

فمشتشف

NATIONAL BUREAU OF STANDARDS MICROCOPY RESOLUTION TEST CHART



BESEARCH REPORT ON A TARGET TRACKER USING & DOPPLER COMPENSATED CORRELATION TECHNIQUE

by

B. R. Eldridge

Prepared in Response to: Contract N00014-81-C-0535 with The Office of Naval Research May 29, 1985

Tetra Tech/Honeywell, Inc 1911 N. Fort Myer Dr. Arlington,VA 22209



This document has been approved for public release and sale; its distribution is unlimited.

CALL AND A

85 6 4 032



RESEARCH REPORT ON A TARGET TRACKER USING A DOPPLER COMPENSATED CORRELATION TECHNIQUE

È

[.

_

Ċ

[]

Ľ

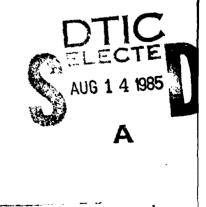
by

B. R. Eldridge

Prepared in Response to: Contract N00014-81-C-0535 with The Office of Naval Research May 29, 1985

Tetra Tech/Honeywell, Inc 1911 N. Fort Myer Dr. Arlington,VA 22209

i



This document has been approved for public release and sale; its distribution is unlimited.

PREFACE

in the second se

1

ب ا

r

Ľ

This document is the final research report on the investigation of a mathematical algorithm to do target tracking, using doppler compensated correlation techniques on input time series streams from several passive acoustic sensors. The algorithm was developed and programmed into a testbed on a VAX-750 computer and was tested using simulated time series data generated by the Tetra Tech Broadband Signal Simulator. Algorithm performance proved dissapointing due to: (1) Numerical instabilities induced by structural anomolies in the sample signal autocorrelation function; (2) The extreme sensitivity of objective function to choice of signal characteristics and processing parameters; (3) Computational intensity of the algorithm.

Accession For NTIS GRA&I DTIC TAB Unannounced Bv____ Distribution/ Availability Codes wail and/or Special Dist



TABLE OF CONTENTS

Ċ

.

e

Ļ

	PREFACE	11
1.	INTRODUCTION	1
2.	ALGORITHM DESCRIPTION	4
3.	THE ESTIMATION SCHEME	7
4.	DOPPLER COMPENSATION	12
5.	THE MINIMIZATION PROCESS	15
6.	ALGORITHM PERFORMANCE	18
7.	SUMMARY AND CONCLUSIONS	24
	APPENDIX A: TESTBED COMPUTER PROGRAM LISTINGS	25

LIST OF FIGURES

Ņ

5.1	Tracker Flow Logic	17
6.1	Test Case Geometry	19
6.2	Objective Function vs. Errors in Velocity Probe	21
6.3	Objective Function vs. Errors in Velocity Probe	21
6.4	Objective Function vs. Errors in Velocity Probe	21
6.5	Objective Function vs. Errors in Velocity Probe	21
6.6	Objective Function vs. Position Error	21
6.7	Objective Function vs. Position Frror	21

1. INTRODUCTION

i C

[]

r.,

.

C

Ľ

For the past year, Tetra Tech has been involved in the development and analysis of an algorithm for tracking maneuvering submarines using Doppler compensated correlation techniques. The intended goal was the identification of an algorithm which would be more responsive to target kinematic changes than the usual Kalman filter, thereby providing timely and accurate estimates of target position course and speed at various times along the track. The work was carried out under contract N00014-81-C-0535 with the Office of Naval Research.

The algorithm was developed and a testbed computer code was generated for implementing and testing the algorithm. This program was written in FORTRAN-77 on a VAX-750 computer and is set up to use the Tetra Tech Broadband Signal Simulator (BSS) outputs as as its input time series. No attempt was made in this effort to use actual at sea data. The FORTRAN listings of the testbed program are included in Appendix A of this report.

The algorithm research carried out by Tetra Tech assumed the availability of base banded, bandlimited digitally sampled time series from several passive acoustic sensors. The availability of information such as bearings and time delays were not assumed in the testbed program, since it was felt that working directly with the time series data would provide a more convenient means of inplementing a maneuvering target tracker. Due to the general structure of the algorithm, additional measurement types such as those mentioned above can be easily added if desired.

As is well known, one of the characteristics of a sequential or Kalman type of estimating scheme is the tendency of the estimator to build up "inertia" and thereby make it unresponsive to changes in target course and speed after long periods of tracking time. This may be overcome by certain ad hoc schemes such as frequent reinitializations or possibly by tampering with the weights so as to cause the algorithm to have a shorter memory, etc.

In view of this it seemed reasonable to employ an estimation scheme which works directly with selected blocks of time series data in a batch mode, and which can be highly overlapped from estimation to estimation. Once the algorithm has been initialized and is operation, the previous estimate of target state can be used to initialize the trial solution for the current estimation. The covariance matrix of the initialization state is not carried over and therfore the process is without a memory. However, if sufficient overlap in the input time series is used, the output states should show mimimal change from estimation to estimation while still reflecting the most current information available from the time series.

いた

Time series generated by a single moving source and received at two or more spacially separated sensors will exhibit different Doppler and time delay characteristics at each of the receivers. These characteristics are, of course, dependent on sensor-target geometry and kinematics, as well as sound propagation physics. By applying the appropriate time and Doppler compensation to the received time series, pairwise time series correlations between sensors can be maximized. By linking the time and Doppler compensation to assumed target motion, one can adjust the target state parameters to maximize (or minimize) an appropriately chosen function of the corresponding pairwise cross correlation estimates.

The assumption of digitally sampled time series sampled at a uniform sample rate suggests that the time and Doppler compensation be done in the time domain using an interpolative resampling technique. This invloves estimating time series values whose sample times lie between the discreet sample times of the input time series. For band limited signals, the well known Sampling Theorem provides a rational means of performing the required interpolation using neighboring time series points and the sinc function as an interpolating function. This, in effect, generates a piecewise continuous representation of the original time series thereby allowing resampling at arbitrary times which are in concert with trial target state parameters. Also, such a scheme allows analytic evaluation of the gradient vector with respect to the state variables and weighting vectors.

As has been mentioned above, Tetra tech has implemented these ideas into a testbed algorithm on the VAX-750 digital computer. Inputs to the algorithm consist of up to 10 channels of time series data. For each channel, the algorithm requires an estimation of signal to noise ratio (SNR) along with the standard deviation of the SNR for that channel. The algorithm is also sensitive to such inputs as integration time, processing bandwidth and center frequency, station location, sound speed in water, time series overlap, and initialization parameters such as position course and speed. The target kinematic model consists of polynomials in water time of up to degree 5 for x,y and z. The order of the polynomials is user specified. Model output consists of estimated target parameters, their associated error variances, and the size and orientation of the 2- σ containment ellipsoid.

The algorithm has been tested using simulated time series generated by the BSS. The BSS can emulate the complex time series generated by a moving target having user specified kinematic and spectral output signal characteristics.

Section 2 of this report gives an overall description of the workings of the algorithm. Section 3 describes the Gauss-Newton estimation scheme as it applies to this effort, and Section 4 outlines and justifies the doppler compensation scheme that we have employed. Section 5 contains the objective function minimization process description and algorithm performance is reported in Section 6. Finally, Section 7 presents the summary and conclusions of this effort. Appendix A contains the FORTRAN listings of the testbed program developed as part of this effort.

2. ALGORITHM DESCRIPTION

The algorithm estimation scheme assumes the availability of several channels of bandlimited, basebanded, discreetly sampled digital data for which all of the pertinent parameters such as sample rate, bandwidth, and center frequency are known, and all of which contain signal from a common emitter. For the purposes of this analysis, we will assume that the noise on each channel is mutually uncorrelated. We will also assume that the target is moving along some 3-dimensional trajectory, which is given by the vector function P(s;t), where s is the state vector to be estimated and t is the time along the trajectory.

The testbed version of the algorithm assumes that each of the components of P(s;t) is an n'th order polynomial in t, and the state vector s consists of the coefficients of these polynomials. The testbed user may specify n to be any non negative integer up to and including 5. In practice, n is usually chosen to be 1, resulting in linear target motion at constant speed. In this case, P(s;t) is given by

$$\mathbf{P}(\mathbf{s};\mathbf{t}) = \mathbf{P}_0 + \mathbf{V}\mathbf{t} \tag{2.1}$$

and the state vector s may be represented in transposed form by

 $\mathbf{s} = [\mathbf{P}_0^{\mathsf{T}}, \mathbf{V}^{\mathsf{T}}]^{\mathsf{T}}$ (2.2)

The superscript T denotes the usual matrix transpose operator.

