A PROCESSING SYSTEM FOR MULTIPATH ESTIMATION  
LOCALIZATION AND TRACKING  

FINAL REPORT – Contract No. N00014-87-C-0147  
20 January 1989

Prepared for:  
J. G. Smith  
Office of Naval Research  
800 N. Quincy Street  
Arlington, VA 2217-5000  
and  
Tom Ballard  
Naval Weapons Surface Center  
White Oaks, Silver Spring  
Maryland 20910

Prepared by:  
Benjamin Friedlander  
Signal Processing Technology  
703 Coastland Drive  
Palo Alto, CA 94303
ABSTRACT

This report describes the development of a processing system for localization and tracking of underwater sources. The system provides an interactive and user-friendly environment for processing sonobuoy data to generate correlograms, extract the time history of the differential delays or time difference of arrivals (TDOA's) between pairs of sensors and perform various operations on the data. This system is also equipped with versatile graphics to allow the user to display the results of each processing step both on a Tektronix 4014 terminal or on a hardcopy device such as a laser printer.
# TABLE OF CONTENTS

1.0 Introduction ........................................... 1

2.0 Algorithms ............................................. 3
   2.1 Correlogram Generation ............................... 4
   2.2 Feature Extraction ................................... 8
   2.3 Sequential Methods .................................. 8
   2.4 Block Method ........................................ 11
   2.5 Display Functions .................................... 14

3.0 Software Description .................................... 22
   3.1 FORTRAN Library of Signal Processing and Utility Functions ................................. 22

4.0 Conclusions ............................................. 27

References .................................................. 29

Appendix A: List of Subroutines in the Multipath Processing Software ................................. 31

Appendix B: User’s Guide .................................... 38
1.0 Introduction

This report provides a summary of the work performed under contract N00014-87-C-0147 for the Office of Naval Research. This report is responsive to the statement of work in the project plan. The goal of the project was to develop a capability for interactive processing of sonoboy data utilizing multipath information. Figure 1 shows the overall data processing system developed in this project. The system consists of five subsystems:

1. Correlogram generator
2. Line set selector
3. Line tracker
4. Track parameter estimator
5. Display functions

The correlogram generator computes the auto-correlation of data from one sensor or the cross-correlation of data from two sensors. Figure 2 shows an example of a correlogram. Section 2.1 discusses this in more detail. The line tracking algorithm uses a maximum a-posteriori probability (MAP) criterion to detect and track the delay curves in a given correlogram. Each target typically produces several delay lines in the correlogram corresponding to the direct path and several multipath reflections. It is necessary to identify this group of lines or linesets as belonging to one target. This operation is performed by the line-set selector. Ideally this operation should be performed automatically, but here we have assumed that linesets are selected by an operator. The reason for this choice is that automatic line selection is not an easy task and requires reliable algorithms that have not yet been developed.

In order to implement a reliable and workable processing system we opted for manual selection of linesets by operator intervention. The system prompts the user/operator to select the linesets. The operator enters the selections by choosing several points on the desired line-set. These points are then automatically passed on to the line tracker subsystem and are used to initialize the line tracking routines. The line tracker extracts delay vs. time curves out of the correlogram data. The line tracker consists of two types of algorithms.

The first algorithm is a sequential algorithm based on maximizing the a-posteriori probability of detecting the delay from the noisy correlogram data. This algorithm needs some initial points in the correlogram domain that are interactively selected by the user. The
second algorithm is a batch method that uses cubic splines to fit a curve to a number of points selected on the delay curve. In this method one can select a number of points, fit a curve and display the computed curve on top of the actual curve. If the fitted curve is not close to the actual curve, then additional points can be selected in the regions where the fit is not good and a new curve is then generated. This process can be repeated until the fitted curve is as close to the actual curve as desired. The cubic spline method is discussed in Section 2.4.

The fourth subsystem in Figure 1 is concerned with track parameter estimation. Here a host of algorithms can be used. These algorithms and their performance are presented in detail in [2],[3]. The input to these algorithms are the delay vs. time curves or tracks extracted by the line-tracker. The output of these algorithms are the $x$, $y$, $z$ coordinates of the target and the target velocity $V$.

Finally the display subsystem provides the user with the capability to view the output of each processing step. It consists of routines to generate X-Y plots, Waterfall plots and grey scale or halftone plots. The grey scale plots such as the one shown in Figure 2 can be generated both on the screen and on hardcopy devices such as a Printronix printer, Talaris or Apple Laser printer. The plotting routines are designed to work with a Tektronix 4014 terminal. The display functions and the software are discussed in Section 3. Appendix A is a list of routines comprising the software package, and Appendix B is a brief user's guide.
Fig. 2. An Example of Correlogram

2.0 Algorithms

This chapter describes the algorithms used for the implementation of the processing system. The discussion in this chapter is brief and is intended to highlight the essential features of each algorithm only. Details are given in the references. Section 2.1 describes the corre-
ogram generation, Section 2.2, 2.3, and 2.4 discuss the line-tracking algorithms. Section 2.4 address the localization problem and Section 2.5 describes the display functions.

### 2.1 Correlogram Generation

A correlogram is a two-dimensional picture showing the time history of differential delays between the signals at two sensors (cross correlogram) or differential delays between a signal (wavefront) and its multipath reflection.

Figure 2 shows a typical correlogram. The vertical axis in this figure is the time axis \( t \) with the positive direction from the top of the page to the bottom of the page. The horizontal axis in this figure is the correlation lag or delay axis \( \tau \) with zero delay corresponding to the middle of the horizontal axis. Let \( x(n) \) and \( y(n) \) be the signals at two different sensors or they may be the signal and its multipath reflection at one sensor. The cross correlation between \( \{x(n)\} \) and \( \{y(n)\} \) at lag \( k \) is given by

\[
R_{xy}(k) = \sum_{n=0}^{N-1} x(n+k)y(n) \quad k = 0, 1, \ldots, M - 1
\]  

(2.1)

where \( N \) is the number of samples in the data sequences \( x(n) \) and \( y(n) \), and \( M \) is the maximum lag or delay for which correlation is computed \( (M < N) \). To generate a correlogram, the data interval is divided into segments of length \( N \).

Each horizontal line in Figure 2 corresponds to one data segment for which the correlation is computed according to (2.1). The actual implementation of a correlogram differs from that in equation (2.1) in two important respects:

(i) The straightforward implementation of (2.1) requires a total of \( MN \) operations (multiplications and additions) for each horizontal line (one data segment) of the correlogram. For a total of \( L \) lines in the correlogram, the total number of operations \( T \) is

\[
T = LMN
\]

(2.2)

Typical values of these parameters are \( L = 1000 \), \( M = 1000 \), and \( N = 2000 \). Thus \( T = 2 \times 10^9 \). These parameters are for generating an 8-minute correlogram from lags -250 ms to 250 ms of a data set sampled at 2048 samples per second. The number of computations for 8 minutes of this data is two billion operations, taking about 80 minutes of CPU time on a 0.4 megaflop computer. The actual computation time could be longer if the CPU is not
dedicated. Hence the computational requirements are demanding. A faster implementation is to use FFTs as follows.

It is a simple matter to show that

\[ R_{xy}(k) = DFT^{-1}[DFT(x(n)) \cdot DFT^*(y(n))] \quad k = 0, 1, \ldots, N - 1. \] (2.3)

That is to compute \( R(k) \) one simply computes the FFTs of the two data sequences \( \{x(n), n = 0, \ldots, N - 1\} \) and \( \{y(n), n = 0, \ldots, N - 1\} \) conjugates one of the FFTs, performs a point by point multiplication and computes the inverse FFT of the product. It should be noted that this procedure produces the correlation sequence \( R(k) \) for all lags \( k = 0 \) to \( N - 1 \) which in general more than necessary because typically \( M < N \).

The number of computations to implement (2.3) is given by

\[ C = 15N \log_2 N + 4N \] (2.4)

Each \( N \)-point FFT requires \( 5 \log_2 N \) operations and the multiplication of the two sequences requires \( 4N \) real operations, the total adds up to the quantity in (2.4). Hence the total number of computations using the FFT method is given by

\[ T_1 = LC + L(15N \log_2 N + 4N) \] (2.5)

For the parameter values of the previous example, \( L = 1000, N = 2000 \), we have \( T_1 = 0.338 \times 10^9 \) which is 6 times less computations than the implementation in (2.1). Whereas the implementation in (2.1) requires 80 minutes of CPU time, the implementation of (2.3) only requires 14 minutes of CPU.

\[ \text{Fig. 3a. Time-Domain Normalization (Prefiltering)} \]
(ii) The second implementational detail in generating correlograms is the normalization issue. Normalization is done to sharpen the peak of the autocorrelation function $R(k)$ in order to reduce the ambiguity in delay estimation. Several normalization strategies are suggested in the literature. Figure 3 shows a general normalization (also referred to as prefiltering) scheme. Figure 3a shows the time-domain normalization and Figure 3b shows the frequency-domain normalization. The normalization essentially amounts to prefiltering the signals $x(n)$ and $y(n)$ and feeding the filtered signals $\hat{x}(n)$ and $\hat{y}(n)$ to the cross correlator. It can be shown that [4] the generalized correlation between $\hat{x}(n)$ and $\hat{y}(n)$ is given by

$$R_{\hat{x}\hat{y}} = DFT^{-1}[\psi(m)G_{xy}(m)]$$  \hspace{1cm} (2.6a)

where

$$\psi(m) = H_1(m)H_2^*(m)$$ \hspace{1cm} (2.6b)

and $H_1(m)$ and $H_2(m)$ are the transfer functions of the two prefilters (Figure 3a) and $G_{xy}(m)$ is the cross power spectrum between the input signal $x(n)$ and $y(n)$

$$G_{xy}(m) = DFT\{R_{xy}(k)\}$$ \hspace{1cm} (2.7)

Table 2.1 shows some common choices for the weighting function $\Psi(m)$ with related references given in the last column. The function $\gamma_{xy}$ in the table is the coherence function:

$$\gamma_{xy}(m) = G_{xy}(m)/\sqrt{G_{xx}(m)G_{yy}(m)}.$$

For a detailed discussion of these schemes refer to [4]. For this project we have selected the SCOT normalization with

$$\psi(m) = |H_x(m)H_y^*(m)|^{-1}$$ \hspace{1cm} (2.8)
Table 2.1 Normalization Schemes

<table>
<thead>
<tr>
<th>Normalization</th>
<th>Weighting Function $\psi(m) = \tilde{H}_1(m)\tilde{H}_2^*(m)$</th>
<th>Relevant References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross Correlation</td>
<td>$1$</td>
<td>[5–7]</td>
</tr>
<tr>
<td>Roth Impulse Response</td>
<td>$1/G_{xx}(m)$</td>
<td>[9]</td>
</tr>
<tr>
<td>SCOT</td>
<td>$\frac{1}{\sqrt{G_{xx}(m)G_{yy}(m)}}$</td>
<td>[10–11]</td>
</tr>
<tr>
<td>PHAT</td>
<td>$1/</td>
<td>G_{xy}(m)</td>
</tr>
<tr>
<td>ML (Maximum Likelihood)</td>
<td>$\frac{</td>
<td>\gamma_{xy}(m)</td>
</tr>
</tbody>
</table>

With this choice

$$R_{xy}(k) = DFT^{-1}[\phi(m)]$$

where $\phi(m)$ is the phase of $G_{xy}(m)$

$$\phi(m) = \text{phase of} \ [\tilde{H}_x(m)\tilde{H}_y^*(m)] \quad (2.9)$$

The reason for this choice is that if $y(n)$ is a truly delayed version of $x(n)$, that is

$$y(n) = x(n - D) \quad (2.10a)$$

then

$$H_y(m) = H_x(m)e^{-j\omega D} \quad (2.10b)$$

and from (2.9)

$$\phi(m) = -\omega D \quad (2.11a)$$

($\omega$ is the continuous frequency variable)

So that

$$R_{xy}(k) = \delta(k - D) \quad (2.11b)$$
where $\delta(k)$ is the Kronecker delta function. Thus the cross correlation function is an impulse at the true delay $D$ — an infinitely sharp function.

**2.2 Feature Extraction**

In Sections 2.3 and 2.4, we discuss the extraction of the time history of a set of differences in time of arrival (TDOAs) embedded in a correlogram—the function performed by the third block in Figure 1. The time history of a single TDOA will be referred to as a line, and the process of determining the set of lines present in a correlogram will be called feature extraction.

Untrained humans can easily trace the TDOA history in a correlogram, even in low-signal-to-noise multiple-TDOA cases such as Figure 1. This procedure is, unfortunately, not easily automated. Depending on operational requirements, one of the two classes of algorithms are used: sequential and batch. Sequential algorithms, including the commonly used ADEC, form an estimate of the line set which is successively updated based on the new correlation function and assumed TDOA dynamics. Batch methods, such as MAPLE for the case of frequency tracking, look at an entire segment of a correlogram, searching for the set of lines best fitting the data and assumed dynamics. In general, a low-signal-to-noise ratio, the lack of a good model for the TDOA dynamics, and the presence of multiple, closely spaced—often the case in practice—leave these methods incapable of accurately finding TDOA lines.

