF(NJOB.EQ.4)  GO TO 120
      IF(NJOB.EQ.1) GO TO 110
      WRITE(44,607) FLOW2,FHIGH2,LENBP,IFLGAP,LCN,NZC
      CALL VLINE(LVERS,70)
      WRITE(44,608) DWARM,WSTRT,ZTAU,XTAU,RHO
      CALL VLINE(LVERS,70)
      IF(NJOB.EQ.2) GO TO 120
  110 WRITE(44,618) NPTS,PLCC
      CALL VLINE(LVERS,70)
      WRITE(44,619) BSIG
      CALL VLINE(LVERS,70)
      WRITE(44,620) BRHO
      CALL VLINE(LVERS,70)
  120 WRITE(44,610) NCTR,NOTRJ,RECENC,NDEC,NPAGES
      CALL VLINE(LVERS,70)
```

```
      WRITE(44,603)
      CALL VLINE(LVERS,70)
      WRITE(44,612)  IJE,DATE,TIME
      CALL VLINE(LVERS,70)
      WRITE(44,600)
      CALL VLINE(LVERS,70)
C
C
      CALL VLINE(' ',1)
      CALL VLINE(' ',1)
      CALL VLINE('                    T W O - W A Y  T R A V E L  T I M E',
     152)
      CALL VLINE(' ',1)
      CALL VLINE(' ',1)
      CALL XCLOCK(IJE,IJS)
      WRITE(6,879) IJE
      CALL XCLOCK(IJS)
      CALL SDUMP
      CALL XCLOCK(IJE,IJS)
      WRITE(44,613) IJE
      WRITE(6,613)  IJE
      CALL VLINE(LVERS,70)
      CALL VLINE('                                END OF PLOT        BIN 314',49)
      READ(5,JOBIN,ERR=977,END=9999)
      WRITE(6,877)
      CALL MCLOCK(DATE,TIME,WEEKDA)
      CALL PCLOCK(IJS)
      CALL ENDTAP
      REWIND 1
      GO TO 2
 9999 STOP
  988 WRITE(6,880) I
      GO TO 999
  909 WRITE(6,878) I
      WRITE(44,878) I
      CALL VLINE(LVERS,70)
      GO TO 999
  977 WRITE(6,881)
      GO TO 9999
C
C         F O R M A T S
  600 FORMAT(70('*'))
  601 FORMAT('*',8X,'CHUKCHI SEA 1972 SEISMIC DATA  U.S.G.S. REEL NO.',
     1 I4,8X,'*')
  602 FORMAT('*',10X,'SINGLE-FOLD MARINE REFLECTION PROFILE NO.',A8,9X,
     1 '*')
  603 FORMAT('*',30X,'<<<<>>>>',30X,'*')
  604 FORMAT('*',15X,'-TIME & SPACE ADAPTIVE DECONVOLUTION-',16X,'*')
  605 FORMAT('*',6X,'PROCESSING SEQUENCE: /FIELD/NMO/DCDEC/AGC/BAFLGO/SE
     1ISGM/',6X,'*')
  606 FORMAT('*   PARM.JOB=',1H','RANGE=',F5.0,',WVFL=',F6.0,',SRATE=',
     1 F6.3,',NSTR=',I4,',          *')
  607 FORMAT('*',13X,'FLOW2=',F3.0,',FHIGH2=',F3.0,',LENBP=',I2,',IFLGAP
     1=',I1,',LCN=',I2,',NZC=',I1,',    *')
  608 FORMAT('*',13X,'DWARM=',F3.2,',WSTRT=',F3.2,',ZTAU=',F5.3,',XTAU='
     1,F5.1,',RHO=',F3.2,',',5X,'*')
  609 FORMAT('*',13X,'RLINCH=',F4.1,',NLEAK=',I3,',NJUMP=',I4,',NBITS=',
     1 I2,',MODADD=',I2,',    *')
  610 FORMAT('*',13X,'NOTR=',I4,',NOTRJ=',I1,',RECEND=',F4.2,',NDEC=',
     1I1,',NPAGES=',I1,1H',9X,'*')
```