In order to keep the algorithm tractible, we have assumed linear, constant speed sound propagation. More complicated propagation models could have been incorporated, but it was deemed an unnecessary complication at these early stages of algorithm development and feasibility analysis.

The central idea of the algorithm is to mimimize a quadratic form of "system functions". The system functions are dependent on the collection of pairwise normalized sample correlation envelope functions which have been adjusted to account for assumed target kinematics. Each sample correlation envelope function is obtained by correlating the samples from a selected reference channel with modified sets of samples that have been interpolated and resampled from each the other channels comprising the tracking system. The resampling times are calculated as a function of the current value of the state vector, sensor kinematics, the assumed propagation model, and channel signal processing parameters such as center frequency, sample rate, and bandwidth.

To be specific, let us consider a pair of channels, say channels X and Y. Pick a set of samples $\{X_1, X_2, ..., X_n\}$ from channel X. These samples correspond to arrival times $\{u_1, u_2, ..., u_n\}$ on channel X. In order to do the motion compensation, we need to calculate the corresponding arrival times $\{v_1, v_2, ..., v_n\}$ on channel Y. This is done by using the candidate source trajectory P(s;t) to calculate emitter times $\{t_1, t_2, ..., t_n\}$ corresponding to the X channel arrival times $\{u_1, u_2, ..., u_n\}$, and using these emitter times to project the corresponding arrival times $\{v_1, v_2, ..., v_n\}$ on channel Y. We then interpolate and resample the Y channel at the $\{v_1, v_2, ..., v_n\}$ thereby obtaining a new set of samples $\{Y_1, Y_2, ..., Y_n\}$. The interpolation is accomplished using a truncated sinc function as an interpolating function. Details of the interpolation scheme are provided in Section 4 of this report.

The magnitude squared cross correlation estimate δxy is then calculated by

$$\delta x_{i} = \left[\sum_{i} X_{i} Y_{i}^{*} \right] / \left\{ \sum_{i} |X_{i}|^{2} \sum_{i} |Y_{i}|^{2} \right\}$$
(2.3)

where the three sums in the above expression are taken over i=1,2,...,n and the superscript (*) denotes complex conjugation. The values of the δxy so obtained are used to form the aformentioned system functions Fxy which are used in the minimization process. The system functions are given by

$$Fxy=ln(Gxy/\delta xy)$$
(2.4)

where Gxy is the a priori expected value of Xxy. Gxy is related to the input SNR estimates on each channel by

 $Gxy=[(1+SNRx^{-1})(1+SNRy^{-1})]^{-1}$ (2.6)

Note that the Fxy are chosen such that they have value 0 when given error free information.

Finally, Q(s), a positive definite quadratic form of the Fxy, is formed over all channel pairs, and by using gradient methods, the state vector s is adjusted so as to mimimize Q(s). The vector s_0 which mimimizes Q(s) is taken as the state estimate.

A fallout of the minimization process is an estimate of the state covariance matrix. This matrix is used to calculate the ellipsoidal containment region which provides the user with a geometric indication of algorithm performance.

3. THE ESTIMATION SCHEME

3.1 Preliminary Details and Notation

this section details the estimation scheme in rather general mathematical terms. Preliminary to the discussion we establish the following notation which will be used throughout the remainder of this report.

If X and Y are n-dimensional complex valued vectors, the complex inner product of X and Y, denoted by $\langle X, Y \rangle$, is defined by

$$\langle \mathbf{X}, \mathbf{Y} \rangle = \sum X_{j} Y_{j}^{*}$$
(3.1)

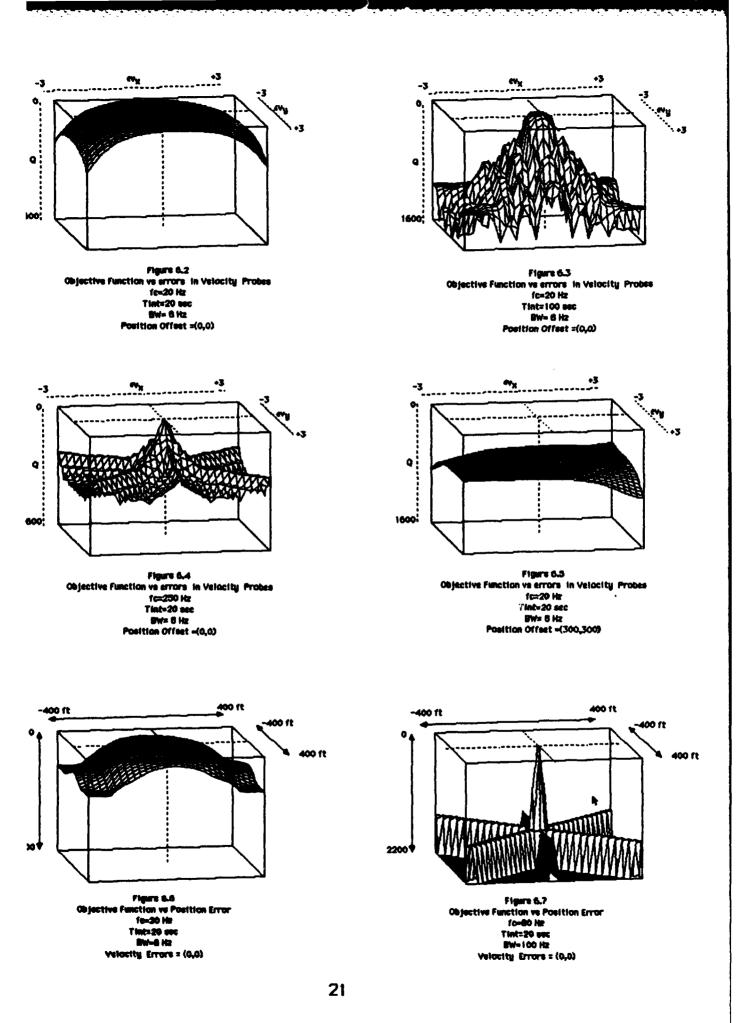
where the sum is taken over i=1,2,...,n, and the X_i and Y_i are the complex valued components of X and Y, respectively. The (*) notation denotes complex conjugation. Note that the complex inner product is conjugate symmetric in that the following relationship holds:

We define the norm of X, denoted by [X], by

In this notation, Equation 2.3 of Section 2 becomes

$$\forall xy = |\langle X, Y \rangle |^{2} | | X | |^{2} | Y | |^{2} \}$$
(3.4)

If each of the components of the complex valued vector is a differentiable function of some real parameter θ , then denote the vector consisting of the corresponding derivatives (partial derivatives) by dX/d θ (∂ X/ $\partial\theta$). It is easy to verify that the following useful relationship is true



: .

۰.

· .

Ŀ

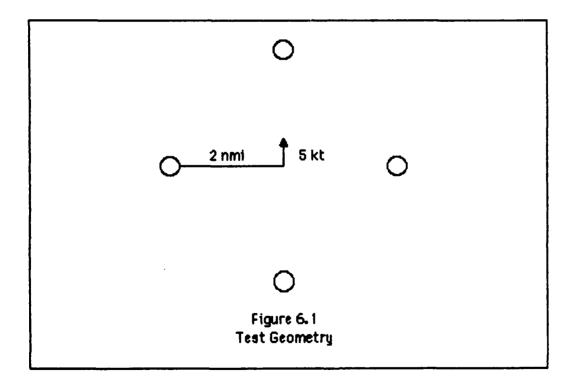
The following figures show the kinds of difficulties one faces even under the best of circumstances. Figures 6.2-6.5 show plots of the objective function versus errors in the probe or trial solution velocities(ft/sec) in both the x and y direction for several sets of processing parameters. The assumed position error is assumed to be fixed at 0 ft. The velocity errors range from ± 3 ft/sec in both the x and y direction. The center of the portion of the x-y plane shown represents 0 error. The z axis represents the values of the objective function and has been inverted for the sake of this presentation. The z axis ranges from 0 (top) to 1600 (bottom). The signal bandwidth is assumed to be 8 Hz.

Figure 6.2 was generated assuming a center frequency of 20 Hz with an integration time of 20 seconds and presents a very clean objective function over the plot range. In Figure 6.3, the integration time has been increased 100 seconds. The resulting plot exhibits a much more spiked peak around (0,0) and the outskirts show a good deal of ripple. Figure 6.4 is similar to Figure 6.2 except that the center frequency of the signal has been shifted to 250 Hz. This band shift has caused the peak to become very sharp with even more ripple evidenced in the outskirts. The processing situations depicted in Figures 6.3 and 6.4 would present problems to the tracker. Figure 6.5 has identical parameters as Figure 6.2 with the exception that the assumed position error has been offset to 300 ft. in both the x and y directions resulting in a much more planar shape and having increased the overall magnitude of the objective function considerably.