For this project, sequential and batch feature extractors were implemented. Both take advantage of interaction with a human operator to overcome the difficulties associated with the fully automated techniques. The sequential extractor is a modest extension of the single line extractor recently presented by Bethel and Rahikka [1]. The batch extractor fits a curve to a set of points selected by the operator; the operator adjusts the set of “interpolation points” until, in his judgement, the fitted line corresponds well with the one appearing in the correlogram. In the following sections, the implementation of the batch and sequential feature extractors are described in detail; their use as part of the software system is described in section 3.0.

**2.3 Sequential Method**

The sequential method of feature extraction used here is the one described in [1]. The basic idea is to treat the delay value as a discrete first-order Markov process, and use the correlation slices to recursively update an estimate of the probability distribution of the
delay value. The peak of the distribution at a given time step is taken to be the delay value at that time.

**Bayesian Approach**

Denote by $R_j$ the $j \times n$ matrix consisting of the first $j$ rows of the correlogram. The $1 \times n$ matrix $r_i$ is the correlation function at time $i$—the $i$th row of the correlogram; its $j$th entry $r_1(j)$ is the $j$th correlation lag at time $i$. The value of the delay at time $k$, denoted by $d_k$, is assumed to take on one of $2N + 1$ discrete values from the set $\{-\delta N, -\delta(N - 1), \ldots, 0, \delta, \ldots, \delta N\}$, chosen to correspond with the correlation times. The probability that given correlation data up to time $i$ $R_i$, the delay value at time $i$ is $j\delta$, is denoted

$$\varphi_i(j) \overset{\text{def}}{=} \varphi\{d_i = j\delta | R_i\}$$

It is desired to write $\varphi_i(j)$ in terms of $\varphi_{i-1}(k)$, $k \in [-N, N]$.

Using Bayes’ rule, $\varphi_i(j)$ can be written in terms of $R_{i-1}$.

$$\varphi_i(j) \overset{\text{def}}{=} \varphi\{d_i = j\delta | R_i\} = \frac{\varphi\{r_i | d_i = j\delta, R_{i-1}\} \varphi\{d_i = j\delta | R_{i-1}\}}{\varphi\{r_i | R_{i-1}\}} \quad (2.12)$$

Assuming that the signal and noise are independent across time slices,

$$\varphi\{r_i | d_i = j\delta, R_{i-1}\} = \varphi\{r_i | d_i = \delta\}, \quad (2.14)$$

and

$$\varphi\{r_i | R_{i-1}\} = \kappa, \forall i. \quad (2.15)$$

The relation (2.12) can then be written as

$$\varphi(j) = \kappa \varphi\{r_i | d_i = j\delta\} \varphi\{d_i = j\delta | R_{i-1}\}, \quad (2.16)$$

where $k$ may be thought of as a normalization constant.

Assuming a first-order Markov model for the changes in $d$ from one time step to the next, the quantity $\varphi\{d_i = j\delta | R_{i-1}\}$ may be expressed as follows:

$$\varphi\{d_i = j\delta | R_{i-1}\} = \sum_{k=-N}^{N} \varphi\{d_i = j\delta | d_{i-1} = k\delta\} \varphi\{d_{i-1} = k\delta | R_{i-1}\}. \quad (2.17)$$
Defining $p_i$ as the vector with $k$th element $p(d_i = k\delta|R_i)$, and $P$ as the matrix with $nm$th element $p_{nm} \equiv p(d_i = n\delta|d_{i-1} = m\delta)$, (2.17) can be written more compactly as

$$p(d_i = j\delta|R_{i-1}) = e_j^T P p_{i-1}, \quad (2.18)$$

where $e_j$ is the $j$th column of the identity matrix. The quantity $p_i(j)$ then becomes

$$p_i(j) = \kappa p(r_i|d_i = j\delta)(e_j^T P p_{i-1}). \quad (2.19)$$

To find $p_i(j)$ in terms of $p_i$, it remains to evaluate $p(r_i|d_i + j\delta)$. This problem was considered in [14] for the case of a single TDOA present, the result being

$$p(r_i|d_i = j\delta) = C e^{r_i(j)}. \quad (2.20)$$

Define $\rho_i$ as the vector with $j$th element $e^{r_i(j)}$. Then, the probability distribution of $d_i$, $p_i$ can be written in terms of the $i$th correlation function and the distribution of $d_{i-1}$, $p_{i-1}$, as follows

$$p_i = K \Lambda \rho_i P p_{i-1}, \quad (2.21)$$

where $\Lambda \rho_i$ is a diagonal matrix with $j$th row-column entry $e^{r_i(j)}$, and $K$ is a constant adjusted so that $1^T p_i = 1$.

The recursion (2.21) generates a sequence of probability distributions on the single TDOA value $d_i$, assuming a first-order Markov model with known state transition matrix $P$, and given an initial distribution $p_0$ and set of correlogram traces $R_k$. The estimated TDOA value at any time $j$, $\hat{d}_j$, is given by the value $k\delta$ maximizing $p_j$—that is, $\hat{d}_j = \delta \text{Arg}[\max_k \{p_j(k)\}]$. As detailed in [1], this algorithm performs very well when a single line is present.

Implementation

Due to the presence of multiple lines, the algorithm implemented in the software package differs slightly from the one described above. Details are described below.

State Transition Matrix It is assumed that it is just as likely for a source at a given location to be moving in one direction as it is for it to be moving in the opposite direction. Accordingly, the state transition matrix is assumed symmetric. The transition from one state $d_i$ to another $d_{i+k}$ is assumed to be only a function of the difference $k$. Therefore $P$ is approximately Toeplitz. (Depending on the array-source geometry, this assumption may not be valid, as shown in Figure 4.) The $d_i$ to $d_{i+k}$ transition is assumed to take place with
\( k \) exponentially distributed, which, for reasons of computational complexity, is truncated. The state transition matrix used is then a symmetric Toeplitz matrix with first row

\[
\begin{bmatrix}
1 & \alpha & \cdots & \alpha^q & 0 & \cdots & 0
\end{bmatrix},
\]

(2.22)

weighted by a diagonal matrix so that all rows sum to one. Note that \( \alpha \in (0,1) \), and in the software, \( \alpha \approx 0.7 \) and \( q \approx 10 \).

**Line Extraction** Given the state transition matrix, an input correlogram and a priori density function \( \varphi_0 \) will generate a sequence of density functions \( \varphi_i, i = 1, \ldots, M \). In the case of one TDOA, Bethel and Rhikka [1] suggest that the maxima of the \( \varphi_i \) form an estimate of the line. When there is more than one line present—and (2.20) is assumed to approximately hold—the top several peaks in \( \varphi_i \) may be tracked. However, since the order of the distribution function maxima is not constant from time step to time step, and the number of TDOAs are unknown, tracking the location of the top \( \ell \) maxima in the \( \varphi_i \) sequence is not much different than tracking the top \( \ell \) maxima given the \( r_i \) sequence.

To circumvent these problems, we propose the following solution: have the operator select a particular line to track by giving the recursion (2.21) a set of *way points*, and reading out the maximum of the resulting \( \varphi_i \). As illustrated in Figure 5, way points are points in \( R_j \) through which the line of interest passes. At time step \( i \), the recursion (2.21) is used to generate \( \varphi_{i+1} \), unless there is a way point at that time, in which case \( \varphi_{i+1} \) becomes the distribution with all weight on the delay associated with the way point. The maxima of the \( \varphi_i \) form the extracted line. The operated-assisted line extraction procedure is then (a) select a way point at the beginning of the line and run the recursion (2.21), tracking the peak of the resulting \( \varphi_i \); (b) stop if the line is properly tracked; or (c) enter a new way point just after the estimated track when astray and go to (1). Note that since (2.21) requires of order \( 20n \) computations per time step [assuming the simplified form of \( P \) (2.22)], this method is relatively quick. Also, since the operator can improve the line estimate from one iteration to the next, a good-quality line (one matching the correlogram) is guaranteed.

### 2.4 Block Method

The block methods extract features by searching over a parameterized set of lines for the ones best fitting the data according to some cost function. This process tends to be very computationally intensive due to the expense in computing the cost function as well as the fact that the cost function is typically nonconvex. Humans, however, can easily decide
Fig. 4. Toeplitz State Transition Matrix Assumption

Shown in the figure above is a two sensor array and lines of constant TDOA. Note that generally a source can transverse a given number of TDOA lines—i.e., change state form $d_i$ to $d_{i+k}$ for some $k$—more quickly when it is aligned with the sensor axis—the case of $i$ large—than when it is aligned with the perpendicular axis—the case of $i$ small. Consequently, the state transition matrix given in (2.22) should have a slightly larger $\alpha$ in the middle of the matrix.
whether a given line “fits” the correlogram, and how the line should change to fit better. In this section, we propose an operator-based line tracker which takes advantage of this ability.

Approach

The method is as follows. As in the sequential case above, the operator selects a set points through which the line passes. A smooth curve is interpolated through the points, and displayed so that the operator may compare the interpolated curve to the desired correlogram line. The operator adds interpolation points (i.e., way points) until the interpolated curve and the correlogram line are in good agreement.

Implementation

It remains only to choose a method for fitting a smooth curve through a set of points. Among available methods, the cubic spline [15, pp. 198-205] was chosen for its simplicity.

The cubic spline is the curve passing through a set of points \((x_i, y_i), i = 1, \ldots, p\), described by

\[
y = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + y_i
\]

(2.23)
in the interval \(x \in [x_i, x_{i+1}]\); the coefficients \(a_i, b_i, \text{ and } c_i\) are chosen so that the curve be-
tween \((x_i, y_i)\) and \((x_{i+1}, y_{i+1})\) joins smoothly—matching the value, first and second derivatives—with the curve from \((x_{i+1}, y_{i+1})\) to \((x_{i+2}, y_{i+2})\). The coefficients are computed as follows.

Define \(h_i \overset{\text{def}}{=} x_{i+1} - x_i\), then

\[
\begin{align*}
a_i &= \frac{1}{h_i} (s_{i+1} - s_i) \\b_i &= 3s_i \\
c_i &= \frac{1}{h_i} (y_{i+1} - y_i) + (h_i s_{i+1} + 2h_i s_i),
\end{align*}
\]

where \(s_1 = 0, s_p = 0, \) and \(s_i, i \neq 1, p\) are given by

\[
\begin{bmatrix}
2(h_1 + h_3) & h_2 & 0 & \cdots & 0 \\
h_2 & 2(h_3 + h_4) & h_3 & \ddots & \vdots \\
0 & \ddots & \ddots & \ddots & 0 \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & h_{p-2} & 2(h_{p-2} + h_{p-1})
\end{bmatrix}
\begin{bmatrix}
s_2 \\
\vdots \\
s_{p-1}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\frac{h_1 y_3 - (h_1 + h_2)y_2 + h_2 y_1}{h_1 h_2} \\
\vdots \\
\frac{h_{p-1} y_{p-1} - (h_{p-2} + h_{p-3})y_{p-2} + h_{p-2} y_{p-3}}{h_{p-2} h_{p-3}}
\end{bmatrix}
\]

An example cubic spline is shown in Figure 6.

Comparison

Comparing the block method to the sequential method (see the two examples presented above as well as others included in the User's Manual section of this report) we see that the sequential method generally requires less way points, and perhaps better follows the details of the line being tracked. The advantages of the batch approach are in the form of the output—the interpolated polynomial is easy to store and manipulate—and the computations required—the batch method only takes of order \(p^3\) operations to produce a interpolating curve, \(p\) being the number of way points.

2.5 Display Functions

2.5.1 Overall Description

The display functions are FORTRAN programs designed for assisting with the inter-
Fig. 6. Example of Cubic Spline

In the figure above, interpolation points are signified by circles, and the interpolated values—found by cubic spline—are shown as dots.

active operation of the multipath line tracking capability. The basic display for the user is produced on a Tektronix 4014 terminal. The data is primarily displayed as a greyscale plot. There are several options available with the greyscale representation. These options are described in Section 2.5.2 of this report.

Next, the setting relative to the screen pinpoint positions and the data are initialized. This information is used later for assisting in selecting points from the screen, and initializing the line tracker. Additional parameters are set for the line tracker.

The user is allowed both a zoom and an offset capability when displaying the data. With this capability, the user can slide through the data and display sections. These sections can then be enhanced through the zoom function.