```
  612 FORMAT ('*  360/67 CPU TIME= ',I5,' CSEC.  DATE: ',A8,2X,
     1 A8,'  STANFORD  *')
  613 FORMAT('   DRUM.DUMP.PLAYBACK.TIME = ',I7,' CSEC.')
  614 FORMAT('*',23X,'-HORIZONTAL BIFILTER-',24X,'*')
  615 FORMAT ('*',6X,'PROCESSING SEQUENCE: /FIELD/NMO/DCDEC/AGC/BIFILT/S
     1EISGM/',6X,'*')
  616 FORMAT('*',10X,'-TIME & SPACE ADAPTIVE DECONVOLUTION + BIFILTER-',
     110X,'*')
  617 FORMAT('*',5X,'PROCESSING SEQ: /FIELD/NMO/DCDEC/AGC/BAFLGO/BIFILT/
     1SEISGM/',5X,'*')
  618 FORMAT('*',13X,'NPTS=',I2,',LOC=',10(F3.1,','),T70,'*')
  619 FORMAT('*',20X,'BSIG=',10(F3.2,','),T70,'*')
  620 FORMAT('*',20X,'BRHO=',10(F3.2,','),T70,'*')
  621 FORMAT('*',9X,'SEISMIC PLAYBACK ONLY SEQ: /FIELD/NMO/DCDEC/SEISGM/
     1',8X,'*')
  622 FORMAT('START OF PLOTJOB ',A8,', BIN 314')
  876 FORMAT(1HC,30X,10('*'),' BEGIN CHUKCHI JOB A ',10('*'))
  877 FORMAT(1H1,30X,10('*'),' BEGIN CHUKCHI JOB B ',10('*'))
  878 FORMAT(1HO,'EST. TIMER SELF-EXIT AT TR. ',I8)
  879 FORMAT(1H ,'W-BLOCK TIMER:',I8)
  880 FORMAT(1H ,'***END OF FILE ENCOUNTERED AT TR. NO.',I6)
  881 FORMAT(1HO,'***ERROR IN INPUT CARDS *** ABNEOJ ***')
  900 FORMAT(1HO,7X,'PARTITION TIMER IN CSEC. ',I8)
C
      END
      SUBROUTINE BAFL(LOUT,CX,IFLGAP,LCN,NWARM,ISTRT,ZTAU,XTAU,NZERO)
C
C                  ----------------------------
C--------------S U B R O U T I N E   B A F L---------------
C                  ----------------------------
C
C        THE BURG ADAPTIVE FILTER LADDER: AN ADAPTIVE OR TIME-VARYING
C        FIXED-LEAD PREDICTION ERROR PROCESSOR.  ADJUSTMENT OF EACH
C        REFLECTION COEFFICIENT IS MADE EVERY JUMP STATE ATTEMPTING
C        TO MINIMIZE THE STAGE OUTPUT POWER.
C        INPUTS:
C          X(1)...X(LX)=INITIAL DATA
C          LCN= LAST NON-ZERO REFLECTION COEFFICIENT
C          IFLGAP=  NUMBER OF GAPS BETWEEN FILTER COEFFICIENTS
C              SETTING IFLGAP=0 DOES NOT GAP THE F.C. AND
C              TRIES TO OPERATE ON THE ENTIRE SPECTRUM FROM
C              0 TO W WHERE W IS THE FOLDING FREQUENCY.
C              SETTING IFLGAP=1 OPERATES ON THE PORTION OF
C              THE SPECTRUM FROM 0 TO W/2 , IFLGAP=2 FROM
C              0 TO W/3 ETC. THUS ALLOWING SPECIFICATION OF
C              WHAT PART OF THE SPECTRUM TO DECONVOLVE.