Figures 6.6 and 6.7 show plots of the objective function versus errors in the trial position estimates with the geometry in Figure 6.1 applying. The position errors range from \pm 400 ft in both the the x and y directions. Figure 6.6 corresponds to a signal bandwidth of 8 Hz and presents a rather smooth function with essentially no unusual structure. In Figure 6.7 we have increased the signal bandwidth to 100 Hz and moved the center frequency to 80 Hz. The resulting plot contains considerable structure with the

parameters, we developed a computer program to generate the expected value of Q(s) for a particular signal autocorrelation function. The outputs are sensitive to the aformentioned signal and processing parameters as well as target/sensor geometry and kinematics. The autocorrelation function we have chosen is triangular on the interval [-T,T] and has a spectral density function of the form $sinc^2(\omega T/2)$. Its bandwidth is approximately 1/T. This particular autocorrelation function has the advantage of being integrable in closed form. The program is setup to generate surfaces of expected values of the tracker objective function, holding the position probes fixed and letting the position probes vary.

Some results are presented for the geometry shown in Figure 6.1. Here the target is assumed to be in the center of a square box with the sensors located at the vertices. The distance from the target to each of the sensors is assumed to be 2 nmi. A SNR of 6 dB with a $\sigma_{\rm SNR}$ =3 dB is assumed.



6. ALGORITHM PERFORMANCE

6.1 Parameter Selection

Since the objective function Q(s) is a very complicated function (most of which is carried out in the complex domain) of a number of processing parameters, some thought had to be given to their effects on algorithm behavior. The user has control over such things as processing bandwidth, center frequency, and integration time.

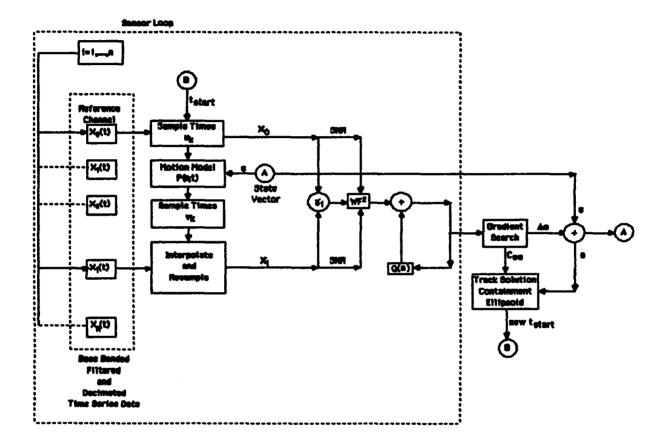
For example, if one chose to process a signal having a sufficiently high center frequency for a sufficiently long period of time, Q(s) could become quite sensitive to trial solution errors in velocity. In fact one would expect to see spike like behaviour in the objective function in the neighborhood of the true velocity components, which could conceivably cause convergence difficulties.

Similarly, a wide bandwidth signal could cause similar spike like behavior in trial solution position estimates. Small positional errors wind up in the ripple of the objective function which wreaks havoc on the convergence process we have chosen.

On the other hand, if the processing band is too narrow, the lack of time delay resolution may not provide any meaningful information to the algorithm, again resulting in poor behavior.

These were in fact some of the problems that were encountered during the early stages of algorithm testing. Since the testbed computer code was newly developed, it was not known whether poor early algorithm performance was due to bugs in the program, or whether the algorithm just did not work, or whether we had just chosen bogus processing parameters. After considerable reexamination ,rechecking, and rederiving the mathematics, we decided that they were correct. We could find no bugs in the program and so our only recourse was to give careful scrutiny to our choice of test parameters.

In order to gain insight to the effects of processing



e nei ei e

Figure 5.1 Tracker Flow Logic

and the second second

which yields

$$\partial \mathbf{s} = -(\mathbf{F}_{\mathbf{s}}^{\mathsf{T}} \mathbf{W} \mathbf{F}_{\mathbf{s}})^{-1} (\mathbf{F}_{\mathbf{s}}^{\mathsf{T}} \mathbf{W} \mathbf{F}) = C_{\mathbf{s}\mathbf{s}} (\mathbf{F}_{\mathbf{s}}^{\mathsf{T}} \mathbf{W} \mathbf{F})$$
(5.5)

The advantage of this approach is that it does not require the computation of second derivatives, and that an estimate of the state covariance matrix falls out of the process.

In summary, if s is the current trial solution for the process, we form the next iterate s' by calculating ∂s as in the preceeding equation and forming s' by

$$s'=s+\partial s$$
 (5.6)

To stop the process we check to see if the magnitude of ∂s is within some predescribed tolerance. If so the process is stopped and the current value of the state vector is returned as the solution. If not the process continues until a solution is returned or the maximum iteration count is exceeded.

The entire process is described in Figure 5.1.

5. THE MIMIMIZATION PROCESS

5.1 <u>The Iteration Scheme</u>

Ê

E

F

•

Ċ

ŕ

The function Q(s) is a complicated function of the state vector and measurement vector, the mimimization of which does not seem amenable to closed form solutions. We therefore must rely on iterative techniques to solve the problem. The following paragraphs describe the technique we have chosen to accomplish the minimization.

Recall that Q(s) is a quadratic form in the system functions and may be written

$$Q(s) = F^{\mathsf{T}} W F \tag{5.1}$$

where the weighting matrix \mathbf{W} is chosen to be the inverse of the covariance matrix of the system residual vector. Under the assumption of slowly varying weights, the gradient vector of $Q(\mathbf{s})$ may be written

$$\nabla Q(s) = 2F_s^T WF.$$
(5.2)

At a local mimimun we have the necessary condition that

$$\nabla Q(s) = 2F_s^T WF = 0 \tag{5.3}$$

An approach which has been used successfully is demonstrated in the following discussion. Suppose that the algorithm has reached a stage such that s is the current value of the trial state vector. We would like to find the perturbation ∂s to add to s which will improve the estimate. A reasonable approach is to solve the the following perturbed gradient equation for ∂s

$$\mathbf{F_s}^{\mathsf{T}} \mathbf{W} (\mathbf{F} + \mathbf{F_s} \partial \mathbf{s}) = \mathbf{0}$$
 (5.4)

receiver times using $Q_h(t)$ as the interpolating function, thereby generating a set of Y channel samples which reflect the Doppler corrections implied by the current value of the state vector.

i

, 100

Ċ

ţ

In order to obtain the receiver times on the Y channel, we solve the following pair of equations for v_k given u_k :

 $\begin{aligned} \mathbf{u}_{k} = \mathbf{T}_{k} + \left| \mathbf{P}(\mathbf{s}, \mathbf{T}_{k}) - \mathbf{P}_{k} \right| / c & (4.6) \\ \mathbf{v}_{k} = \mathbf{T}_{k} + \left| \mathbf{P}(\mathbf{s}, \mathbf{T}_{k}) - \mathbf{P}_{k} \right| / c & (4.6) \end{aligned}$

where P_X and P_y are the respective position vectors of the X and Y channel receivers, c is the speed of sound in water, and T_k is the emitter time. This is done by solving the first equation for T_k using a Newton- Raphson technique, and then using the second equation with the value of T_k so obtained to obtain v_k .

Recall that we are assuming that all of the input time series data has been basebanded from some center frequency f_c . Because of this, a phase correction prior to correlation is required on both channels. This merely amounts to heterodyning the samples back up to their original center frequency f_c . Form the complex vectors X and Y whose k'th components are given by $exp{2\pi j f_c(u_k-u_0) }X_{n+k-1}$ and $exp{2\pi j f_c(v_k-v_0) }Y_{n+k-1}$, respectively. Then δxy is given by

 $\delta x y = \langle X, Y \rangle |^{2} \langle |X|^{2} |Y|^{2} \rangle$ (4.7)

4.3 <u>System Function Derivatives</u>

The algorithm uses gradient methods to minimize Q(s), which is the quadratic form of the system functions Fxy as discussed above. This necessitates the calculation of the derivatives of Q(s)with respect to each of the state variables. Most of the work is done in computing the derivatives of the δxy with respect to the measurement vector and each of the state variables. The mathematical development of these derivatives is straightforward but extremely tedious and will not be included here.

reconstructed from its samples, provided that the digital sample rate is at least as great as the bandwidth of the signal. The reconstruction uses the "sine x over x" or sinc as an interpolating function. Specifically, if Z(t) is a time series having non zero frequency content only in the interval [-b/2,b/2], and if Z(t) is uniformly sampled over all time at a sample rate $f_d \ge b$, producing samples Z_n , $-\infty \le n \le \infty$, and such that Z_0 corresponds to a sample time of 0, then Z(t) is reconstructed exactly from its samples by

$$Z(t) = \sum \sin(\pi (f_d t - n)) Z_n$$
(4.3)

The above sum is taken over all n. This, however, involves summing over an infinite number of elements. It therefore seems reassonable to approximate the reconstruction of Z(t) from its samples by using a time limited version of the sinc function. If we define the interpolating function $Q_h(t)$ by

 $\begin{array}{cccc}
sinc(t), & |t| ≤ h \\
Q_{h}(t) = & (4.4) \\
0 & |t| > h
\end{array}$

, then Z(t) may be approximated by

Î,

Ì

E

.