Once the data is selected containing the line for tracking, the user can evoke cross-hairs on the screen for point selection. The user can manipulate the cross-hair across the screen by using the thumbwheels located on the Tektronix 4014 terminal. The thumbwheels allow for continuous movement of the cross-hairs in all directions on the screen. The user is then allowed to select points from the screen. Once these points are selected, the intensity of the
pixels which represent that point of data are amplified. Thus, the beam output of these pixels increases and the point visually stands out to the user from the rest of the greyscale displayed on the screen.

After the user has finished selecting points from the screen, the user enters back into the alpha-numeric display where a final editing can be performed on the points selected. This selection is made by the user visually inspecting the location and value of the point selected. Once final selection is completed these points and associated values are sent to the line tracker for tracking.

The line tracker produces a matrix of ones and zeros which represents a pure line tracked through the data. The matrix is the same size as the original greyscale of the real data produced on the screen before point selection.

The matrix of ones and zeros is then compared with the original raw data matrix on a point-by-point basis. The maximum value resulting from this comparison is then stored in a third matrix. This third matrix is then displayed as a greyscale; thus, producing an overlay effect where the line tracker predicted the line is highlighted and the original data is in the background. The user can then repeat this process.

2.5.2 Greyscale on the Screen

When the line tracking option is selected, the user is first prompted for a file name. Once the user selects a file, the system then loads the file into the main array. Quite frequently, the array size will exceed the resolution of the screen. The greyscale then provides the user with several options for displaying the data.

The first prompt of the greyscale display is a question to either change parameters, plot data, or leave and return to the main menu. The greyscale will wrap the array around across the screen with a row consisting of 512 points. The first row contains the first 512 points. This wrapping function produces a matrix for display. The “change parameters”, is the option which allows the user to plot portions of the data array. Several parameters can be changed.

Offset

First, the user can change the $x, y$ coordinate of the first point in the array to be plotted. This essentially slides a viewing window down the matrix. Figure 7 illustrates this sliding
Fig. 7. Example of Viewing Window:
The dots represent data points.
The box illustrates the viewing window.
The arrows indicate a new start point.
The default is the first element of the first column.

capability. The default value is $x$ of 1 and $y$ of 1. This corresponds to the first element in the array.

Zoom

Another option is to change the total number of points to be viewed. This change produces a zoom effect on the data. This change is illustrated in Figure 8.

Fig. 8. The dots represent the data points.
The box illustrates the reduced viewing window.

There are several other options which allow the viewer a "better display" of the data. Once an appropriate array size is selected, the program takes the array or subarray and divides the data into 16 bins. The data ranges from the maximum value in the first bin, to the minimum value in the 16th bin. Several pixels are selected to represent a data point and
a beam intensity (between 1-16) allowed for those pixels. The intensity depends upon the bin where the data value falls. The number of pixels per point is a function of the screen resolution and the array size being plotted. The user may also change the number of pixels to a point by changing the zoom parameter. These options allow the user great flexibility when viewing data.

2.5.3 Interaction with Line Tracker for Initialization

Every time the user selects the plot option for the greyscale, the \( x, y \) screen coordinate of the first point displayed is saved. Also saved is the position of the sub-array being plotted. If no sub-array is being plotted, then the default is the whole array. This information is later used by the line tracker to reconstruct the data points selected from the \( x, y \) screen coordinates resulting from the cursor selection.

2.5.4 Cursor Functions

Once the user had displayed the data in a satisfactory manner with the greyscale options, the cursor option can be enacted by typing 0 at the prompt. Typing 0 enters the user into another menu called screen operations which provides the user with 6 choices as follows:

1. CUR display the cursor on the screen
2. ADD add the point selected by the cursor
3. CLR clear all points selected by the cursor
4. DEL delete the point selected by the cursor
5. EXI exit this program
6. PRN print out the coordinates of the selected

SCREEN OPERATION (HELP=?) ?

To exercise any option either the option number (1 through 6) or the first two letters of the option must be entered. For example to add new points either enter 2 or AD (or ad). A brief description of these options follow.

2.5.4.1 CURSOR

As soon as the user selects the cursor option from the menu, a cross-hair appears on
the screen. The 4014 terminal has thumbwheels which the user can use to manipulate the cross-hairs across the screen.

2.5.4.2 ADD POINT

The user moves the cross-hairs across the screen until they wish to select a point. To select the point, the user strikes any key. The key stroke signals the software to store the \( x, y \) coordinate from the screen which is at the cross section of the cross-hairs. The beam intensity of the pixels, which represent the data point selected, then increases to the maximum intensity. This then highlights the point selected. This operation can be performed for as many points as desired.

2.5.4.3 Clear Points

Entering 3 or CL (or cl) will clear the buffer containing the selected points and allows the user to start all over again.

2.5.4.4 DELETE POINT

The Delete option allows the user to delete any point from the list of selected points. To use this option the user must first view the selected points using option 6. The result of typing 6 or PR (or pr) is shown below:

<table>
<thead>
<tr>
<th>NO.</th>
<th>X-COORDINATE</th>
<th>Y-COORDINATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>232</td>
<td>544</td>
</tr>
<tr>
<td>2</td>
<td>139</td>
<td>379</td>
</tr>
<tr>
<td>3</td>
<td>651</td>
<td>562</td>
</tr>
<tr>
<td>4</td>
<td>791</td>
<td>672</td>
</tr>
<tr>
<td>5</td>
<td>768</td>
<td>68</td>
</tr>
<tr>
<td>6</td>
<td>512</td>
<td>105</td>
</tr>
<tr>
<td>7</td>
<td>325</td>
<td>141</td>
</tr>
</tbody>
</table>

The selected points are listed in the chronological order that they were selected and their \((x, y)\) coordinates are also shown. The DELETE option inquires the user as to what point to be deleted. A sample of the inquires and the user response is as follows:
SCREEN OPERATION (HELP=?) DE
Want to DELETE a point (YES) ?
Value = Y
points are ordered according to the PRINT list
WHICH POINT DO YOU WANT TO DELETE ? MN = 1 MX = 7
DF = 7 ? 5
Value = 5

Note in the above example that the total number of points (seven) is shown. At the question mark (?) the user enters the number 5 indicating that he wants to delete the fifth point. After this operation is complete the user can verify that the point was deleted and look at the new list by using option 6. The result of this operation is shown below.

SCREEN OPERATION (HELP=?) ?PR
NO. X-COORDINATE Y-COORDINATE
1 232 544
2 139 379
3 651 562
4 791 672
5 512 105
6 325 141

Note that the total number of points has been reduced from 7 to 6 and the points following the deleted point have been renumbered. The default for the DELETE operation is the last selected point.

2.5.4.5 EXIT

Entering 5 or EX (or ex) will exit the user from screen operations menu back to the greyscale plotting menu.

2.5.4.6 PRINT

Entering 6 or PR (or pr) will cause the (x, y) coordinates of the selected points to be printed on the screen.

2.5.5 Overlay/Highlighting

After the line tracker completes an array, the same size as the original array is returned
to the display software. The array containing the line tracker output contains ones and zeros. The ones are the values representing the predicted line.

A comparison is then made, on a point-by-point basis of the original data and the line tracking array. A third array is created from this comparison consisting of the maximum value from the comparison. This maximum value array is then plotted and the predicted line is highlighted. This visually overlays the line tracker results with the original data and the user can actually view the performance of the line tracking algorithm.
3.0 Software Description

We have developed a signal processing package containing over 400 FORTRAN subroutines. The structure of the software is designed for real-time operation. By having this structure, the software allows the user to interface with the code on a combination of levels. Currently, the software is running on a VAX 11-750. The graphics interface of the code is mainly designed for a Tektronix 4014 device. DEC's Command Language (DCL) is the first user interface level accessed. The Command Language provides users with an extensive set of options for interactive program development, device and data file manipulation, and finally, interactive batch program execution and control.

3.1 FORTRAN Library of Signal Processing and Utility Functions

The FORTRAN subroutines compose a large library of over 400 subroutines. These subroutines provide the user with a large set of signal processing functions. The software is divided into three basic sections. The first section consists of the signal processing functions, the second consists of the utility functions, and the third, the basic line tracking capability. The user directly interfaces with these three levels. The basic description of the structure and use of the line tracking capability is described in Section 2.5. The signal processing and utility functions are accessed through different levels of user selected options. Included in this large FORTRAN library, are all the appropriate device drivers for both the graphic and interactive communications. The entire software is menu-driven and designed for real-time use.

The FORTRAN subroutines are designed to operate as a fully integrated signal processing system. The library subroutines and associated programs are designed in a modular fashion to facilitate a series of operations and functions which accomplish various signal processing tasks. The first level of interaction for the user is with the main menu. This menu allows the user to enter either the DCL data level, or into the signal processing and utility area, or directly into the line tracking process.

The DCL level generally performs file acquisition. This level queries the user for the name of the specific file to be loaded, performs the actual file input and output, and then proceeds to call the programs to perform different tasks with the data. The final functions enacted are the statistics. The results are then displayed for the user to view.

The following is the list of operations available in the utility menu.
Legal operations...
Operat-arrays.. description

? .. HELP
$ DCL COMMAND .. WILL EXECUTE DCL COMMAND
EXIT .. EXIT FROM PROGRAM

************************** DATA MANIPULATION OPERATIONS **************************

ZERO-A,B,C,D,E.. MAKE ZERO ARRAY
PARAM -A,B,C,D.. SET OR CHANGE ARRAY PARAMETERS
DIM -A,B .. SET THE SUBAREA DIMENSIONS
AND SET Min,Max,Ave,Sdev.
FXTAIL- .. FIX TAIL ON FILE
LS- .. LIST AND CHANGE ARRAY CONTENTS
MANUAL-A .. MANUALLY ADD,DELETE,CHANGE OR LIST ARRAY
MIN .. SET Min, Max USING ACTUAL DATA
PARAM -A,B,C,D,E SET OR CHANGE ARRAY PARAMETERS
SDEV -A SET Min, Max USING STANDARD DEVIATION
SET - SET Min, Max INTERACTIVELY

************************** STATISTICAL OPERATIONS **************************

HIST -A .. COMPUTE A HISTOGRAM
LI - .. LIST SIZE AND STATISTICS
STAT -A .. CALCULATE STATISTICS
SNR - .. SIGNAL TO NOISE RATIO CALCULATION

************************** GRAPHICS **************************

LEVEL -A .. LEVEL Plot on the CRT
PLOT - .. X.Y PLOT WITH INTERACTIVE INPUT OF DATA
PTX -A .. HALFTONE PLOT ON THE PRINTRONIX
By entering SP at the utility level, the user enters the signal processing level.

The signal processing level is designed for the analysis of underwater signals and consists mostly of signal processing functions. These functions allow for a range of parametric and non-parametric feature extraction. The following is a list of options available in the signal processing menu.
RESULTS WILL BE PUT INTO [A, (B)]

SYN...... SYNTHETIC DATA OPTION
AR...... AR SPECTRAL FEATURES
RMA...... ARMA SPECTRAL FEATURES

BPF...... BAND PASS FILTERING
NG...... ENERGY FUNCTION

EXIT.... EXIT OPERATION
$....... ACCESS TO DCL
STATS... GET STATISTICS

SROOT... SQUARE ROOT
MAG...... MAGNITUDE
MSQD.... MAGNITUDE SQUARED
SCALE... SCALE AND OFFSET (K1*A + K2)
LOG...... LOGRITHM BASE 10 OR e
PHASE.... PHASE
PPIC.... PICK THE PEAKS & CORRESPONDING AMPLITUDES
XPON.... EXPONENTIAL (10 OR e) (EXP(K1*RE(A)+K2)
SQUARE.. SQUARE OF DATA
ZCR...... ZERO CROSSING RATE
+........ [A,(B)] + [C,(D)]
-........ [A,(B)] [-C,(D)] OR R
-........ [A,(B)] [-C,(D)] OR REVERENCE
*........ [A,(B)] * [C,(-D)]
/........ [A,(B)] / [C,(D)] OR REVERENCE
1DFFT... 1-DIMENSIONAL FFT
FFTM.... FFT MAGNITUDE
2dFFT... Two-Dimensional FFT
i2dFFT... Inverse two-dimensional FFT
I1DFFT... INVERSE 1-DIMENSIONAL FFT
HBFREQ... HANNING WEIGHTING BY CONVOLUTION (FREQ)
The utility support software is used either directly by the user or as a subprocess of a signal processing function. The data is manipulated by using five basic arrays: A-E. The utility programs mainly address the manipulation of these five arrays. These arrays are used for storing and setting parameters relative to the data. The utilities used for this are ZERO, MANUAL, FXTAIL, PARAM, COPY, LIST, STAT, and DIM. Additional utilities are HIST, SET, and MIN. The final set of utilities are basically for display of the data. These consist of LEVEL, PTX, PRSPCT, SLICE, PLOT and WATERF. A description of each of these functions mentioned above can be found in the utilities menu. Appendix A lists the subroutines used in the software package.
4.0 Conclusions

This report attempted to provide a description of a processing system for multipath data which was developed in the course of this project. The main components of this system are a correlogram generator, a line set selector, a line tracker, a track parameter estimator and some display functions. The goal of this phase of the project was to integrate the results of research conducted during the previous contracts into a software system that embodies most of the relevant algorithms developed for multipath delay estimation, localization and tracking. This system provides a workbench for experimentally studying the performance of these algorithms on sonobuoy data. It also acts as a demonstration system to see how realistic and useful such a system can be for coping with the problems encountered in real data.