C
C          NWARM=DURATION OF STATIONARY C ESTIMATION CYCLE
C          ISTRT=START OF STATIONARY GATE
C          ZTAU=TEMPORAL RELAXATION TIME TO 1/E
C          XTAU=SPATIAL RELAXATION DISTANCE TO 1/E
C        OUTPUTS:
C          X(1)...X(LOUT)=FORWARD ERROR PREDICTION TRACE
C        OTHER VARIABLES:
C          F(1)...F(LCN)=FORWARD STATE VECTOR
C          B(1)...B(LCN)=BACKWARD STATE VECTOR
C          C(1)...C(LCN)=REFLECTION COEFFICIENTS AT EACH STATE
C          CX=REFLECTION COEFF. INTEGRATED IN SPACE & TIME
C          DEN(1)...DEN(LCN)=  STAGE AUTOPOWER
C          NUM(1)...NUM(LCN)= STAGE CROSSPOWER
```

```
C
C           CALLING 'BAFL' FIRST SETS UP THE LOOPING AND PASSING
C           ARRAYS FOR THE PARTICULAR PROBLEM AS SPECIFIED BY LCN &
C           IFLGAP.  THEN IT COMPUTES A SHORT (LENGTH=NWARM) ESTIMATE
C           OF THE REFLECTION COEFFICIENT SERIES IN ORDER TO START THE
C           ADAPTION OUT WITH SOME REASONABLE NUMBERS. THEN IT LOADS UP
C           THE CX ARRAY WITH THE INITIAL VALUES AND PASSED INTO ENTRY
C           'BAFLGO'.
C
C           THE USUAL ENTRY IS 'BAFLGO' WHICH FIRST INITIALIZES THE
C           BACKWARD ARRAY THEN PASSES TO THE MAIN ALGORITHM.
C           THE CX SERIES IS UPDATED EVERY IFLGAP DATA POINTS AND IN
C           THE INTERMEDIATE STEPS THE OUTPUT ARE INTERPOLATED OR
C           PROCESSED AS THOUGH THE R.C. WERE STATIONARY.
C
C
C
C
      REAL NUM(50),DEN(50),B(50),F(50),C(50),EM(1000),EP(1000),
     1 CX(LCN,LOUT),A(50)
      COMMON /BLK3/X(1000)
      DATA A,B,C/150*0.0/
      FTEST=1.00
      LCNP1=LCN+1
      LCNP2=LCN+2
      IFGM1=IFLGAP-1
      LBSP=LOUT-IFLGAP
      NZERP1=NZERO+1
      NEND=LCN-NZERP1
      DO 80 K=LBSP,LOUT
   80 EP(K)=0.
C BEGIN STATIONARY WARM-UP
      DO 10 I=1,NWARM
      EM(I)=X(I*IFLGAP+ISTRT)
   10 EP(I)=X(I*IFLGAP+ISTRT)
C
      DO 11 J=2,LCNP1
      DEN(J)=0.
      NUM(J)=0.