•

•

Ċ

Ľ

which, for any given value of n involves only finite sums.

Let us assume we are working with channels X and Y and we wish to calculate δ_{Xy} for the current value of the state vector. Pick a set of reference samples $\{X_{n}, X_{n+1}, ..., X_{n+M-1}\}$ from channel X. These samples correspond to the set of channel X receiver times $\{u_{n}, u_{n+1}, ..., u_{n+M-1}\}$. We the use P(s;t) and the assumed propagation model to determine the corresponding set of channel Y receiver times $\{v_{n}, v_{n+1}, ..., v_{n+M-1}\}$. Channel Y is then interpolated at these

4. DOPPLER COMPENSATION SCHEME

4.1 <u>Time Series Assumptions</u>

r

(

-

.

-

R.....

.

Ċ

3

Ę

The algorithm assumes the availability of M channels of data, and that the time series for each data channel contains basebanded, bandlimited data that has been sampled at at least the Nyquist sampling rate. For the sake of notational simplicity, we will assume that all of the channels have the same center frequency, f_c , and the same digital sampling rate f_d . We also assume that the n'th sample on each channel occurs at the same time. The time between samples, Δt , is given by

$$\Delta t = 1/f_{\rm fl} \tag{4.1}$$

Therefore if the 0'th sample corresponds to t_0 , then the n'th sample corresponds to t_n , where

 $t_n = t_0 + n\Delta t \tag{4.2}$

The algorithm requires an estimate of the signal to noise ratio and its corresponding error variance on each channel of input data. It is assumed that these are independently specified and will be denoted SNR_m , where the subscript m refers to the particular channel designator.

4.2 Resampling and Phase Compensation

In order to effect the proper Doppler compensation, it is necessary to interpolate between samples in the time domain, with the interpolation times reflecting the current value of the state vector.

The well known sampling theorem from signal processing states that a complex valued bandlimited signal can be completely

Suppose we want to add a new measurement set to the algorithm that is independent of those already incorporated and can be described with a single system equation. This merely requires the specification of the corresponding system function and its associated partial derivatives. In keeping with our earlier discussion, let us further assume that the partial of the new system equation with respect to its measument vector are independent of the state vector. In this case we first form the new scalar weight w by

$$w=1/\sum[(\partial F/\partial m_i)\sigma_{m_i}]^2$$
 (3.19)

where ${\sf F}$ is the new system function, the ${\sf m}_j$ are its associated measurements and

$$\partial F / \partial \mathbf{m}_{F} = [\partial F / \partial \mathbf{m}_{i}].$$
 (3.20)

The updated B vector and Z matrix are given by

Ē

þ

Ċ

Ċ

Ľ

$$B=B_{old} + wF(\partial F/\partial S)$$
 (3.21)

 $\mathbf{Z} = \mathbf{Z}_{old} + \mathbf{w} (\partial F / \partial \mathbf{S}) (\partial F / \partial \mathbf{S})^{\mathsf{T}}$ (3.22)

where $\partial F/\partial S$ is the vector of partials of F with respect to the elements of the state vector.

where F_s denotes the matrix whose mn'th element is $\partial F_m / \partial s_n$. If the above equation holds, a perturbation in the measurement vector ∂m induces a perturbation in the state vector ∂s , which to within first order terms, obeys the relationship

$$F_s^T W(F + F_s \partial s + F_m \partial m) = 0.$$
 (3.14)

This implies

$$\partial \mathbf{s} \approx -(\mathbf{F_s}^T \mathbf{W} \mathbf{F_s})^{-1} \mathbf{F_s}^T \mathbf{W} \mathbf{F_m} \partial \mathbf{m},$$
 (3.15)

which yields, after substituting C_{mm}^{-1} for W

$$C_{ss} \approx E(\partial s \partial s^{T}) = (F_{s}^{T} W F_{s})^{-1}$$
(3.16)

The expression for C_{ss} given above is extremely convenient and can be used to provide geometrical insight to algorithm behaviour. In particular, it is used to derive the ellipsiodal containment region of the current estimate of the state vector.

3.3 Incorporation of Additional Measurements

The form of the estimator used in this algorithm has the advantage that it is easy to incorporate new types of measurements should the need arise. If, for example, the system can provide independent estimates of position and velocity or if another independent sensor comes on line, the new measurements so provided may be entered as an additive partitions to the system weighting matrix and gradient vector. Let us first establish the following notation. Let

(3.17)

 $Z = (F_S^T W F_S)$

r

ð

: •

E

r_

t

Ľ

Ċ

ġ

We now provide our rational for our choice of W. Let m denote the vector of measurements associated with F. In our case, each component of m is an SNR estimate from one of the output channels. For any fixed value of s, the perturbation in the vector F induced by a perturbation in the vector m is approxumated by

Here $[F_m]$ is the matrix whose ij'th element is $\partial F_i / \partial m_j$. Assuming the error distribution is zero mean with covariance matrix C_{mm} , and that the approximation to ∂F holds over the probable values of **m**, we may approximate the covariance matrix of **F** by

The weighting matrix **W** referred to in in Equation (3.8) is chosen to be $(C_{FF})^{-1}$. Therefore the scalar function Q(s) is given by

$$Q(s)=F^{T}(C_{FF})^{-1}F$$
(3.11)

This choice of W has intuitive appeal in that measurements with higher variance get less weight and, therefore, have less of an effect on the final outcome.

We now turn our attention to the approximation of the output state covariance matrix, which will be denoted by C_{ss} . The components of the state vector s_0 that minimizes Q(s) satisfy the equation

$$\partial Q/\partial s_i = 2(\partial F/\partial s_i) WF + F^{T}(\partial W/\partial s_i)F = 0, i = 1, 2, ..., k$$
 (3.12)

If we assume slowly varying weights which enables us to ignore the second term in the above equation, then s_0 satisfies the system of equations

The above relationship is indespensible in computing the gradient derivatives during the minimization process.

3.2 <u>Theoretical Development</u>

E

E.

14

÷.,

Ċ

Ľ

Same and a second

The general theoretical basis for the estimation scheme is an extension of the techniques of linear optimal estimation theory to the nonlinear case. We evaluate a set of system equations Fxy as given by Equation () and then minimize a positive definite quadratic form of the Fxy. Suppose we have k such system functions for a particular application. Changing notation for convenience, let the system functions be denoted by F_{1} , F_{2} ,..., F_{k} and let

$$F(s) = [F_1, F_2, ..., F_k]^T$$
 (3.6)

denote the k-dimensional vector of system equations. Note that the solvability of the above equations implies that $n_s \le k$, where n_s is the dimentionality of the state vector.

If the geometric and signal assumptions are perfectly compatible with the measurement data, there exists a value of the state vector \mathbf{s}_0 for which

F(s)=0. (3.7)

In general, however the data does not support perfect compatibility, and for each value of the state vector \mathbf{s} , $\mathbf{F}(\mathbf{s})$ may be considered a vector of F-residuals. An optimal estimate of \mathbf{s} is a vector which minimizes a particular quadratic form in the F-residuals. If \mathbf{W} is an appropriately chosen positive definite symmetric matrix, the optimal estimate is the value which minimizes the scalar function

 $Q(s)=F^TWF.$

(3.8)

crisscrosses being corresponding to time delays among the various sensors. Again, finding the minimum of such a function if the solution starts off of the main peak presents quite a formadible task to the tracker.

F

Ę

5

.

Ì

ŕ

r

We have presented these results to demonstrate difficulties in choosing a set of operating parameters for which we may have some hope of achieving positive results. To this end and after having a large number of cases as above, we chose to work with a time series centered at 20 Hz containing an 8 Hz signal in 10 Hz wide total processing bin. The signals were generated by the BBS for sensors having the geometry described in Figure 6.1 above, and had an overall SNR of +6dB in the processing band to each of the sensors. Using an integration time of 20 seconds and initialiazing the algorithm with "truth" at various points along the target track, the algorithm evidenced unstable convergence in nearly every case. Since we had gone over the computer program and the mathematics very carefully and could find no errors, we had to look elsewhere for an explanation of the poor algorithm performance. We believe the answer lies in the form of the system functions.