In the development of this software, the goal was to maximize the its utility by making it as easy to use as possible. This was accomplished by making as many of the algorithms and processing functions as automatic as possible. The functions that are fully automated, such as the correlogram generator, are the ones which are most reliable and whose performance is well understood. Functions or algorithms such as line set selection were made semi-automatic or manual (requiring operator intervention) since at this time we do not feel they are sufficiently reliable and fully tested.

Special emphasis was given to versatile graphic capabilities to provide visual aid to the system operator/user. These capabilities include several plotting options on the screen. Of special importance is a greyscale plotting capability with options to select/change the plotting region, the viewing angle and the granularity of the greyscale plot. There is a cursor positioning function which can be invoked in the greyscale mode. The cursor positioning function is a reliable and practical way of selecting linesets and initial points for tracking. One can select as many points as desired simply by positioning the cursor and hitting RETURN to automatically pass the coordinates of the selected points to the line tracker. There are also options to add or delete a point or to clear all the points and re-start the selection process. The Tektronix 4014 was selected as the plotting device for graphics functions because of its popularity and the fact that many other terminals are compatible with it.

We attempted to make the user interface as friendly as possible by providing an interactive environment and also the necessary queries to the user for each selected function. Another important consideration in designing the software was the issue of transport.
ity. The software was written in VAX FORTRAN 77 and can be easily ported to any VAX 11/750 or 11/780 or other computers which are compatible with these machines. The system is designed so that new algorithms or functions can be easily incorporated into the system.

The current version of the software is a working version and can be used to perform experiments with real data. However, there are a number of specific improvements which can enhance the performance of this processing system. Color graphics would make the displays more easily to read and interpret and also provide higher resolution. The addition of an automatic line set selection algorithm would be very useful. As the software is exercised on real data other improvements and modifications are likely to be suggested.
References


29

Appendix A: List of Subroutines in the Multipath Processing Software

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A001.FOR</td>
<td>Block data for TEKMG</td>
</tr>
<tr>
<td>ABSE</td>
<td>REAL+4 function for absolute values</td>
</tr>
<tr>
<td>ADATA</td>
<td>General ARMA filtered white noise (IMP=0) or the Impulse response of an ARMA filter.</td>
</tr>
<tr>
<td>ADDCON</td>
<td>Add a constant to array</td>
</tr>
<tr>
<td>ADDMX</td>
<td>Add files N1 and N2</td>
</tr>
<tr>
<td>ADDWIT</td>
<td>Add white noise to array</td>
</tr>
<tr>
<td>ADNOIS</td>
<td>Add zero mean Gaussian noise to a signal with a given</td>
</tr>
<tr>
<td>ARCLR</td>
<td>Clear (zero out) array</td>
</tr>
<tr>
<td>ARCPY</td>
<td>Copy NS elements from array S to first NS elements of array D, and zero-fill the rest of D. if NS&lt;0, the negated elements of S are transferred to D.</td>
</tr>
<tr>
<td>ASYMTR</td>
<td>Perform transpose for subarray TRANSPO using TEMP DISK FILE</td>
</tr>
<tr>
<td>BDRBOT</td>
<td>Put border on bottom of plots</td>
</tr>
<tr>
<td>BORTOP</td>
<td>Put border on top of plots</td>
</tr>
<tr>
<td>BSORTTR</td>
<td>Bubble sorter for real arrays. A(N) is sorted to have decreasing values. The array K is returned having the index map used. A(I) upon return equals the original A(K(I)).</td>
</tr>
<tr>
<td>BUFOUT</td>
<td>Output buffer manipulation</td>
</tr>
<tr>
<td>CALC</td>
<td>Interactive calculator with reverse polish notation and unlimited stack memory.</td>
</tr>
<tr>
<td>CHGGRN</td>
<td>Change grandularity (subsampling rate)</td>
</tr>
<tr>
<td>CHGRNG</td>
<td>Change range</td>
</tr>
<tr>
<td>CHGSTR</td>
<td>Change row &amp; column</td>
</tr>
<tr>
<td>CHGVIEW</td>
<td>Change viewpoint</td>
</tr>
<tr>
<td>CHGWID</td>
<td>Change row and column widths</td>
</tr>
<tr>
<td>CHKPHT</td>
<td>Used with some peak routing</td>
</tr>
<tr>
<td>CHINT</td>
<td>Convert character data to integer and change following blanks to zero.</td>
</tr>
<tr>
<td>CINV</td>
<td>Invert the complex number R + SQRT(-1)*I.</td>
</tr>
<tr>
<td>CIS</td>
<td>Convert integer to character return number of characters in an integer and encode to character data.</td>
</tr>
<tr>
<td>CLEAR</td>
<td>Clear the TEKTRONIX screen</td>
</tr>
<tr>
<td>CLOSE</td>
<td>Close file specified by JFN</td>
</tr>
<tr>
<td>CLRSCRN</td>
<td>Clears the screen</td>
</tr>
<tr>
<td>CLS</td>
<td>Closes file on unit CH</td>
</tr>
<tr>
<td>CMULT</td>
<td>Complex multiplication of two file one page at a time</td>
</tr>
<tr>
<td>CNCATR</td>
<td>Files are concatenated by reading them one row at a time and writing them one row at a time to the output file via other SUBR.</td>
</tr>
<tr>
<td>COPFIL</td>
<td>Copy a file into another.</td>
</tr>
<tr>
<td>COPSWP</td>
<td>Copy and swap data from arrays A,B,C,D.</td>
</tr>
<tr>
<td>COPY</td>
<td>Copy NUM words from array A to B.</td>
</tr>
<tr>
<td>CPUPLF</td>
<td>Copy GRID points from array A into array B.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CROP</td>
<td>CROP subarea out of file into another file</td>
</tr>
<tr>
<td>CROSS</td>
<td>Generate a correlogram</td>
</tr>
<tr>
<td>CRTHAF</td>
<td>Plot levels on CRT</td>
</tr>
<tr>
<td>CRTHIS</td>
<td>Print histogram on CRT</td>
</tr>
<tr>
<td>CURSOR</td>
<td>Display the cursor (cross-hair on 4014 terminal) on the screen and perform the screen operations.</td>
</tr>
<tr>
<td>DEBUG</td>
<td>Set DEBUG logical unit and level FLAG from user inputs.</td>
</tr>
<tr>
<td>DENO</td>
<td>Optionally open an output file for DEBUG.</td>
</tr>
<tr>
<td>DENOM</td>
<td>Sets definitions for string data</td>
</tr>
<tr>
<td>CURSOR</td>
<td>Display the cursor (cross-hair on 4014 terminal) on the screen and perform the screen operations.</td>
</tr>
<tr>
<td>DFT</td>
<td>Computes an AR model of the covariance function in R using the modified Yule-Walker equations.</td>
</tr>
<tr>
<td>D1DF1</td>
<td>Perform inverse 1-d FFT on a range of rows or columns.</td>
</tr>
<tr>
<td>DIVM</td>
<td>Divides one file by another (N1/N2)</td>
</tr>
<tr>
<td>DLYLIN</td>
<td>Places INSIG into delay line of length len ad returns current output of PTR &lt; 1, the array D is cleared and PTR is set to 1.</td>
</tr>
<tr>
<td>DOADD</td>
<td>Add arrays A, (B) and C, (D)</td>
</tr>
<tr>
<td>DOOD</td>
<td>Do slice plots</td>
</tr>
<tr>
<td>DOLOG</td>
<td>Take logarithm of data buffer</td>
</tr>
<tr>
<td>DOMAG</td>
<td>Take magnitude of array</td>
</tr>
<tr>
<td>DOMGSQ</td>
<td>Take MAG SQVRED of array</td>
</tr>
<tr>
<td>DOMLT</td>
<td>Multiply array by constant</td>
</tr>
<tr>
<td>DOSLIC</td>
<td>Do slice plots</td>
</tr>
<tr>
<td>DOSQRT</td>
<td>Take SQRT of array (if &lt; 0=0)</td>
</tr>
<tr>
<td>DOSUB</td>
<td>Subtract arrays</td>
</tr>
<tr>
<td>DOXFER</td>
<td>Transfer FER subarea for SUBR.CROP.</td>
</tr>
<tr>
<td>FFT</td>
<td>Computes complex FFT using the BUTTERFILES</td>
</tr>
<tr>
<td>FFT1</td>
<td>Perform one-Dim FFT (magnitude only)</td>
</tr>
<tr>
<td>FFM</td>
<td>Perform 1-D magnitude FFT on a range of rows or columns.</td>
</tr>
<tr>
<td>FILER</td>
<td>Reads file from disk</td>
</tr>
<tr>
<td>FILESTEM</td>
<td>Finds a max filter stem—\text{i.e., del the suffix.}</td>
</tr>
<tr>
<td>FILTFT</td>
<td>Apply filter using FFT’s</td>
</tr>
<tr>
<td>FLTOUT</td>
<td>Output floating point number to terminal</td>
</tr>
<tr>
<td>FLTTIN</td>
<td>Integer \rightarrow real</td>
</tr>
<tr>
<td>FRXFM</td>
<td>Reverse order FFT</td>
</tr>
<tr>
<td>FXTAIL</td>
<td>Fix tail of the file</td>
</tr>
<tr>
<td>GAUSS</td>
<td>Comp. uncorr. unit var., zero means, norm. rand var. by adding 12 random #'s uniformly in [0,1].</td>
</tr>
<tr>
<td>GETJFN</td>
<td>Get JFN for bookeeping table</td>
</tr>
<tr>
<td>GETPEC</td>
<td>Read file sequentially</td>
</tr>
<tr>
<td>GETROW</td>
<td>Get row of file</td>
</tr>
<tr>
<td>GETSTR</td>
<td>Prompts for string</td>
</tr>
<tr>
<td>GETVAL</td>
<td>Get value from terminal (real)</td>
</tr>
<tr>
<td>GETVAR</td>
<td>Calculate variance of array</td>
</tr>
<tr>
<td>GETVLVR</td>
<td>Get value from term. supply prompt</td>
</tr>
<tr>
<td>GNOISE</td>
<td>Generates zero mean, unit variance Gaussian noise by adding 12 uniform random variables.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>GTDM</strong></td>
<td>Prompts for distance, returns #, IDM, corresponding to choice.</td>
</tr>
<tr>
<td><strong>GTINSF</strong></td>
<td>Get file information and open file</td>
</tr>
<tr>
<td><strong>GTINT</strong></td>
<td>Gets integer from terminal</td>
</tr>
<tr>
<td><strong>GTMAX</strong></td>
<td>Get maximum value and location of file</td>
</tr>
<tr>
<td><strong>GTNAME</strong></td>
<td>Get name of file to open</td>
</tr>
<tr>
<td><strong>GTPAGE</strong></td>
<td>Get page (512 words) from file specified by JFN</td>
</tr>
<tr>
<td><strong>GTPART</strong></td>
<td>Get part of one file, and put into another</td>
</tr>
<tr>
<td><strong>GTPRMS</strong></td>
<td>Get file parameters</td>
</tr>
<tr>
<td><strong>GTPXL</strong></td>
<td>Gets pixels for plot support</td>
</tr>
<tr>
<td><strong>GTRANG</strong></td>
<td>Find Min. and Max. in a subarea grid</td>
</tr>
<tr>
<td><strong>GTREAL</strong></td>
<td>Get real part of file into another</td>
</tr>
<tr>
<td><strong>GTREL</strong></td>
<td>Get real*4 from terminal</td>
</tr>
<tr>
<td><strong>GTSTR</strong></td>
<td>Get string from terminal</td>
</tr>
<tr>
<td><strong>GTTAIL</strong></td>
<td>Gets tail from a file</td>
</tr>
<tr>
<td><strong>GTTYCH</strong></td>
<td>Gets char. from terminal</td>
</tr>
<tr>
<td><strong>GTYES(funct)</strong></td>
<td>Yes or No from user</td>
</tr>
<tr>
<td><strong>HAFTON</strong></td>
<td>For halton plots on PRINTOX</td>
</tr>
<tr>
<td><strong>HANFRQ</strong></td>
<td>Hanning weighting by convolution</td>
</tr>
<tr>
<td><strong>HANTIM</strong></td>
<td>Hanning weighting by multiplication of time</td>
</tr>
<tr>
<td><strong>HARD</strong></td>
<td>TEKTRONIX DEF.</td>
</tr>
<tr>
<td><strong>HISTO</strong></td>
<td>Computes histogram of a file</td>
</tr>
<tr>
<td><strong>HISTOG</strong></td>
<td>Computes histogram of an array</td>
</tr>
<tr>
<td><strong>IDFT</strong></td>
<td>Perform inverse 1-D FFT on array</td>
</tr>
<tr>
<td><strong>IGETYN</strong></td>
<td>Get Yes/No answer from terminal</td>
</tr>
<tr>
<td><strong>IGTVLR(funct)</strong></td>
<td>Get input value (integer) from terminal</td>
</tr>
<tr>
<td><strong>INSERT</strong></td>
<td>Insert part of one array into part of another</td>
</tr>
<tr>
<td><strong>INSRT</strong></td>
<td>Insert subarray from [A,B] into [C,D]</td>
</tr>
<tr>
<td><strong>INTRPG</strong></td>
<td>Do two-dimensional interpolation to get the value at position (row, col.)</td>
</tr>
<tr>
<td><strong>INTFLT</strong></td>
<td>Put integer part of array A into B</td>
</tr>
<tr>
<td><strong>INTOUT</strong></td>
<td>Output integer variable to terminal</td>
</tr>
<tr>
<td><strong>INTSCL</strong></td>
<td>Scale real data to range from 0 - to max.</td>
</tr>
<tr>
<td><strong>INTTIN</strong></td>
<td>Get integer variable from terminal</td>
</tr>
<tr>
<td><strong>IOSDEF</strong></td>
<td>Sets VAX DEF. (err #’s) VAX UTIL.</td>
</tr>
<tr>
<td><strong>ISOPLT</strong></td>
<td>For plotting</td>
</tr>
<tr>
<td><strong>JJASC</strong></td>
<td>Process a user ASCII sequence.</td>
</tr>
<tr>
<td><strong>JJBLANK</strong></td>
<td>Blank fill a buffer</td>
</tr>
<tr>
<td><strong>JCHN</strong></td>
<td>To channel strings</td>
</tr>
<tr>
<td><strong>JJCJPI</strong></td>
<td>To get job process information from DEC. (function)</td>
</tr>
<tr>
<td><strong>JJCMMD</strong></td>
<td>Returns the input command to the user</td>
</tr>
<tr>
<td><strong>JJCTC(funct)</strong></td>
<td>Resets -C</td>
</tr>
<tr>
<td><strong>JJCTCSET</strong></td>
<td>Actual sets -C</td>
</tr>
<tr>
<td><strong>JJCTO</strong></td>
<td>Used to reset a 0, \ie, enable printing</td>
</tr>
<tr>
<td><strong>JJDAY</strong></td>
<td>Calculate the day of the week</td>
</tr>
<tr>
<td><strong>JJDCL</strong></td>
<td>Used to issue a command to DCL</td>
</tr>
<tr>
<td><strong>JJFST</strong></td>
<td>Returns column of first printing characters of string.</td>
</tr>
<tr>
<td><strong>JJHALF</strong></td>
<td>Halftone plot</td>
</tr>
<tr>
<td><strong>JJINBY</strong></td>
<td>Read in a byte from a channel</td>
</tr>
<tr>
<td><strong>JJINES</strong></td>
<td>Process get arrow keys</td>
</tr>
</tbody>
</table>
JJLCV  
Make a list entry of a character value
JJLEN  
Returns CHAR LEN of STR
JJLOW  
Convert string to lower case
JJMAR  
Macro entry point for terminal formatting
JJMESS  
Print on unit 6 a system message
JMJINS  
Compute minutes of day of the week
JJUBY  
Write out a byte sequence
JJOUCH  
Output a character array
JJSCR  
Support screen functions
JJTIME  
Reads time since last call
JJUCMD  
Process a UNIX type command line
JJUPP  
Converts the string into uppercase
JJWAIT  
Will suspend processing for $X$ seconds
JSAIO  
File I/O
LENSTR(funct)  
Returns length of string
LFCR  
Carriage return/linefeed to terminal
LIMIT  
Output find minimum and maximum of an array
LINSUT  
Will output line to screen
LOCMIN  
Find local minimum in window about target location and interpolate to find peak position
LSTPAG  
Get last page number from file
MAGNTD  
Takes HAG of file
MAGSOD  
Takes MAG SQD of file
MLCONJ  
Multiply file N1 by N2
MOVE  
Move cursor
MULCOM  
Multiply complex files one page at a time
MULCON  
Multiply array by constant
MULCX  
Compute product of complex data
MULFIL  
Multiply two arrays
MULMAG  
Compute product and magnitude on complex data
MULSCL  
Multiply file by constant
MULT  
Multiply page (512 words) of real by imaginary
MXMULT  
Multiply real file by complex file one page at a time
NLNFOR  
Used for FORTRAN namelist read
NORCRS  
Normalized cross-correlation of arrays.
NORMAL  
Used to normalize input
NORMILZ  
Normalized maximum to less than 1
OPERAT  
Signal processing driver program
OPMULT  
Multiply [A,B] and [C,d} and put result into A,B
OPN  
Opens file w/NAME=NAM
OPSUB  
Subtract [C,D] from [A,B] and place into A,B
OPSUBR  
Subtract [A,B] from [C,D] and place the result into A,B
OPY  
Support of the copy subroutine
QTSTR  
Prints string on current line
OUTLIN  
Put out single line of greyscale on TEKTRONIX
PAD  
PAD array w/zeros
PGTXT  
Put character string A into B
PLTTR  
Plot utility
PLTSTF  
Plot utility
PLTVX  
Plot 1-D array on VAX
POLFAC Find Poly. roots
POLMLT Multiply two polynomials $P_1$ and $P_2$
NPOW2(funct) Find number of powers of 2 in N select $2^N$ POWZ=N
OPADD Add [A,B] to [C,D] and put into [A,B]
OPDIV Divide [A,B] by [C,D] and put into [A,B]
OPDIVR Divide [C,D] by [A,B] and put into [A,B]
PRINT Prints a character histogram
PRSPCT Create perspective type plot
PRTCOM Prints characteristics of data and file structure
PRTERR Prints runtime error codes
PSORTM Passive sort for matrices
PTPAGE Places (512 words) page into file specified by JFN
PTPXL Places tail on existing file
PUTPEL Write a disk file sequence one piece at a time
RBFOUT TEKTRONIX def. send char. buff to TEKTRONIX
RCLEAR Clear screen of TEKTRONIX
RCOPY Copy NUM of words from real array A to B
READ Read file into arrays A, B and return name and complex flag
READIM Read function
READIN Read file into arrays given JFN, length, and complex flag
READRL Read function
RECIPI Set array B to the reciprocal of array A
REDFIL Read array from file--get name if not given
REDMAN Define an array manually via interactive input of data
REDMAS Read array from file-get name
RELINT Real to integer converter
REMVMN Remove mean from array
RESET Reset file tail def.
RHIARD Get hard copy on TEKTRONIX
RHFTON Put out halftone (grey-scale) pixel on TEK 4014
RITFIL Write array to file
RMNVAR Remove mean from file and calculate variance
RMOVE Remove pixel address on TEK
RMULT Multiply 2 files one page at a time
RNAM REN file A to file B
RREC Reads fr-sr 512 bytes rec. from ch to buf, starting at rec sr finishing at fr
RRECXY Reads starting sr finishing fr
RSEND Send plot buffer to TEK
RSET Init. TEK screen
RVEC Write vector on TEK
RWAIT Wait for TEK to finish plotting
RWCHR Write char. on TEK
RWTHRU Close plotting buffer of TEK
S2I Converts string to integer
S2R Converts string to real
SCALE Scale grey-scale plot for TEK
SEND Flush display buffer
SET Init. TEK screen