      DO 12 I=J,NWARM
      DEN(J)=DEN(J)+EP(I)*EP(I)+EM(I-J+1)*EM(I-J+1)
   12 NUM(J)=NUM(J)+EP(I)*EM(I-J+1)
      DIV=NWARM-J+1
      NUM(J)=NUM(J)/DIV
      DEN(J)=DEN(J)/DIV
      C(J)=-2.*NUM(J)/DEN(J)
      DO 11 I=J,NWARM
      EPI=EP(I)
      EP(I)=EPI+C(J)*EM(I-J+1)
      KS=IFLGAP*LCN+1
   11 EM(I-J+1)=EM(I-J+1)+C(J)*EPI
      DO 8 J=  1,LCN
      DO 8 K=1,KS
    8 CX(J,K)=C(J+1)
      DO 33 J=1,LCN
      DEN(J)=DEN(J+1)
   33 NUM(J)=NUM(J+1)
C END WARM-UP CYCLE
C SET RELAXATION TIMES
      DLX=EXP(-1./XTAU)
```

```
      DL=EXP(-1./ZTAU)
      DRX=1.-DL*X
      DR=1.-DL
C
C---------------USUAL--ENTRY-----------------------------------------
      ENTRY BAFLGO
C
C           INITIALIZE BACKWARD VECTOR
      B(1)=X(KS-IFLGAP)
      A(1)=1.0
      DO 1000 J=2,LCN
      B(J)=X(KS-J*IFLGAP)
      A(J)=CX(J,KS)
      DO 1000 I=2,J
      A(I)=A(I)+CX(J,KS)*A(J-I+1)
 1000 B(J)=B(J)+A(I)*X(KS+(I-J-1)*IFLGAP)
C
C--------BEGIN--MAIN--LOOP-------------------------------------------
C
      DO 5000 K=KS,LOUT,IFLGAP
      Z=X(K)
      DO 1010 J=1,NZERP1
 1010 F(J)=Z
      DO 2000 J=NZERP1,LCN
      DEN(J)=(F(J)**2+B(J)**2)*DR+DEN(J)*DL
      NUM(J)=F(J)*B(J)*DR+NUM(J)*DL
      IF(FTEST.LE.1.1) CX(J,K)=-2.*NUM(J)/DEN(J)
      CX(J,K)=-2.*DRX*NUM(J)/DEN(J)+CX(J,K)*DLX
 2000 F(J+1)=F(J)+CX(J,K)*B(J)
      X(K)=F(LCNP1)
      DO 3000 JR=1,NEND
      J=LCN-JR
 3000 B(J+1)=B(J)+CX(J,K)*F(J)
      IF(NZERO.EQ.0) GO TO 5000
      DO 4000 JR=1,NZERO
      J=NZERP1-JR
 4000 B(J+1)=B(J)
 5000 B(1)=Z
C
C--------END--OF--MAIN--LOOP-----------------------------------------
C
C       NOW GO BACK AND FILL IN THE GAPS....
      IF(IFLGAP.EQ.1) GO TO 9050
      DO 9000 L=1,IFGM1
      KSTART=KS+L
      DO 9000 K=KSTART,LOUT,IFLGAP
      KC=K-L
      Z=X(K)
      DO 6000 J=1,NZERP1
 6000 F(J)=Z
      DO 7000 J=NZERP1,LCN
 7000 F(J+1)=F(J)+CX(J,KC)*B(J)
      X(K)=F(LCNP1)
      DO 8000 JR=1,NEND
      J=LCN-JR
 8000 B(J+1)=B(J)+CX(J,KC)*F(J)
      IF(NZERO.EQ.0) GO TO 9000
      DO 8050 JR=1,NZERO
      J=NZERP1-JR
 8050 B(J+1)=B(J)
```

```
 9000 B(1)=Z                                                            C
C                                                                       C
 9050 FTEST=FTEST+1.0                                                   C
      RETURN                                                            C
      END                                                               C
      SUBROUTINE SEISGM                                                 C
C                                                                       C
C        SUBROUTINE 'SEISGM'   VARIABLE AREA SEISMIC SECTION PLOTTER.   C
C        THROUGH MULTIPLE CALLS TO SEISGM ADJACENT TRACES ARE WRITTEN   C
C        TO THE VERSATEC FILLING THE FULL-PAGE WIDTH (560 BITS).        C
C        DATA: INPUT DATA SERIES                                        C
C        NT = LENGTH OF INPUT DATA                                      C
C        NBITS= WIDTH OF VARIABLE AREA TRACE IN VERSATEC BITS           C
C        NPAGES: WHEN NPAGES=1 THE RECORD SECTION FILLS ONE PAGE        C
C                OF THE VERSATEC (7.536 IN.) FOR ANY LENGTH RECORDS.    C
C                WHEN NPAGES=2 A SPECIAL OPTION IS INVOKED ALLOWING     C
C                SOMEWHAT MORE GENERAL DISPLAY AS FOLLOWS.....          C
C                'RLINCH' INDICATES THE DESIRED RECORD SECTION LENGTH   C
C                IN INCHES. THUS THE RECORDS OF LENGTH NT ARE SCALED    C