The Z matrix and B vector defined in Section 3 of this report are used extensively in the Gauss-Newton mimimization technique that we chose to implement for this algorithm. Note that both Z and B contain the partial derivatives of the system functions with respect to each of the elements of the state vector. When the system is near a solution, the partials of the system functions are near 0, since they are essentially the derivatives of the signal autocorrelation function at t=0. This causes the Z matrix to become numerically unstable as the process nears a solution, thereby causing unpredictable algorithm behaviour. Efforts to accomodate alternate forms of the system functions turned proved unsucessful because we could not find a tractible method of comparing time series that did not involve correlations. The only other alternative to sucessfully implement the algorithm would be to incorporate a more sophisticated minimization technique which would overcome the instability problems. This, too, could have drawbacks since most such algorithms involve several one dimensional searches which require several objective function evaluations which is quite computationally intensive.

r

Ŋ

-)

•••

ř

Ļ

We actually tried a method which involves halving the length of the search vector ∂s until $Q_{k+1} < Q_k$ at which time we would recalculate a new search vector and continue the process. This proved to be too computationally intensive even for TW products on the order of 200 and including 4 sensor geometry. We did, however obtain some success in achieving algorithm convergence. However, each evaluation of Q(s) took about 2 minutes CPU time on the VAX, resulting in enormous total algorithm processing times. This was deemed unsatisfactory from the point of view of any practical application. 7. SUMMARY AND CONCLUSIONS

F

ļ

٠.

.

_

Ì

,-• •

E

.

⊈_

Í,

**

An algorithm to use doppler compensated resampling and correlation techniques on digitally sampled time series was developed and tested using synthetic time series data generated by the Tetra Tech Broadband Signal Simulator (BBS). The algorithm was designed to provide timely track estimates by using highly overlapped time series segments from the receiving sensors in a batch mode, thereby giving the process a short memory and increased responsiveness to target maneuvers. The target motion model consisted of polynomial functions of time of arbitrary order up to 5.

Algorithm performance proved dissapointing for several reasons:

(1) Computational intensity was more than was originally envisioned.

(2) Structure of the correlation functions induced numerical instabilities in the convergence process which could not be easily overcome.

(3) The structure of the objective function is, in general, quite irregular, thereby requiring careful, and perhaps limited, choice of processing parameters in order to have any hope of successful performance.

Some improvement in running time could be achieved by simplifying the resampling technique to use first and perhaps second order time series stretches based on the current value of the position and velocity estimates.

Curing the numerical instabilities seems to be the most difficult hurdle to overcome due to the often unusual structures evidenced in correlation functions. For this reason, we feel that any further endeavers to improve upon such an algorithm would prove to be dissapointing and recommend that further research be discontinued.



r

.

t

•

1

r

-

-

TESTBED COMPUTER PROGRAM LISTINGS

· · · · · · · · · · · ·

F.age VAX-11 FORTRAN V3.2-37 DISK\$USER1:CELDRIDGEJTRDRIV1.FOR;21 COMMON/TRWOKK/SV(3,0:10),F,FX(31),BB(31),FH0,FHOSG,Z2(961),ITCOUNT,TS 28-May-1985 17:32:45 14-May-1985 12:14:28 COMMON/TRTIMS/FILE_KEF(32),CFQ_REF,RS2_REF,SR_REF,T_REF,FOS_REF(3) CALL GTSHDR(LN,FILE_KES(1,N),IKCMX,FNS_RES(N),BSZ_RES(N), STARTING TARGET ESTIMATION TIME(SEC) = ', TSTRT_REF) COMMON/TRCNTRL/NDEG,NSAMF,TSTRT_REF,TEND_REF,NSKIF,MAXIT,EFSILON ,FILE_RES(32,5),CFQ_RES(5),BSZ_RES(5),SR_RES(5),T_RES(5) ,FOS_RES(3,5),FNS_RES(5),SNR_RES(5),SIGSNR_RES(5),COH(5),BIAS(5) CALL QAA(5,'+ TIME SERIES INPUT FILE: ',FILE_KES(1,N)) WRITE(5,1300)N,CFQ_RES(N),BSZ_RES(N),SK_RES(N),T_RES(N) SK_KES(N), TSSR, CFQ_RES(N), T_RES(N), FOS_RES(1,N), IERK) QAR(5, 'ENDING TARGET ESTIMATION TIME(SEC) = ',TEND_REF) QAI(5, DEGREE OF FOSITION VS TIME FOLYNOMIALS = ',NDEG) QAI(5, 'TIME SKIF RETWEEN ESTIMATIONS(SEC) = ', TSKIF) STINE CALL GAR(5,'+ SIGMA SNR(DB) = ',SIGSNR_RES(N)) GAI(5, ' NO OF SAMFLES FER INTEGRATION = ', NSAMP) (LINIX4, QAR(5,' INITIAL Z-COORDINATE(DEFTH-FT) = ',ZINIT) QAR(5,' INITIAL SFEED(KTS) = ',VINITK) (TINIS, (LINIX', SINC FUNCTION TRUNCATION PT = ',TRUNC) SRATE SNR(DB) = ', SNR_RES(N)) 11 QAR(5, 'INITIAL Y-COORDINATE(FT) = BINSZ FOS_RES(1,N)=FOS_RES(1,N)#6076. FOS_RES(2,N)=FOS..RES(2,N)#6076. FORMAT(I5,F12.3,F10.3,F10.3,F7.0,F10.0) FOS_RES(3,N)=FOS_RES(3,N)#6076. CALL DAI(5, NO DF CHANNELS = ', NCHAN) INITIAL X-COORDINATE(FT) DATA LN/21/,IRCMX/256/,EFSILON/1.E-5/ DOUBLE FRECISION ZZ, DX, F, FX, BB, RHOSO CALL TEXTI(5, '\$CHANNEL',N) CALL TEXTI(5, '\$CHANNEL',N) CALL TEXTI(5, '\$CHANNEL',N) slgs0C0H(5),ZKESBUF(0:4200),T_BUF CALL DAR(5, 'INITIAL HEADING(DEG) . DX(31),X(31),COV(4,4),DETCOV FRO. , FNS_REF, SNR_REF, SIGSNR_REF UINIT=UINITK#6076.029/3600. BYTE FILE_REF,FILE.RES CALL DAR(5,'+ , NCHAN, TRUNC, NSTATE, C TRCNTRL CMN CALL TEXT(5, CHANNEL INCLUDE 'TRUCKN.CMN' INCLUDE 'TRTIMS.CMN' DIMENSION F(3), U(3) FNS_RES(N) SU(1,N)=0. SU(2,N)=0. CALL TEXT(5, ') CALL TEXT(5, ') DO 20 N=1, NCHAN COMPLEX ZRESBUF 00 10 N=1,NCHAN PROGRAM TRDRIV 00 30 N=0,NDEG DATA C/4900./ QAR(5, CAR(5, CA QAR(5, ' (\ HEAMF \) CONTINUE CONTINUE INCLURE CALL CALL CALL. CALL CALL CALL CALL CALL CALL CALL 20 1300 10 0000 0016 9000 8000 0015 0018 0020 0026 0036 0038 0040 0003 0004 0005 0002 0012 0013 0014 0019 0023 0024 0025 0027 0028 0029 0030 0033 0034 0039 0041 0042 0043 0045 0047 0048 0049 00200 0055 0002 0011 0021 0022 0031 0032 2500 0044 0046 0051 0053 0054 0057 0001 0052

.