35
SETCON  Set array to constant value
SETPD   Set-up file bookeeping array
SETUP   Set-up function
SPFIL   Fill char str. w/blanks
SQRTMX  Place SQRT of file into another
SQUARE  Square array
START   Calc. statistics for a file
STATAL  Stats. for array
STIME   Find 2 indices for array
STRNGI  Get CHAR STR from TERM.
STRNGO  Output CHAR STR to TERM.
STRTND  Get start and end indices
SUBAR   Get subarea stats.
SUBMX   Subtract file N2 from file N1
SUM     Summation of array
SUM2    Sum of squares of array
SUM24   Sum of squares and 4th power of array
TBCOPY  Copy string from array into char.
TEK     Driver for TEK util.
TKCLIP  Clip and plot vector on TEK
TKFINSH Finish a plot on TEK
TKGETS  Receive scaling parameters for TEK
TKGRID  Draw grid lines
TKLABL  Label a tick mark
TKMARK  Marks point on a curve
TKNAME  Mark name of a curve
TKPARM  Set or show display parameters
TKPRINT Plot a line segment
TKSCAL  Init TEK
TKTICK  Draw a tick on a curve
TGOLEND Read files based upon a mean selection
LSTCHG  Listing function
TOREAD  Read function
TOREAD1 Read function
TOREAD2 Reads files
TOREADB Read function
TORED   Read function
TORITE  Writes files
TORITE2 Write function
TORITEB Write function
TRAK    Tracking information for identification of arrays A-E
TRAKFUNCT Interpret abbreviations of processing
TRAKLOG Maintaining log of current operations for tracking system
TRANS   Used by GTPART
TRANSPO  Perform transpose on array
TXCOPY  Copy character string from text to array
TYPACT  Pack numbers function
TYPCHG  Change case of alpha characters
TYPROW  Type a row

36
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRTFIL</td>
<td>Write array to file and attach tail</td>
</tr>
<tr>
<td>MULMAG</td>
<td>Comp product and MAG of complex data</td>
</tr>
<tr>
<td>SSQROW</td>
<td>Sum of squares for each row</td>
</tr>
<tr>
<td>SSQCOL</td>
<td>Sum of squares for each column</td>
</tr>
<tr>
<td>ZMNROW</td>
<td>Remove mean from each row</td>
</tr>
<tr>
<td>ZMNCOL</td>
<td>Remove mean from each column</td>
</tr>
<tr>
<td>ZMNR</td>
<td>Remove mean</td>
</tr>
<tr>
<td>NORMAL</td>
<td>Normalize function</td>
</tr>
<tr>
<td>MLLCX</td>
<td>Product of complex data</td>
</tr>
<tr>
<td>PEPROW</td>
<td>Sum of squares for each row</td>
</tr>
<tr>
<td>REPCol</td>
<td>Sum of squares for each column</td>
</tr>
<tr>
<td>DODEC</td>
<td>Decimal function</td>
</tr>
<tr>
<td>VEC</td>
<td>Draw vector on TEK</td>
</tr>
<tr>
<td>WAIT</td>
<td>Wait function</td>
</tr>
<tr>
<td>WATERF</td>
<td>Perform waterfall plots</td>
</tr>
<tr>
<td>WRTCHR</td>
<td>Write chr. to TEK</td>
</tr>
<tr>
<td>WRTHRU</td>
<td>Write function</td>
</tr>
<tr>
<td>WRTOUT</td>
<td>Write array into file</td>
</tr>
<tr>
<td>XYPLOT</td>
<td>Do $x$-$y$ plots allowing multiple curves</td>
</tr>
<tr>
<td>ZERO</td>
<td>Zero NUM words of array 4</td>
</tr>
</tbody>
</table>
Appendix B: User's Guide

B.1 Description

Before using the multipath signal processing software, it is important to already be familiar with the VAX-VMS system, and the DEC command language (DCL) operations level. The DCL prompt is "$", this is the first level accessed by the user. At the DCL level, the user can access a FORTRAN main driver by prompts for possible operations. Based on the users' answers to prompts, FORTRAN and MACRO subroutines are called to process the data. The FORTRAN subroutines compose the majority of the technical abilities available to the user. Included in the subroutine library are all the appropriate device drivers for both graphics and interactive communication.

B.2 Utility Operations

To begin using this signal processing environment, simply enter RUN MAIN at the DCL operating level. The software will then respond by prompting the user for verification of the debug level. The minimum value is 0 and the maximum value is 100. With the default for this selection being 1. The user is just required to hit return to enter the default selection. Example:

```
Debug level MN= 0  MX= 100
DF= 1 ?
Value = 1
```

Once a return is entered, the debug level is on, this means a log file of the session is created. The log file created has two options for the user: (i) to be sent to the screen, (ii) to be to an output file to be viewed or printed later. The file created is named "MAIN.DBG", and the second prompt will ask the user where to place the file. The default value is to have the information placed into the current directory, and is obtained again by entering a RETURN. Throughout the software system there are default values, these default values all can be entered by entering a RETURN. The user can select any one of the options.