C                INTO A VARIABLE OBJECT SPACE THAT MAY TAKE UP MORE     C
C                THAN ONE PAGE-WIDTH.  THIS IS ACCOMPLISHED BY COMPOSING C
C                THE PLOT INTO A LONGER BUFFER THAN THE VERSATEC CAN    C
C                HANDLE(WHICH IS 70 BYTES) AND DUMPING THIS PLOTTED     C
C                DATA ONTO A 2301 DRUM (FT UNIT 1) .  THEN WE REWIND AND C
C                FETCH AND WRITE THE FIRST 70 BYTES ON ONE PAGE; THEN   C
C                GO BACK AND FETCH AND WRITE THE NEXT 70 BYTES OR       C
C                LESS.                                                  C
C                                                                       C
C     'TAXIS' MUST BE CALLED BEFORE 1ST 'SEISGM' CALL SINCE             C
C     A REFERENCE TABLE MUST BE GENERATED.                              C
C                                                                       C
      INTEGER*4 IREF(1120)                                              C
      LOGICAL*4 LMZ,LZ,LASK,LINE(35,10),LOAD(18),LOAD2(35),LBOT*1(72)   C
      LOGICAL*4 LTAX(35),LIGHT(35),DARK(35),LMASK(32)                   C
      COMMON /BLK2/DATA(1000),NT,N5,N6,R5,R6,NBITS                      C
      EQUIVALENCE (LBOT(1),LOAD2(18))                                   C
      DATA LMZ,LZ,LIGHT,DARK/Z80000000,36*Z00000000,35*ZFFFFFFFF/       C
      DATA LMASK/Z80000000,Z40000000,Z20000000,Z10000000,Z08000000,     C
     1Z04000000,Z02000000,Z01000000,Z00800000,Z00400000,Z00200000,      C
     2Z00100000,Z00080000,Z00040000,Z00020000,Z00010000,Z00008000,      C
     3Z00004000,Z00002000,Z00001000,Z00000800,Z00000400,Z00000200,      C
     4Z00000100,Z00000080,Z00000040,Z00000020,Z00000010,Z00000008,      C
     5Z00000004,Z00000002,Z00000001/                                    C
      NMAG=MIN0(10,NBITS)                                               C
      SUM=0.                                                            C
      DO 121 K=1,NT,4                                                   C
      T=DATA(K)                                                         C
  121 SUM=SUM+ABS(T)                                                    C
      ERMS=4.*SUM/NT                                                    C
C     WRITE(6,900) ERMS                                                 C
C 900 FORMAT(1H+,100X,F11.1)                                            C
      SCALE=0.5*NMAG/ERMS                                               C
      IPIAS=(NMAG+1)/2                                                  C
      DO 20 I=1,JBYTES                                                  C
      DO 20 J=1,NMAG                                                    C
C   SETTING = LTAX WRITES TIME MARKS ACROSS ENTIRE SECTION,LZ DOESN'T   C
C  20 LINE(I,J) = LZ                                                    C
   20 LINE(I,J)=LTAX(I)                                                 C
      DO 40 IBIT=1,32                                                   C
      LASK=LMASK(IBIT)                                                  C
```

```
      IBYTE = 1
      DO 40 IL=IBIT,JBITS,32
      IMAG=IBIAS+DATA(IREF(IL))*SCALE
      IF(IMAG.LE.0) GO TO 40
      IMAG=MIN0(NMAG, IMAG)
      DO 30 I=1,IMAG
   30 LINE(IBYTE,I) = LINE(IBYTE,I).OR.LASK
   40 IBYTE = IBYTE + 1
      DO 50 I=1,NMAG
   50 WRITE(1)  (LINE(K,I),K=1,JBYTES)
      LKNT = LKNT + NMAG
      RETURN
C
C        ENTRY 'TAXIS' WRITES A TIME SCALE TO THE VERSATEC.