-			
		2	
ř		0.	
		Fade	
]		FOR	
ß		V1.I	
		s de la	
		37 E311	
Ì		73.2 2116	
		AN 1 ELEI	
		VAX-11 FORTRAN V3.2-37 DISK\$USER1:EELDRIDGEJTRDRIV1.FOR\$21	
		1 F(USEF	
		X-1 SK\$	
لم	ļ	33	
لہ. ا	{	សួយ	
		17:32:45 12:14:28	
		17:	
		985 985	
	•	1 - 7 1 - 7	
		28-May-1985 14-May-1985	
		N 4	vi e
-			SV(3,N)=0. SV(3,N)=0. SV(1111 ZINIT ZINIT ZINIT ZINIT ZINIT ZINIT ZINIT ZUNITSCOBICRINIT) VUNTIFCOBICREST, 25 BB/IERF) ZALL TRACKIEST, 25 BB/IERF) CALL TRACKIEST, 25 SFEED) CALL TRACKIEST, 25 SFEED) CALL TRACKIEST, 25 SFEED) CALL TRACKIEST, 2008 (1) , V(2)) CALL TRACKIEST, 5 COURSE(DEG) = ',5COURSE) SV(1,0)=F(1) SV(1,0)=F(1) SV(1,0)=F(1) SV(2,0)=F(2) SV(1,0)=F(1) SV(2,0)=F(2) SV(1,0)=F(1) SV(2,0)=F(2) SV(1,0)=F(1) SV(2,0)=F(2) SV(2,0)=F(2) SV(3,0)=COURSE = ',5COURSE) SV(1,0)=F(1) SV(2,0)=F(2) SV(3,0)=F(2) SV(3,0)=F(2) SV(3,0)=F(2) SV(3,0)=F(2) SV(3,0)=F(2) SV(1,0)=F(1) SV(2,0)=F(2) SV(3,0)=F(2)
•	•		KK) S S S S S S S S S S S S S S S S S S S
	•		E E E E E E E E E E E E E E E E E E E
f	Ĭ		SV(3,N)=0. XINIT VINIT#SIND(BINIT) VINIT#SIND(BINIT) VINIT#COSD(COST.EFFF) CALL TRACK(TEST.Z2FFF) CALL TEXTR(5, ` Y(TS) = `,F(1) CALL TEXTR(5, ` Y(TS) = `,F(2) CALL TEXTR(5, ` Y(TS) = `,F(2) COURSE=ATAN2D('(1),V(2)) CALL TEXTR(5, ` Y(TS) = `,F(2) COURSE ATAN2D('(1),V(2)) SV(1,0) = F(2) SV(1,0) = F(2) SV(1,0) = F(2) SV(1,0) = F(2) SV(1,0) = F(2) SV(2,0) = F(2
	-		(1) (1) (1) (1) (1) (1) (1) (1)
Ľ.			SV(3,N)=0. = XINIT = VINIT45IND(BINIT) = VINIT45IND(BINIT4) = VIN
•.			
			SV(3, N)=0. = XINIT = XINIT
	•		
			SV(1,0)=XINIT SV(1,0)=XINIT SV(1,0)=XINIT SV(1,0)=XINIT SV(1,1)=UINIT#COBI(BINIT) SV(2,1)=UINIT#COBI(BINIT) SV(2,1)=UINIT#COBI(BINIT) SV(2,1)=UINIT#COBI(BINIT) SV(2,1)=UINIT#COBI(BINIT) SV(2,1)=UINIT#COBI(BINIT) SV(2,1)=UILT#CK(FEST,ZZ CALL TEXTEG CALL TEXTEG, Y(TEST,ZZ) SV(1,0)=F(1) SV(1,1)=U(2) SV(1,0)=F(1) SV(2,1)=U(2) SV(2,1
			CONTI CO
Ľ	ۍ ۲		30 1000
		>	
	•	TRDRIV	0059 0055 0065 0065 0065 0065 0065 0005 0075 007
		Н	55555555555555555555555555555555555555
Ĺ	- -		

~		 ~ ~	 •	•
E.				
F. 84 C				
812				
F01				
171				
RDR				
VAX-11 FORTRAN V3.2-37 DISK\$USER1:[CELDKIDGE]TKDKIV1.FOK#21				
3.2				
N N N N N N N				
TKA CEI				
FOR ER1				
11 \$US				
AX- ISK				
ラニ				
មាយ	S II			
132145 114128				
1.01				
	11 · 13			
198 198	61) 1,E			
ר ת מים	2 (
28-May-1985 14-May-1985	2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4			
N 4	SO LNS			
	31),FKHO,FKHOSQ,ZZ(961),ITCOUNT,TS END_REF,NSKIF,MAXIT,EFSILON			
	~ ~ ~			
	(31) TEN			
	34) 31) 1			
	<pre>(NE_TKUFIAT(F,V,TIME)</pre>			
	SUBROUTINE TRUFFAT(INCLUDE 'TRUGKK.CHW BOUBLE FRECISION ZZ COMMON/TRUGKK/SV(3, DX(31),X(31),COV(4) INCLUDE 'TRCNTRL/NDEG OMMON/TRCNTRL/NDEG OMMON/TRCNTRL/NDEG NCDAN/TRCNTRL/NDEG OMMON/TRCNTRL/NDEG OMMON/TRCNTRL/NDEG OMMON/TRCNTRL/NDEG OMMON/TRCNTRL/NDEG OMMON/TRCNTRL/NDEG P(N)=SV(N,N OMMON/TRCNTRL/NDEG OMTINUE DO 20 N=1,3 DO 20 N=1,3 DO 20 N=1,3 DO 20 N=1,3 DO 20 N=1,3 DO 20 N=1,3 DO 20 N=1,3 CONTINUE CONTINUE CONTINUE CONTINUE CONTINUE CONTINUE CONTINUE CONTINUE CONTINUE			
	(NE TRUCK			
	ŬUÔMŬŽŽ4 MNN UAMŽŽ ULUJDIU N XXXXXIIIII N XXXXIIIIII			
	SUFFOUTINE TRUFT INCLUDE 'TRUGRK FOUBLE FRECISIO COMMON/TRUGRK/SI , DX (31) , X (31) , C INCHAN TRUCK/SI , DX (31) , X (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , C INCHAN TRUCK SI , DX (31) , Z (31) , Z (31) , C INCHAN TRUCK SI , DX (31) ,			
	CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC			
	SUBKROUTINE TRUFLAT(F,V,TIME) INCLUBE 'TRWORK,CMN' FOUBLE FRECISION 22, DX,F,FX(31),FB(31) DOUBLE FRECISION 22, DX,F,FX(31),FB(31) DX(31),X(31),COV(4,4),FFX(31),FB(31) DX(31),X(31),COV(4,4),FFX(31),FB(31) DX(31),X(31),COV(4,4),FFX(31),FB(31) DX(2000 NOUNTRUNC,NSTATE,C DIMENVTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENTRUNC,NSTATE,C DIMENSION F(3),V(3) T1=TIME-TS DIMENSION F(3),V(3) T1=TIME-TS DIMENS			
	· ·			
	0 00 M 0 0			
	3322 848111111110080080000000000000000000000			
	0001 0000 0000 0000 0000 0000 0001 0001 0011 0011 0011 0011 0011 0011 0011 0011 0011 0011 0012 0012 0012 0022 0022			

Fage VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELDRIDGEJTRDRIV1.FOR;21 COMMON/TRWDRK/SU(3,0:10),F,FX(31),BB(31),FHO,FHDSG,ZZ(961),ITCOUNT,TS 28-May-1985 17:32:45 14-May-1985 12:14:28 CALL GTSHDR(LN,FILE_RES(1,NREF),IRCMX,FNS_REF,RSZ_REF,SR_REF,TSSR, COMMON/IKTIMS/FILE_REF(32),CF0_REF,BS2_REF,SR_REF,T_REF,POS_REF(3) COMMON/TRCNTRL/NDEG,NSAMF,1STRT_REF,TEND_REF,NSKIF,MAXIT,EPSILON DATA NBUFREF/4000/,NBUFRES/4500/,TWOPIJ/(0.,6.283185307)/,LN/21/ .f.OS_RES(3,5),FNS_RES(5),SNR_RES(5),SIGSNR_RES(5),COH(5),BIAS(5) DIMENSION F(3),V(3),UO(3),UN(3),DF(3),DT(3),AWK(31),BWK(31) CALL GTSDAT(LN,FILE_RES(1,NREF),IRCMX,NMIN,NSAMF,ZREF,IERR) CALL HET(ZREF,1,,NSAMF,CFQ,REF,SR,REF) COH(NC) = FCCAL C / (1,00001+10, ##(- SNR .. RES(NC) / 10.)) FILE_RES(32,5),CFQ_RES(5),BSZ_RES(5),SR_RES(5),T_RES(5) DOUBLE FRECISION FT,Z,DET,AWK,BWK,B,RFACT,ZV,ZW COMFLEX ZREF(4000),ZRES(4000),DZRESDT,TWOFIJ COMFLEX FTC,ZEXF,DZRESDTH DATA IRCMX/256/,MXLOOF/20/,ALN10/.230258509/ NMIN=MAX(TRUNC+(TT-TINT/2.-T_REF)#SR_REF) COMFLEX#16 ZU, ZDW(33), ZDU(33), DFACT, CDOT FCCALC=1./(1.00001+10.##(-SNR_REF/10.)) DOUBLE FRECISION ZZ, DX, F, FX, BB, RHOSO DIMENSION Z(NSTATE, NSTATE), B(NSTATE) slgsqcoh(5),ZRESBUF(0:4200),T_BUF SFACT=(NSAMF-1.)/(NSAMF##2+NSAMF) , DX(31), X(31), COV(4,4), DETCOV SUBROUTINE TRACK(TT,Z,B,IERK) FNS_REF, SNR_REF, SIGSNR_REF X(I+NDEG+2)=SV(2,I)CFQ_REF, T_REF, FOS_REF, IERK) FOS__REF(1)=FOS__REF(1)*6076. POS_REF(2)=POS_REF(2)#6076. POS_REF(3)=POS_REF(3)#6076. SIGSNR_KEF=SIGSNR_RES(NREF) X(NSTATE)=SU(3,0) ZV=CEOT(ZREF,ZREF,NSAMF) BYTE FILE_REF,FILE_RES 700 NC=NKEF +1 + NCHAN X(I+1)=SV(1,1) , NCHAN, TRUNC, NSTATE, C 700 NREF=1,NCHAN-1 SNR_REF = SNR_RES (NREF) INCLUDE TRONTRL.CMN INCLUDE 'TRTINS.CMN' INCLUDE TRUORN, CMN TINT=NSAMF#DELTA_T DELTA_T=1./SR_REF Z([,])=0. Z(J,I)=0. 00 14 I=1,NSTATE B(I) = 0.COMPLEX ZRESBUF 00 5 I=0,NIGG GWGOL []=1.E30 10 14 J=1,I CONTINUE CONTINUE (L00F=0 GWG=0. 11=SJ g 5 10 4 စင 0015 0016 0017 0018 0032 0033 0034 0035 0037 0050 0008 0000 00100 0013 0019 0020 0021 0023 0026 0027 0028 0039 0040 0046 0047 0048 0053 2000 0000 0002 0000 0002 0014 0024 0025 0029 0030 0031 0036 0041 0042 0043 0044 0045 0049 0052 0054 0055 0056 0001 0002 0011