Signal Processing Functions

The Signal Processing option has two levels; the first, provides utility options, and the second, is a signal processing environment. The following is a list of operations: for the first level.

```
OPERATION(HELP) ??

Legal operations ..
Operat-arrays .. description

? - .. HELP
$ DCL COMMAND .. WILL EXECUTE DCL COMMAND
EXIT - .. EXIT FROM PROGRAM
```

38
QUE .. QUERY THE ARRAYS

**************** DATA MANIPULATION OPERATIONS ****************
CLEAR .. CLEAR THE SCREEN
COPY .. COPY AND SWAP ARRAYS
CROP .. CROP ARRAYS A & B INTO C AND D
DIM -A,B .. SET THE SUBAREA DIMENSIONS
           SET Min,Max,Ave,Sdev.
FXTAIL- .. FIX TAIL ON FILE
IMBED-A .. DEFINE NEW AREA AND IMBED THE PREVIOUS
           AREA AS A SUBAREA AT UPPER LEFT CORNER
LS - .. LIST AND CHANGE ARRAY CONTENTS
MANUAL-A .. MANUALLY ADD,DELETE,CHANGE OR LIST ARRAY
MIN .. SET Min, Max USING ACTUAL DATA
PARAM- A,B,C,D,E SET OR CHANGE ARRAY PARAMETERS
SDEV -A .. SET Min, Max USING STANDARD DEVIATION
SET - .. SET Min, Max INTERACTIVELY
SHIFT - A,B .. CIRCULAR SHIFT
WRITE .. WRITE ARRAYS A & B TO A FILE
ZERO-A,B,C,D,E .. FILL ARRAYS WITH ZERO

***************** STATISTICAL OPERATIONS *******************
HIST -A .. COMPUTE A HISTOGRAM
LI - .. LIST SIZE AND STATISTICS
STAT -A .. CALCULATE STATISTICS
SNR - .. SIGNAL TO NOISE RATIO CALCULATION

****************** GRAPHICS ***********************
LEVEL -A .. LEVEL Plot on the CRT
PLOT - .. X,Y PLOT WITH INTERACTIVE INPUT OF DATA
POST - .. HALFTONE PLOT ON THE APPLEWRITER
PTX -A .. HALFTONE PLOT ON THE TALARIS LASER PRINTER
SLICE -A .. PLOT SLICE (CRT,PTX,TEK)
TEK -A .. HALFTONE PLOT ON THE TEKTRONIX
WATER-A,B,C,D,E .. WATERFALL PLOT(CRT,PTX,TEK)

****************** SIGNAL PROCESSING OPERATIONS ***************
SP -A,B,C,D .. PERFORM SIGNAL PROCESSING OPERATIONS

****************** ARITHMETIC OPERATIONS **********************
CALCUL-A,B,C,D .. CALCULATOR (RPN)

The next prompt is for loading a data file, the default answer is yes. The system will prompt for the file name, at this point the user can enter the DCL level to see what files
exist in the directory for selection. To load a file, simply type the file name. The file is read by the software and statistics are displayed of the file structure and the data.

**Example of File Statistics**

```
OPERATION(HELP) stat

SUBAREA ONLY (NO) ?

Value = N

1 THRU 32 =NROW
1 THRU 64 =NCOL
5 =NRNP 6 =NCNP 0 =ICX
-60.000 =MIN.VAL ( 7, 52)
78.000 =MAX.VAL ( 31, 31)
1.8960 =AVE.VAL 18.181 =SIG.VAL

Subarea.param

1 THRU 32 =ROW
1 THRU 64 =COL
0.00000E+00 =MIN.SUB ( 0, 0)
0.00000E+00 =MAX.SUB ( 0, 0)
0.00000E+00 =AVE.SUB 0.00000E+00 =SIG.SUB

MIN/MAX IN USE
-60.000 =MIN.USE 78.000 =MAX.USE
```

The main menu and entrance to the data processing levels appears next. The user can perform data manipulation or signal processing operations.

**Main Menu**

After displaying the file statistics, the main menu is then displayed with the following options:

1. Signal Processing Functions
2. Classification Functions
3. Read New File
4. Database Management
5. Recognition
6. Exit

The programs within FORTRAN database use a file format that is well suited for data manipulation and operations. The specifics on each data file are described by parameters in its "tail" or physical header record. These descriptions can be modified for processing which will change the attributes of the data file. The files themselves are random access format with 256 four-byte words per record. The tail contains all the file attributes, such as: number of rows and columns, a complex or real flag, and a description of the file itself. Since the tail is a complete record of 256 words, it is being further utilized in the VAX software as
storage of other parameters or information for programs specifically designed for a particular
type of data. This information is stored in a common, thus allowing the user to specify the
format of the common parameters. The tail also has the option of being manipulated by the
user.

The data is organized in the file by rows, that is, the first column entries comprise the
first row and the next column entries comprise the second row, etc. Accordingly, the array
is dimensioned (NCOL,NROW) (NROW represents number of rows, and NCOL represents
number of columns).

The first category of operations for file manipulation are as follows:

Legal operations ...
Operat-arrays .. description

? - .. HELP
$ DCL COMMAND .. WILL EXECUTE DCL COMMAND
EXIT - .. EXIT FROM PROGRAM
QUE .. QUERRY THE ARRAYS

ZERO-A,B,C,D,E.. FILL ARRAYS WITH ZERO
MANUAL-A .. MANUALLY ADD,DELETE,CHANGE OR LIST ARRAY
FXTAIL- .. FIX TAIL ON FILE
PARAM- A,B,C,D,E SET OR CHANGE ARRAY PARAMETERS
COPY .. COPY AND SWAP ARRAYS

File attributes - manipulation operations
Array Creation - specify array attributes and values.

The next operations are as follows:

LS - .. LIST AND CHANGE ARRAY CONTENTS
STAT -A .. CALCULATE STATISTICS
HIST -A .. COMPUTE A HISTOGRAM
DIM -A,B .. SET THE SUBAREA DIMENSIONS
  SET Min,Max,Ave,Sdev.

Array/subarray Statistics - actual and those in use.

and finally, the following operations:

SDEV -A SET Min, Max USING STANDARD DEVIATION
SET - SET Min, Max INTERACTIVELY
MIN .. SET Min, Max USING ACTUAL DATA
Array Display – row/col., 3-D grayscale, histogram

Array Operations – arithmetic, and signal processing functions

All options require only the first two unique characters for execution. All file I/O (read/write operations) are defined when entering and leaving the main menu which is outside the operations and signal processing menu.

All operations are done using five arrays A, B, C, D, E. A is the primary array. The statistics and plotting options in the operations menu are only done for the data contained in array A. Array B is used for the imaginary part of array when the data is complex. Arrays C, D and E are used to hold the real and imaginary parts of a secondary array (the secondary array is used for multiple array operations such as addition, subtraction, etc.). There are copy and swap operations for the primary and secondary arrays to aid the user with shifting data (i.e., array C may be swapped with array A so that statistics may be calculated with it). Data arrays [A, (B)] and [C, (D)] must have the same dimensions. The various options specify the arrays used for each specific option.

Three types of statistics are kept for array A only. Statistics of the entire array, a user-defined subarray, and the minimum and maximum currently being used. The min/max currently in use are the last statistics calculated (menu option (ST)at, for array or subarray A) or the values set by options (SE)t, (MI)n, and (ST)dev. The plotting operations always use this min/max value before entering plotting operations and whenever array A is redefined (i.e., by signal processing operations) the (ST)at option must be used to update both the array and subarray statistics if they are to be used in any plotting operation.

Array A can be displayed in the following ways:

- (SL)ice — row or column 2-D
- (Wa)terfall — waterfall of rows or columns
- (LE)vel — 3D level plot using characters
- (PT)x — 3D gray scale on the printronix

The 3D plots use the array/subarray statistics provided, while the row/column plots allow for row/column specific statistics. Any of the vector line-drawn plots may be plotted using the output devices defined in the operations prompt.

Operations on the arrays are done using option (SP). (S)ignal (P)rocessing will process the array or subarray using specific signal processing functions selected from a menu and results will be placed into [A,(B)]. All of the functions will allow for real or complex data.

Arrays (and files) may be created using several of the array attributes options. (ZE)ro sets the selected array values to zero and for array A, will allow for specification of the number of rows and columns. (PA)ram will allow the user to set the dimension of the arrays while not changing any array values. This is useful for time data where FFT’s may be needed on vectored segments. If the data had 4096 samples (NCOL=4096, NROW=1) the user could set NCOL=128 and NROW=32 and perform a FFT (in Signal Processing menu) on the 32 rows giving a time series of spectra. The (LS)tchg option can be used to list or exchange array/subarray values and (CA)lculate is a Reverse Polish Notation calculator which can access and enter array values.

For future planning and updates, options are easily added to the operations or Signal Processing menu in subroutine form so any existing applicable subroutines would broaden its scope. All future additions would be designed to eventually develop an “automatic” system for identification.
?-HELP-

The question mark is the symbol used to evoke the help routines. The actual word (HE)lp will also evoke the help option. Currently the help option will just supply the user with a list of operations available to the user within the current menu.

Example of help output:

OPERATION(HELP) ??

Legal operations ..
Operat-arrays .. description

? - .. HELP
$ DCL COMMAND .. WILL EXECUTE DCL COMMAND
EXIT - .. EXIT FROM PROGRAM
QUE .. QUERRY THE ARRAYS

*************** DATA MANIPULATION OPERATIONS ***************

CLEAR .. CLEAR THE SCREEN
COPY .. COPY AND SWAP ARRAYS
CROP .. CROP ARRAYS A & B INTO C AND D
DIM -A,B .. SET THE SUBAREA DIMENSIONS
        SET Min,Max,Ave,Sdev.
FXTAIL- .. FIX TAIL ON FILE
IMBED-A .. DEFINE NEW AREA AND IMBED THE PREVIOUS
        AREA AS A SUBAREA AT UPPER LEFT CORNER
LS - .. LIST AND CHANGE ARRAY CONTENTS
MANUAL-A .. MANUALLY ADD,DELETE,CHANGE OR LIST ARRAY
MIN .. SET Min, Max USING ACTUAL DATA
PARAM- A,B,C,D,E SET OR CHANGE ARRAY PARAMETERS
SDEV -A .. SET Min, Max USING STANDARD DEVIATION
SET - .. SET Min, Max INTERACTIVELY
SHIFT - A,B .. CIRCULAR SHIFT
WRITE WRITE ARRAYS A & B TO A FILE
ZERO-A,B,C,D,E .. FILL ARRAYS WITH ZERO

****************** STATISTICAL OPERATIONS ******************

HIST -A .. COMPUTE A HISTOGRAM
LI - .. LIST SIZE AND STATISTICS
STAT -A .. CALCULATE STATISTICS
SNR - .. SIGNAL TO NOISE RATIO CALCULATION

*************** GRAPHICS ******************
LEVEL -A .. LEVEL Plot on the CRT
PLOT - .. X,Y PLOT WITH INTERACTIVE INPUT OF DATA
POST - .. HALFTONE PLOT ON THE APPLEWRITER
PTX -A .. HALFTONE PLOT ON THE TALARIS LASER PRINTER
SLICE -A .. PLOT SLICE (CRT,PTX,TEK)
TEK -A .. HALFTONE PLOT ON THE TEKTRONIX
WATER-A,B,C,D,E. WATERFALL PLOT(CRT,PTX,TEK)

*************** SIGNAL PROCESSING OPERATIONS ***************

SP -A,B,C,D .. PERFORM SIGNAL PROCESSING OPERATIONS

*************** ARITHMETIC OPERATIONS ***************

CALCUL-A,B,C,D.. CALCULATOR (RPN)

This operation environment can also lead the user into a signal processing laboratory environment. The following example shows the signal processing functions available:

SIG. PROCESSING OPERATION (HELP=?) ??

RESULTS WILL BE PUT INTO [A,(B)]
SYN .. SYNTHETIC DATA OPTION
AR .. AR SPECTRAL FEATURES
AT .. ATACH PHASE TO [A,B] , COMPUTE [A,B]*EXP(PHASE)
GB .. GABOR COEFFICIENTS
RMA .. ARMA SPECTRAL FEATURES
RN .. GENERATE RANDOM NOISE
RS .. GENERATE SYMMETRIC RANDOM IMAGE
NC .. NOISE CANCELLATION OPTION
NG...... ENERGY FUNCTION

EXIT .. EXIT OPERAT
$ ...... ACCESS TO DCL
STATS .. GET STATISTICS

SROOT .. SQUARE ROOT
MAG .. MAGNITUDE
MSQD .. MAGNITUDE SquARED
SCALE .. SCALE AND OFFSET (K1*A + K2)
LOG .. LOGRITHM BASE 10 OR e
PHASE .. PHASE
XPON .. EXPONENTIAL(10 OR e) (EXP(K1*RE(A)+K2)
SQUARE .. SQUARE OF DATA
ZCR..... ZERO CROSSING RATE

44
+ .. [A,(B)] + [C,(D)]
- .. [A,(B)] [-C,(D)] OR REVERSE
* .. [A,(B)] * [C,(-D)]
/ .. [A,(B)] / [C,(D)] OR REVERSE
  QUI...... QUERY WHAT IS STORED IN ARRAYS
1DFFT .. 1-DIMENSIONAL FFT
FFTM .. FFT MAGNITUDE
I1DFFT.. INVERSE 1-DIMENSIONAL FFT
2DFFT .. PERFORM 2D FFT
I2DFFT.. PERFORM INVERSE 2D FFT
HFREQ .. HANNING WEIGHTING BY CONVOLUTION (FREQ)
HTIME .. HANNING WEIGHTING BY MULTIPLICATION (TIME)
PPIC .. PICK THE PEAKS AND THE CORRESPONDING AMPLITUDES
PEAK .. PICK THE HIGHEST PEAKS(MAXIMA)
TDA .. SYNTHETIC TRANSIENT DATA

The “Operation (help=?)?” seen on the previous example is the user prompt for the operations menu.