C        SRATE IS THE SAMPLING RATE.  TICK MARKS ARE AT 100 MILS.
C
      ENTRY TAXIS(NT,SRATE,NPAGES,RLINCH)
      JBYTES=18
      JBITS=560
      IF(NPAGES.LE.1) GO TO 55
      JBYTES=35
      JBITS=MIN1(1120.,RLINCH*73.25)
   55 SCALE=(JBITS-1)/(NT*SRATE)
      ITMAX=SRATE*NT*10. + 1.
      TYME=0.
      DO 60 I=1,JBYTES
   60 LTAX(I)=LZ
      DO 70 I=1,ITMAX
      IR=SCALE*TYME + 1.0
      IBYTE=(IR-1)/32  + 1
      IBIT=IR - (IBYTE-1)*32
      LASK=LMASK(IBIT)
      LTAX(IBYTE)=LTAX(IBYTE).OR.LASK
   70 TYME=TYME + 0.1
      SCALE=(NT-1.)/(JBITS-1.)
      DO 90 IL=1,JBITS
   90 IREF(IL)=SCALE*(IL-1) + 1.49999
      DO 83 I=1,8
   83 WRITE(1) LIGHT
      WRITE(1) DARK
      DO 84 I=1,10
   84 WRITE(1) LTAX
      WRITE(1) DARK
      DO 81 I=1,10
   81 WRITE(1) LIGHT
      LKNT=30
C
      RETURN
C
C        ENTRY SDUMP DUMPS THE DRUM TO THE VERSATEC
C
      ENTRY SDUMP
      DO 85 I=1,380
   85 WRITE(1) LIGHT
      REWIND 1
      NL=LKNT + 878
      NFIN=NL
      IF(NPAGES.GT.1) NFIN=NL-340
      DO 82 I=1,NFIN
      READ(1) LOAD
```

```
   82 CALL WRITER(LOAD,70)
      IF(NPAGES.LE.1) GO TO 87
      REWIND 1
      DO 86 I=1,NL
      READ(1) LOAD2
   86 CALL WRITER(LBOT(3),70)
   87 CONTINUE
      RETURN
      END
      SUBROUTINE BIFILT(ND,SIG,RHO,PLOC,NPTS,SRATE2)
C                 BILINER KX FILTER
C                 USEFUL FOR EXTINGUISHING HORIZONTAL PRIMARIES
C                 AND MULTIPLES INTERFERING WITH DIPPING REFLECTORS
C                 AND/OR FOR STACKING IN  X  TO ENHANCE LATERAL
C                 CORRELATION IN THE PRESENCE OF ADDITIVE NOISE.
C                 ALGORITHM OPERATES IN TIME VARYING MODE BETWEEN
C                 PURE  KX  LOWCUT (RHO=0) AND PURE  KX  HIGHCUT
C                 (SIG=1) AS SET IN THE RHO/SIG PROFILES.
C
C            PARAMETER LIST:
C            IN      INPUT TRACE
C            OUT     OUTPUT TRACE
C            ND      LENGTH OF INPUT/OUTPUT TRACE
C            SIG     SIGMA PARAMETER PROFILE
C            RHO     RHO PARAMETER PROFILE
C            LOC     LOCATION POINTS OF PARAMETER CHANGES
C            NPTS    NO. OF SIG/RHO/LOC POINTS
C
C                 RHO                    SIGMA
C     LOC1----------X-------   --X--------------
C          |          . RHO1       |   .
C          |        .              |     .
C     LOC2 |       XRHO2          |     XSIG2
C          |        .              |        .
C     LOC3 |       XRHO3          |       XSIG3
C          |      .                |         .
C          |    .                  |           .