` à

BIAS(NC)=(1CDH(NC))##2#(1.+2.#CDH(NC)/NSAMF)/NSAMF SIGSGREF=(ALN10#CDH(NC)/(1.+10.##(SNR_REF/10.))#SIGSNR_REF)##2 SIGSGREF=(ALN10#CDH(NC)/(1.+10.##(SNR_RES(NC)/10.)) #SIGSNR_RES(NC))##2 SIGSGCOH(NC)=SIGSGREF+SIGSGRES 1.NSTATE 1.NSTATE	ZDU(N)=(00.) CONTINUE E=0. ZU=(00.) ZNU=(00.) TF=T_REF+NMIN/SR_REF CALL TRUFLAT(F.V.TF) TG=TF-ABSRUSM(1F1FOS-REF,3)/C ANDRRES=0. ZEXF=(10.)	1(TAUBAK,TW,TG,TF,FOS_REF,FOS_RES(1,NC),EFSILON,ERK) RF=1,NSAMF CALL TRUFDAT(F,V,TW) CALL RSMADD(U0,1,FF,-1,FOS_REF,3) CALL RSMADD(UN,1,FF,-1,FOS_REF,3) CALL RSMADD(UN,1,FP,-1,FOS_REF,3) D0=ABSRV(UN,3) D0=ABSRV(UN,3) D0=ABSRV(UN,3) CALL RVSCM(U0,1,FP,-1,FOS_RES(1,NC),3) D0=ABSRV(UN,3) CALL RVSCM(U0,1,FP,-1,FOS_RES(1,NC),3) CALL RVSCM(U0,1,FP,-1,FOS_RES(1,NC),5) CALL RVSCM(U0,1,FP,-1,FOS_RES(1,NC),5) CALL RVSCM(U0,1,FP,-1,FOS_RES(1,NC),5) CALL RVSCM(U0,1,FF,-1,FOS_RES(1,NC),5) CALL RVSCM(U0,1,FF,-1,FOS_RES(1,FF),5) CALL RVSCM(U0,1,FF,-1,FOS_RES(1,FF),5) CALL RVSCM(U0,1,FF,-1,FOS_RES(1,FF),5) CALL RVSCM(U0,1,FF,-1,FOS_RES(1,FF),5) CALL RVSCM(U0,1,FF,-1,FOS_RES(1,FF),5) CALL RVSCM(U0,1,FF,-1,FOS_RES	ZUU(NS)=ZUU(NS)+DZRESDITACONJG(ZREF(NRF)) ZUU(NS)=ZDU(NS)+DZRESDITACONJG(ZREF(NRF)) ZU=ZU+ZRES(NRF)*CONJG(ZRES(NRF)) ZU=ZU+ZREF(NRF)*CONJG(ZRES(NRF)) TAULAST=TAUBAR TG=TU+PELTA_T TF=TF+PELTA_T TF=TF+PELTA_T TF=TF+PELTA_T TF=CFQ_RES(NC)*(TAUBAR-TAULAST) FT=CFQ_RES(NC)*(TAUBAR-TAULAST) FTC=MOD(FT+1.D0) ZEXF=ZEXF*CEXF(TWOFLJ*FTC)
BIAS(NC)=(1C BIGSOREF=(ALN1 SIGSOREF=(ALN1 SIGSOREF=(ALN1 *SIGSORERES(NC SIGSOROH(NC)=S BIGSOROH(NC)=S 30 DO 40 N=1,NSTATE	ZDU(N)=(00 2DU(N)=(00 ZU=(00.) ZU=(00.) ZNUM=(00.) ZNUM=(00.) TF=T_REF+NMIN/SR_REF CALL TRUFEGAT(F.V.TF) TG=TF-ABSRUSM(1F ANORRES=0. ZEXF=(10.)	CALL TOT1(TAUBAK,TW,TG CALL TRUFDAT(F CALL TRUFDAT(F CALL TRUFDAT(F) CALL RSMADD(UN CALL RSMADD(UN,3) DN=ABSRV(UN,3) DN=ABSRV(UN,3) CALL RVSCM(UN,3) CALL RVSCM(UN,3) CALL RVSCM(UN,13) CALL RVS	200(NS 200(NS 200(NF 2000 20=2W+ZRE5(NRF 20=2W+ZRE5(NRF 20=2W+ZRE5(NRF 70=2W+ZRE5(NRF 70=10+2CA 70=100 70=100 71=10 700 7100 7100 7100

TRACN		28-May-1985 17:32:45 14-May-1985 12:14:28	VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELDRIDGEJTRDRIV1.FOR121	Pase 6
01115 01116 01117 01117 01117 01117 01117 01117 01117 01177 00000000	650 670 680 680 680 720 720 720	(NS))/ZW-2.*DREAL(ZDU(NS)/CONJG(2 *FX(J)*WF -2(I.K)*B(K) -2(I.K)*B(K) ABSB ABSB 642)+DX(I+NDE642) 642)+DX(I+NDE642)		
0144 0146 01447 01447 0151 0151 0151 0153 0153 0153 0153 0153	730 810 810	Sv(2,1)=X(1+WEG+2) CONTINUE Sv(3,0)=X(WSTATE) Sv(3,0)=X(WSTATE) GWGOLD=GWG GWGOLD=GWG ILOOF=ILOOF+1 IF(1LOOF+LE.MXLOOF)GO TO 10 IERR=1 KETUKN IERR=0 KETUKN END		

UISK#USER1:[[EL_DRIDGE][KUTILS1.FOR#2			
ли наз'тточ т/точ т/точ/// 15-0ct-1984 12:35:21 Анесентик т/т/тт ти т/т/с/ сл т/с/н бес/	SUBROUTINE TOTICTI.TW.TG,TO.FO.F1.EFSLN.ERR) THIS SUBROUTINE SOLVES FOR THE RECEIVER TIME. T1. ON CHANNEL 1. WHICH CORRESPONDS TO THE RECEIVER TIME.TO.ON CHANNEL O. THE CURRENT VALUE OF THE STATE VECTOR IS USED TO MAKE THESE CALCU- LATIONS. FO AND F1 ARE THE FOSTITON VECTORS OF STATIONS O AND 1. RESFECTIVELY. TG IS THE INITIAL GUESS AT THE INTERMEDIATE WATER TIME.TW. THE SUBROUTINE RETURNS T1.TW.AND ERR. THE VARIABLE EFSLN IS THE TOLERENCE IN SECONDS FOR THE CONVERGENCE CRITERION. THE SUBROUTINE WILL ITERATE AT MOST 20 TIMES IF CONVERGENCE IS NOT MET. IT WILL THEN SIGNAL WITH AN ERROR MESSAGE TO THE OFERATOR TERMINAL AND SET ERR = 1. OTHERWISE THE SUBROUTINE WILL RETURN WITH ERRINAL AND SET ERR = 1. OTHERWISE THE SUBROUTINE WILL RETURN WITH	•	TI = TW+61/C FETURN END
		00000 P	
	0001 0000 0000 0000 0011 00010 0011 00010 0011 00010 0011 00010 0011 00010 0011 00010 0000 0011 0000 0000 0000 0000 0000 0000 0000 0000	0014 0015 0015 0016 0019 0023 0023 0023 0023 0023 0023 0023 002	0000 940000 940000

7

.