$DCL Command –

At the operations menu selection, spawned DEC Command Level operations can be executed. At the above-defined user option prompt a “$” followed by a DEC command will spawn off a DEC level execution.

Example:

Operation (help=?)? $ DIR

This “$ DIR” will then prompt the VAX to supply the user with a content of the current directory without leaving the execution of the current work.

Zero –

(ZE)ro will allow the user to zero out any of the four arrays. If data does exist in the array, zero will prompt the user for number of rows and columns the user wishes to use for defining the area to be filled with zero's. The default values are the maximum numbers or rows and columns for that array selected. All default parameters can be entered in by having the user just enter a RETURN. (ZE)ro also has a complex data flag where the default is No.

Example:

OPERATION(HELP) ?zero

ZERO ARRAY (A,B,C,D,E,*) ?a

This allows the user to select an array to be used in the zero operation.
After specifying the complex data flag, the section of data selected within the desired array is made into an array of zeros.

**MANual**

The (MA)nual operation will allow the user to manually edit row dimension, column dimensions and values of the array. This supplies the user with the ability to manually enter new data. The (MA)nual data editing is as follows:

```
OPERATION(HELP) ?ma

NROW =  32  NCOL =  64
Options Add, Change, Delete, List, Exit ?c

Start row to change MN= 1  MX= 32
DF= 0 ? 32

Value =  32
End row to change MN= 32  MX= 32
DF= 32 ?

Value =  32
Start col to change MN= 1  MX= 64
DF= 0 ? 1

Value =  1
End col to change MN= 1  MX= 64
DF= 1 ? 64

Value =  64
( 32, 1) 0.00000000E+00 ?

Value =  0.00000000E+00
( 32, 2) 0.00000000E+00 ?

46```
The defaults for NROW and NCOL are the maximum parameters.

The (FX)tail option is for user modification into the actual tail to the data file. The editing done will specify the type of data and the window in which data is to be accessed.

**Example:**

```
OPERATION(HELP) ?fx

($ DCL COMMAND OPTION) READ FROM : a:a01ti1.raw
CURRENT TAIL IS:

NUM ROWS= 32, NUM COLS= 64
PK: 0  CX: 0
STRING: bugs

CHANGE? ( NO) ?yes

Value = YES
# ROWS  32.00000  ? 64

Value = 64.00000
#COLS 64.00000  ? 32

Value = 32.00000
PACKED FLAG 0.00000000E+00 ?

Value = 0.00000000E+00
COMPLEX FLAG 0.00000000E+00 ?

Value = 0.00000000E+00

MESSAGE STRING: bugs

CHANGE STRING ( NO)?
```

47
CURRENT TAIL IS:

NUM ROWS= 64, NUM COLS= 32
PK: 0 CX: 0
STRING: bugs

CHANGE? ( NO) ?

Value = N

With (FX)tail the changes are also permanently stored in the header record and will be kept even after the session is completed. (FX)tail only modifies the tail on array A. (PA)ram

Like (FX)tail, (PA)ram will modify the tail for all four arrays; A, B, C, D, E. However, the new values placed into the tail, are only used in the session which they are set, and not permanently attached to the data file. For permanent changes refer to (FX)tail. (PA)ram will also operate on the current array in use.

Example:

```
OPERATION(HELP) ?pa

# ROWS  32.00000  
Value = 32.00000
# COLS  64.00000  
Value = 64.00000
COMPLEX FLAG  0.00000000E+00  
Value = 0.00000000E+00
```

(CO)py

The (CO)py option had a variety of copy and swap options. To obtain a help listing of the available options enter (-1) at the copy's selection prompt. Example:
OPERATION(HELP) ?co

COPY OPTION (-1 = ?) 0.00000000E+00 ? -1

Value = -1.000000

COPY AND SWAP OPTIONS

0 - EXIT
SWAP  1 - A > < B
2 - C > < D
3 - A,(B) > < C,(D)

COPY  4 - A > B
5 - B > A
6 - A,(B) > C,(D)
7 - C,(D) > A,(B)

8 - SUB[A,(B)] > (NEW)SUB[C,(D)]
9 - SUB[C,(D)] > (NEW)SUB[A,(B)]

COPY OPTION (-1 = ?) 0.00000000E+00 ?

Value = 0.00000000E+00

(ST)atistics

(ST)atistics is first seen by the user when a file is selected. The (ST)atistics will refer to the original data file first read. The first question in the operations menu is for the use of the entire subarea or full array of data. Once the area is chosen, statistics are calculated and displayed on the terminal.

Example:

OPERATION(HELP) ?stat

SUBAREA ONLY ( NO ) ?

Value = N

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>THRU</td>
</tr>
<tr>
<td>1</td>
<td>THRU</td>
</tr>
<tr>
<td>5</td>
<td>=NRNP</td>
</tr>
<tr>
<td>6</td>
<td>=NCNP</td>
</tr>
<tr>
<td>0</td>
<td>=ICX</td>
</tr>
<tr>
<td>0.000000E+00</td>
<td>=MIN.VAL ( 1, 1)</td>
</tr>
<tr>
<td>0.000000E+00</td>
<td>=MAX.VAL ( 1, 1)</td>
</tr>
<tr>
<td>0.000000E+00</td>
<td>=AVE.VAL 0.000000E+00 =SIG.VAL</td>
</tr>
</tbody>
</table>

49
Subarea.param

1 THRU 32 =RDW
1 THRU 64 =CDL
0.00000E+00 =MIN.SUB (0, 0)
0.00000E+00 =MAX.SUB (0, 0)
0.00000E+00 =AVE.SUB 0.00000E+00 =SIG.SUB
MIN/MAX IN USE
0.00000E+00 =MIN.USE 0.00000E+00 =MAX.USE

(LI)st

(LI)st will list size and statistics of the array.

Example:

```
OPERATION(HELP) ?li
```

```
1 THRU 32 =NROW
1 THRU 64 =NCOL
5 =NRNP 6 =NCNP 0 =ICX
0.00000E+00 =MIN.VAL (1, 1)
0.00000E+00 =MAX.VAL (1, 1)
0.00000E+00 =AVE.VAL 0.00000E+00 =SIG.VAL
```

Subarea.param

```
1 THRU 32 =ROW
1 THRU 64 =CDL
0.00000E+00 =MIN.SUB (0, 0)
0.00000E+00 =MAX.SUB (0, 0)
0.00000E+00 =AVE.SUB 0.00000E+00 =SIG.SUB
MIN/MAX IN USE
0.00000E+00 =MIN.USE 0.00000E+00 =MAX.USE
```

B.3 The Signal Processing Functions

To invoke the signal processing option, enter SP (= Signal Processing) while in the operations menu:

```
Operation (help?) ? SP
```

Upon entering the SP option, the code will respond with the following inquiry

```
Operate on subarea only (NO)?
```

```
Value = N
```

At this point, the user has the option of entering “No” or “Yes” for performing the signal processing operations on the entire data (in program array A) or on a subarea defined by
the command DIM. The default value entered by pressing the RETURN key is N(No), that is the entire data is processed.

Example:

SIG. PROCESSING OPERATION (HELP=?) ??

RESULTS WILL BE PUT INTO [A,(B)]
SYN .. SYNTHETIC DATA OPTION
AR .. AR SPECTRAL FEATURES
AT .. ATACH PHASE TO [A,B] , COMPUTE [A,B]*EXP(PHASE)
GB .. GABOR COEFFICIENTS
RMA .. ARMA SPECTRAL FEATURES
RN .. GENERATE RANDOM NOISE
RS .. GENERATE SYMMETRIC RANDOM IMAGE
NC .. NOISE CANCELLATION OPTION
NG...... ENERGY FUNCTION

EXIT .. EXIT OPERAT
$ ....... ACCESS TO DCL
STATS .. GET STATISTICS

SROOT .. SQUARE ROOT
MAG .. MAGNITUDE
MSQD .. MAGNITUDE SQUARED
SCALE .. SCALE AND OFFSET (K1*A + K2)
LOG .. LOGRITHM BASE 10 OR e
PHASE .. PHASE
XPON .. EXPONENTIAL(10 OR e) (EXP(K1*RE(A)+K2)
SQUARE.. SQUARE OF DATA
ZCR..... ZERO CROSSING RATE
+ .. [A,(B)] + [C,(D)]
- .. [A,(B)] -[C,(D)] OR REVERSE
* .. [A,(B)] * [C,(D)]
/ .. [A,(B)] / [C,(D)] OR REVERSE
QU...... QUERY WHAT IS STORED IN ARRAYS
1DFFT .. 1-DIMENSIONAL FFT
FTM .. FFT MAGNITUDE
11DFFT.. INVERSE 1-DIMENSIONAL FFT
2DFFT .. PERFORM 2D FFT
12DFFT.. PERFORM INVERSE 2D FFT
HFREQ .. HANNING WEIGHTING BY CONVOLUTION (FREQ)
HTIME .. HANNING WEIGHTING BY MULTIPLICATION (TIME)
PPIC .. PICK THE PEAKS AND THE CORRESPONDING AMPLITUDES
PEAK .. PICK THE HIGHEST PEAKS(MAXIMA)
TDA .. SYNTHETIC TRANSIENT DATA
The functions incorporated within the data base management and processing software are
designed for processing of underwater signals and are mostly of signal processing type. These
functions allow for a range of parametric and nonparametric feature extraction. Among the
features are energy and zero-crossing functions, FFT spectrum, Autoregressive (AR) and
Autoregressive Moving Average (ARMA) spectral features. Each function has to be provided
with a set of proper inputs that are specific to the particular signal processing operation.
The code prompts the user for each input and provides a short description for it. The default
as well as the minimum and maximum allowable values of input are given for most of the
functions and are entered by pressing the RETURN key. Signal processing functions operate
on the data stored in the program arrays A and B depending on whether the data is real
or complex. If data is real, operation is performed on the contents of array A only, if it is
complex (such as a complex DFT array) then both arrays A and B are operated on. The
results of the signal processing operations may be written back into the arrays A and B and
also into the file MAIN.DBG. Thus the data may be written over in the process. If the data
has to be used for later processing, the copy option must be exercised to save the contents
of array A (or A and B) in the array C (or C and D).

The STAT option produces the following statistics about the data in array A
1. Number of rows and columns in the array and in the subarea (subarea dimensions) and
logarithm in base 2 of these numbers.
2. Minimum and maximum values and where they occur for the entire array and the sub-
area.
3. Average values and standard deviations for the entire array and the subarea.

Following is an example of typical output using the STAT option. There are several
categories of operations. The first involves getting information about the data (STATS
command).

SIG. PROCESSING OPERATION (HELP=?) ?stat

1 THRU 32 =NROW
1 THRU 64 =NCOL
5 =NRNP 6 =NCNP 0 =ICX
0.00000E+00 =MIN.VAL ( 1, 1)
0.00000E+00 =MAX.VAL ( 1, 1)
0.00000E+00 =AVE.VAL 0.00000E+00 =SIG.VAL

Subarea.param
1 THRU 32 =ROW
1 THRU 64 =COL
0.00000E+00 =MIN.SUB ( 0, 0)
0.00000E+00 =MAX.SUB ( 0, 0)
0.00000E+00 =AVE.SUB 0.00000E+00 =SIG.SUB

MIN/MAX IN USE
0.00000E+00 =MIN.USE 0.00000E+00 =MAX.USE

52
The second category of functions perform arithmetic operations on the data such as addition, subtraction, multiplication and division, magnitude, square root or square of data, scale and offset and logarithm or exponentiation. These functions are as follows:

- **SROO**T .. SQUARE ROOT
- **MAG**  .. MAGNITUDE
- **MSQD** .. MAGNITUDE SQUARED
- **SCALE** .. SCALE AND OFFSET (K1*A + K2)
- **LOG**  .. LOGARITHM BASE 10 OR e
- **PHASE** .. PHASE
- **XPON** .. EXPONENTIAL(10 OR e) (EXP(K1*RE(A)+K2)
- **SQUARE**.. SQUARE OF DATA
- **ZCR**.... Zero CrossiNG RATE
- +  .. [A,(B)] + [C,(D)]
-  .. [A,(B)] [C,(D)] OR REVERSE
- * .. [A,(B)] * [C,(-D)]
- /  .. [A,(B)] / [C,(D)] OR REVERSE

The first ten operations on the above list do not require any inputs and are performed by entering the operation symbol (such as + for addition or SR for square root) for the particular operation and pressing the RETURN key. The last two operations (scaling and exponentiation) require two parameters K1 for scaling and K2 for offset.

If an operation cannot be performed such as square root or logarithm of a negative number, warning messages are issued to that effect. Following examples are typical applications of these functions:

**SIG. PROCESSING OPERATION (HELP=?) ?+**

[A,(B)] + [C,(D)] (YES) ?

Value = Y

**SIG. PROCESSING OPERATION (HELP=?) ?-**

[A,(B)] - [C,(D)] (YES) ?

Value = Y

**SIG. PROCESSING OPERATION (HELP=?) ?***

[A,(B)] * [C,(-D)] (YES) ?

Value = Y

53
SIG. PROCESSING OPERATION (HELP=?) ?ma

MAG(A,B) (YES) ?
Value = Y

SIG. PROCESSING OPERATION (HELP=?) ?sr

SQRT(A) (YES) ?
Value = Y

TAKING MAGNITUDE OF NEGATIVE VALUES

SIG. PROCESSING OPERATION (HELP=?) ?msg

MAGSQ(A,B) (YES) ?
Value = Y

SIG. PROCESSING OPERATION (HELP=?) ?lo

LOG((A,B)) (YES) ?
Value = Y
BASE 10 (ELSE e) (YES) ?
Value = Y

NEGATIVES REPLACED WITH -100.

SIG. PROCESSING OPERATION (HELP=?) ?ph

PHASE(A,B) (YES) ?
Value = Y

SIG. PROCESSING OPERATION (HELP=?) ?sc

K1*(A,B) + K2 (YES) ?
Value = Y
K1 * Q + K2
INPUT K1  1.000000 ?

54
The third category of functions perform windowing on the data. The window is a Han-
ning window and could be applied either in the time domain or frequency domain (see [1],
eq (3.8c))

HFREQ .. HANNING WEIGHTING BY CONVOLUTION (FREQ)
HTIME .. HANNING WEIGHTING BY MULTIPLICATION (TIME)

Windowing is done row by row, thus length of the window is equal to the row length. Following examples illustrate a typical application

SIG. PROCESSING OPERATION (HELP=?) ?ht
The fourth category of functions are signal processing functions. These functions perform parametric (AR and ARMA) and nonparametric (FFT) spectral estimation as well as other features such as energy and zero crossing rate. These functions are:

1DFFT . . 1-DIMENSIONAL FFT
FFTM . . FFT MAGNITUDE
1DFFT . . INVERSE 1-DIMENSIONAL FFT
2DFFT . . PERFORM 2D FFT
I2DFFT . . PERFORM INVERSE 2D FFT
HFREQ . . HANNING WEIGHTING BY CONVOLUTION (FREQ)
HTIME . . HANNING WEIGHTING BY MULTIPLICATION (TIME)
PPIC . . PICK THE PEAKS AND THE CORRESPONDING AMPLITUDES
PEAK . . PICK THE HIGHEST PEAKS (MAXIMA)
TDA . . SYNTHETIC TRANSIENT DATA

In the sequel we will describe these functions in more detail.

**One-dimensional FFT**

This signal processing function is invoked by entering 1D and pressing the RETURN key while in the signal processing mode.

SIG. PROCESSING OPERATION (HELP=?) ?1d

1-D FFT (FULL ARRAY ONLY) (YES) ?

Value = Y
(R)OW/(C)OLUMN FFT ?r
The code will then inquire the user about performing the FFT on the full array or a subarea. The default value entered by pressing the RETURN key is YES, that is FFT is performed on the entire array. The next inquiry is for row by row processing (by entering R). If FFT is done row by row, then length of FFT is equal to the row length. If it is done column by column, then FFT length is equal to the column length. In any case FFT length must be a power of 2. If this condition is not met a warning message is issued to this effect. One-dimensional FFT is performed on array A only if data is real, otherwise it is performed on both arrays A and B. The real and imaginary parts of FFT are placed into arrays A and B respectively. Hence the original data is written over in the process. If data is to be used for later processing, the Copy option must be used to save the data in arrays C and D respectively.

**FFT Magnitude**

Another version of FFT is the FFM option. This option produces the magnitude square of the discrete Fourier transform (DFT), that is the DFT spectrum instead of the real and imaginary parts. The sequence of user queries are the same as for one-dimensional FFT and the same length requirements apply here too. Following example illustrates the application of FFM option