C     LOC4 |   XRHO4               |           XSIG4
C          |                       |
C
      REAL*4 SIG(1),RHO(1),PLOC(1),D(1000),E(1000),F(1000)
      REAL*4 IN,OUTH(1000),OUTHH(1000),INH(1000)
      INTEGER*4 LOC(10)
      COMMON /BLK3/IN(1000)
      LOC(1)=1
      DO 5 K=2,NPTS
      LOC(K)=PLOC(K)/SRATE2
      IF(LOC(K-1).GT.LOC(K)) GO TO 97
    5 CONTINUE
      LOC(NPTS)=ND
      NPM1=NPTS-1
      WRITE(6,800) RHO(1),SIG(1)
  800 FORMAT(1HO,6X,'PARAMETER PROFILE'//1H ,3X,'DEPTH',
     1 4X,'RHO',5X,'SIGMA'//1H ,6X,'1',2(3X,F6.3))
      DO 10 IW=1,NPM1
      WRITE(6,805) LOC(IW+1),RHO(IW+1),SIG(IW+1)
  805 FORMAT(1H ,3X,I4,2(3X,F6.3))
      FMR=(RHO(IW+1)-RHO(IW))/(LOC(IW+1)-LOC(IW))
      FMS=(SIG(IW+1)-SIG(IW))/(LOC(IW+1)-LOC(IW))
      NEND=LOC(IW+1) - 1
```

```
      NSTRT=LOC(IW)
      IF(NSTRT.GT.NEND) GO TO 97
      DO 10 K=NSTRT,NEND
      R=RHO(IW) + FMR*(K-NSTRT)
      S=SIG(IW) + FMS*(K-NSTRT)
      D(K)=0.5*(1.+S)*(1.-R)
      E(K)=S+R
   10 F(K)=S*R
      E(ND)=SIG(NPTS)+RHO(NPTS)
      F(ND)=SIG(NPTS)*RHO(NPTS)
      D(ND)=0.5*(1.+SIG(NPTS))*(1.-RHO(NPTS))
      DO 20 K=1,ND
      OUTH(K)= 0.0
      INH(K)= 0.0
   20 OUTHH(K)=0.0
      RETURN
C
      ENTRY BIGOT
      DO 70 K=1,ND
      TEMP=D(K)*(IN(K)-INH(K))+E(K)*OUTH(K)-F(K)*OUTHH(K)
      OUTHH(K)=OUTH(K)
      OUTH(K)=TEMP
      INH(K)=IN(K)
   70 IN(K)=TEMP
      RETURN
   97 WRITE(6,900)
  900 FORMAT(1HO,'***PARM.RHO/SIG ERROR***ABNEOJ***')
      STOP
      END
      SUBROUTINE GFILT(FLOW,FHIGH,SRATE,LFILT,FLOW2,FHIGH2,LX)
      REAL*4 FILT(20),FILT2(20),X(1,1000),Y(1,1000)
      COMMON /BLK2/Y
      COMMON /BLK3/X
      BW=FHIGH-FLOW
      AS=BW*SRATE*3.1415927
      AC=SRATE*2.*3.1415927*(FHIGH-BW/2.)
      M=LFILT/2
      MID=M+1
      DO 10 I=1,M
      FILT(MID-I)=COS(AC*I)*SIN(AS*I)/(AS*I)
   10 FILT(MID+I)=FILT(MID-I)
      FILT(MID)=1.0
      BW=FHIGH2-FLOW2
      AS=BW*SRATE*3.1415927
      AC=SRATE*2.*3.1415927*(FHIGH2-BW/2.)
      DO 15 I=1,M
      FILT2(MID-I)=COS(AC*I)*SIN(AS*I)/(AS*I)
   15 FILT2(MID+I)=FILT2(MID-I)
      FILT2(MID)=1.0
      NEND=LX-LFILT+1
      RETURN
      ENTRY BPASS
C  Y(IN) X(OUT)
      DO 20 I=1,LX
   20 X(I,1)=0.
      DO 30 I=1,NEND
      DO 30 J=1,LFILT
   30 X(I,J)=X(I,J)+Y(I+M,1)*FILT(J)
      RETURN
      ENTRY BPASS2
```