```
SIG. PROCESSING OPERATION (HELP=?) ?fftm

FFT-MAGNITUDE (FULL ARRAY ONLY) (YES) ?

Value = Y
(R)OW/(C)OLUMN FFT ?r
```

**Inverse One-Dimensional FFT**

Inverse one-dimensional FFT can be performed by using 1DFFT (or just 1) option. Application of this option is similar to 1DFFT and FFM options. Following example illustrates a typical situation

```
SIG. PROCESSING OPERATION (HELP=?) ?1

INVERSE 1-D FFT (FULL ARRAY ONLY) (YES) ?

Value = Y
(R)OW/(C)OLUMN FFT ?r
```

**AR Spectral Features**

AR spectral features and AR spectrum can be computed by involving the AR option in
the signal processing menu. Upon invoking this option the code will issue a reconfirmation message as to whether to continue with AR processing or not. If the user enters YES (or just Y) then the code will inquire the user about whether to do the AR processing row by row or column by column. The following example illustrates these steps.

SIG. PROCESSING OPERATION (HELP=?) ?ar

AR SPECTRAL FEATURES (YES) ?

Value = Y
(R)OW/(C)OLUMN PROCESSING ?r

There are several different versions of AR estimators (refer to Section 4 of [1]). The particular technique implemented here is called the pre-and-post windowed (autocorrelation) method.

To perform AR spectral analysis two parameters must be specified: length of the analysis frame and order of the AR spectrum. The first parameter is the number of data points to which an AR model (all-pole filter) is to be fitted, the second parameter specifies the order of this AR model. The code inquires the user about these two parameters. The minimum, maximum and default values for analysis frame are 0, 512 and 128 data points. The corresponding values for the order are 1, 40 and 10. Frame length must be a power of 2 due to AR spectrum computation constraint. If this requirement is not met, a warning message is issued to this effect. AR spectral features are computed for data in array A, the spectrum is computed and written over the data in array A in exactly the same locations as the original data itself. In other words, an in-place AR spectrum is computed. The AR spectral features, that is the predictor coefficients (coefficients of the all-pole filter), the reflection coefficients and log area ratios (area functions) are written to the file MAIN.DBG. The contents of this file could be displayed or printed out by using the system print option. The following example illustrates the parameter inputting procedure.

LEN- LENGTH OF ANALYSIS FRAME( POWER OF MN= 0 MX= 32
DF= 128 ? 32

Value = 32
ORDER OF THE AR PROCESS MN= 1 MX= 40
DF= 10 ?

Value = 10
SPECTRAL FEATURES ARE IN FILE MAIN.DBG SPECTRUM IS IN ARRAY A

ARMA Spectral Features

ARMA spectrum estimation is very close in concept to AR estimation, the difference being that a pole-zero model instead of an all-pole model is fitted to the data. There are
a host of different ARMA estimators. The particular technique implemented here is due to Friedlander and is based on modified Yule-Walker equations. Refer to Section 5 of [1] for more details. ARMA spectrum estimation can be performed by invoking the RMA option of the signal processing menu. To perform ARMA estimation four parameters must be specified by the user. These parameters are the following:

1. ILEN-length of the analysis frame: is the number of data points to which the pole-zero model is fitted. ILEN must be a power of 2. If this requirement is not met, a warning message is issued to this effect.

2. P-order of the ARMA process: is the denominator degree of the pole-zero model.

3. PHAT-order of the modified Yule-Walker equations: this parameter specifies the order of the AR model that is fitted to the data before mode selection. This parameter is automatically set to be three times the order P unless otherwise specified by the user.

4. NLAG-number of correlation lags to minimize: this is the number of correlation lags that is minimized in the estimation procedure. The value of this parameter is internally set to be four times PHAT unless otherwise specified by the user. Refer to Section 5 of [1] for more detailed information on these parameters. The minimum, maximum and default values for each of these parameters are given in the code. The RMA option performs in-place ARMA spectral estimation. The computed spectrum is the AR part of the ARMA spectrum and is written over the data. The spectral features, that is the denominator coefficients of the pole-zero model, the reflection coefficients and log area ratios are written to file MAIN.DBG. Following example illustrates the application of the RMA option.

<table>
<thead>
<tr>
<th>SIG. PROCESSING OPERATION (HELP=?)</th>
<th>rma</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARMA SPECTRAL FEATURES (YES) ?</td>
<td></td>
</tr>
<tr>
<td>Value = Y</td>
<td></td>
</tr>
<tr>
<td>(R)OW/(C)OLUMN PROCESSING ?r</td>
<td></td>
</tr>
<tr>
<td>ILEN-LENGTH OF ANALYSIS FRAME (POWER OF MN= 0 MX= 32 DF= 128 ? 32)</td>
<td></td>
</tr>
<tr>
<td>Value = 32</td>
<td></td>
</tr>
<tr>
<td>P-ORDER OF ARMA PROCESS MN= 1 MX= 30 DF= 10 ? 2</td>
<td></td>
</tr>
<tr>
<td>Value = 2</td>
<td></td>
</tr>
<tr>
<td>PHAT-ORDER OF KYW EQS. MN= 1 MX= 120 DF= 6 ?</td>
<td></td>
</tr>
<tr>
<td>Value = 6</td>
<td></td>
</tr>
<tr>
<td>NLAG-NO. OF COR. LAGS TO MINIMIZE MN= 6 MX= 31 DF= 24 ?</td>
<td></td>
</tr>
<tr>
<td>Value = 24</td>
<td></td>
</tr>
<tr>
<td>59</td>
<td></td>
</tr>
</tbody>
</table>
Energy Function

Energy function provides the average energy of the signal over a sliding rectangular window, that is, the energy versus time. Energy function can be invoked by using the NG option of the signal processing menu. To compute the energy function, one parameter must be provided: half window length of the sliding window. This parameter specifies the smoothness of the energy function (degree of averaging). The larger this parameter, the smoother the energy function. Refer to Section 2.1 of [1] for more details. The minimum, maximum and default values for this parameter are 1, 100 and 15. The following example illustrates the use of NG option.

```
SIG. PROCESSING OPERATION (HELP=?) ?ng

ENERGY FUNCTION (YES) ?
Value = Y
(R)OW/(C)OLUMN ENERGY ?r
HALF WINDOW LENGTH MN= 1 MX= 100
DF= 15 ?
Value = 15
```

The result (energy function) is written into array A.

Zero-crossing Function

Zero-crossing function is the number of times a signal crosses the time axis. Zero-crossing function is an indicator of the signal "activity" regardless of the signal amplitude. Zero-crossing function can be invoked by using the ZC option in the signal processing menu. Like energy function, to compute the zero-crossing function, the half width of the sliding rectangular window must be specified. The bigger the window length, the smoother the zero-crossing function will be. The minimum, maximum and default values for the half window length are 1, 100 and 15. The following example illustrates a typical application of the ZC option.

The result of this operation (zero-crossing function) is written into array A.

Example:

```
SIG. PROCESSING OPERATION (HELP=?) ?zc

ZERO CROSSING (YES) ?
Value = Y
60
```
Synthetic Data

The SYN option is to produce synthetic data. The synthetic data options are as follows:
- Sinewaves
- AR Data
- ARMA Data

Once the user has selected an option, the user is prompted for the length of the synthetic data.

Signal-to-Noise

Once the user has selected a data type, the user is prompted for the length of the synthetic data. There also is a signal-to-noise ratio which ranges from -100dB to 1000dB. The user can select an appropriate response. NP is also queried from the user for the number of sinewaves. The minimum NA is 1 and the maximum is 100. J-th normalized frequency, I-th amplitude, and I-th phase also need to be entered by the user. These last three options are requested for each sinewave. After answering the above queries, synthetic data is created and placed into array A.

PPIC

The option PPIC is for picking peaks. This computes the peaks and the corresponding energies in a given spectrum. The results are sent to the MAIN.DBG output file as well as the terminal.

Query

The QU, or query option allows the user to view what is in the five working arrays. Once QU is invoked, the system prompts the user as to which array is to be queried. When the user enters the array desired (A-E), the system responds with the current data-file and the type of data existing in the array selected.

The user can also enter a DCL level command from the signal processing environment by entering a $ before the command.