-

• -

•

1.00

FUNCTION ABSRU(P.N)		THIS FUNCTION COMPUTES THE EUCLIDEAN NORM OF AN N-DIMENSIONAL	REAL VECTOR P		DIMENSION F(1)	ABSRV=0.	[0] 10 1=1,N	ABSRV=ABSRV+F(I)#F(I)	CONTINUE	ABSRU=SQRT(ABSRU)	R E T URN	END
	с С	сı	с С	сı					10			
0001	0002	0003	0004	0005	9000	0002	0008	0000	0010	0011	0012	0013

FUNCTION ABSRVSM(S1,F1,S2,F2,N)	THIS FUNCTION COMFUTES THE ABSOLUTE VALUE OF THE REAL	VECTOR	S1#E1+S2#E2	WHERE SI AND S2 ARE REAL SCALORS AND F1 AND F2 ARE REAL	N-THIMENSIONAL VECTORS.		DIMENSION F1(1),F2(1)	ABSRUSM=0.	PD 10 I=1,N	T=S1#F1(I)+S2#F2(I)	AFSKUSM=ABSKUSM+1 # T	CONTINUE	ABSRVSM=SQRT(ABSRVSM)	RETURN	END
ن ا	ں	ပ	ų	ပ	ပ	с С						10			
0001 0002	0003	0004	0005	9000	0007	0008	6000	0010	0011	0012	0013	0014	0015	0016	0017

••••••

_	-		-	-	•	•	•					۲
	IS A											
	IIFL											
	HULT FS F											
	LOK ECTO											
	THIS SUKROUTINE FERFORMS A REAL SCALOR MULTIFLY F=S1*F1 WHERE F*F1 ARE N-DIMENSIONAL REAL VECTORS AND 51 REAL SCALOR.											
	EAL REf											
î	A R DNAL											
1	ISNS ISNS	•										
F, S1	ERFC DIME	N F(1),F1(1) =1,N F(1)=S1#F1(1)										
SCM C).F1 51#F										
л У Ч	NUTI S1#1 ARI	F(1)=										
1 INE	UKRC F= F+F1 CALC		4									
SUBROUTINE RUSCH(F,S1,F1,N)	THIS SURFOUT F=S1 WHERE F.FI A REAL SCALOR.	The state of the s	LUNI LING RETURN END									
INS	TH WHE											
ť		، • د	01									

SUBRUUTING FLISUMTILNIUFIFILJIKLMAINNIN,NSMUFILLILLILAINNIN,NSMUFILLILLILLILLILLILLILLILLILLILLILLILLILL	THIS SUBROUTINE WRITES COMFLEX DATA TO FILES COMPATIBLE WITH	•	CUMPLEX AKKAY Z INTO NSAMP COMPLEX	SAMPLES STANTING WITH THE ANMIN'TH SAMPLE IN THE FILL.		THIS SUBROUTINE ASSUMES IRCMY COMPLEX DET'S PER RECORD.		THEFT I FORMER EV (7)			1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.	i	IF(IRCMX.GI.MAXZBUF)GO TO 30	CALL CLOSE (LN)	OFEN(UNIT=LN,FILE=DFTFIL,STATUS='UNKNOWN',RECL=IKCMX#2,ACCESS='DIRECT',	. BLOCKSIZE=IRCMX#8,ASSOCIATEVAKIABLE=IR)	I F R = 0	KCHX = I KCHX		1 5555 C = 111 11 1 1 15 2 15 5 1 1		IERCC (AMMAX-1.)/KCMX+IHEAUEK+I	a -	1=0	= I SKE	DO IO IREC=ISREC, IEREC	I & U F 1 = A M O F (A F A T A 1 - 1 . , R C M X) + 1	ALATA2=AMIN1((IREC-IHEALER)*RCMX,ANMAX)	IBUF2=AMOD(ADATA2-1.,RCMX)+1	IF(IBUF2-IRUF1+1.EQ.IRCMX)GO TO 4	FEAD(LN'IF,EFF=3)(ZRUF(I),I=1,IRCMX)	IR=IREC	IO 5 I=IRUF1,IBUF2	N1=N1+1	ZEUF(I)=Z(N1)	CONTINUE	WRITE(LN'IR,ERR=20)(ZBUF(I),I=1,IRCMX)	ADATA1=ADATA1+IBUF2-IBUF1+1	CONTINUE	CALL CLOSE(LN)		CALL TEXT(5, 'FTSDAT/FCTSDAT; ERROR WRITING FILE; ')	-		KN		TEXTI(5, CURRENT VALUE OF MAXZBUF = ',MAXZBUF)	TEXI(5)			
Ľ	ں ر	U	с o		ى د	ے د) ر	1				Ľ	1																			ň	4			ۍ ۱			10			20				30					
2000 0003	0002	0000	0007	8000	6000	01100	0012	2100	0014	0015	0016	0017	0018	0019	0020	0021	0022	0023	4000 4000	1000 1000		0027	0028	0056	0030	0031	0032	0033	0034	0035	0036	2037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047	0048	0045	0200	0051	0052	0013	000	8000	
		. ·		_	_															_							÷ .)	

BBGENT. THE FROGRAM RETURNS WITH THE FIRST NSAMF VALUES OF THE COMFLEX ARRAY Z LOADED FROM THE NSAMFLE IN THE FILE. THE FIRST SAMFLE IN THE FILE IS NUMBERED O TO THE USER SAMFLE IN THE FILE IS NUMBERED O TO THE USER THIS SUBROUTINE ASSUMES IRCMX COMFLEX DFT'S FER RECORD. IMFLICIT COMFLEX (2) DIMENSION Z(1), ZBUF (256) BYTE DFTFIL(1) DATA IHEADER/1/, MAXZBUF /256/ IF(IRCMX.6T.MAXZBUF /256/ IF(IRCMX.6T.MAXZBUF /256/ IF(IRCMX.6T.MAXZBUF /256/ IF(IRCMX.6T.MAXZBUF /256/ IF(IRCMX.6T.MAXZBUF /256/ IF(IRCMX.6T.MAXZBUF)60 TO 30 CALL CLOSE(LN) OFEN(UNIT=LN,FILE=ENFTLL,STATUS='OLD', RECL=IRCMX#2, ACCESS='DIRECT', BLOCKSIZE=IRCMX#8,ASSOCIATEVARIABLE=IR)
SAMFLES ST SAMFLE IN THIS SUBRO THIS SUBRO IMFLICIT C DIMENSION BYTE DFTFI DATA IHEAD IF(IKCMX.6 CALL CLOSE CALL CLOSE OFEN(UNIT= BLOCK
THIS SUBRO IMFLICIT C DIMENSION BYTE DFTFI DATA IHEAD IF(IKCMX.G CALL CLOSE CALL CLOSE OFEN(UNIT= BLOCK
THIS SUBRO IMFLICIT C DIMENSION BYTE DFTFI DATA IHEAD IF(IKCMX.G CALL CLOSE CALL CLOSE OFEN(UNIT= BLOCK
IMFLICIT C DIMENSION BYTE DFTFL DATA IHEAD IF(IKCMX.G CALL CLOSE OFEN(UNIT= BLOCK
DATA INENSION BYTE DETEN DATA INEAD IF(IKCMX.G CALL CLOSE OFEN(UNIT= BLOCK
BYTE DFTFI DATA IHEAD IF(IKCMX.G Call Close Ofen(UNIT= Block
DATA IHEAD IF(IKCMX.G CALL CLOSE OFEN(UNIT= BLOCK
IF(IKCMX.G CALL CLOSE OFEN(UNIT= BLOCK
IF (IKCHX.6 CALL CLOSE OFEN(UNIT= BLOCK
DFEN(UNIT= DFDCK BLOCK
BLOCK
IERR=0
RCMX=IRCMX
ANMIN=NMIN+1
ISREC=NMIN/RCMX+IHEADER+1
ANMAX=NMIN+NSAMF
IEREC=(ANMAX-1.)/RCMX+IHEADER+1
IR=ISKEC
AUATA1=ANMIN M1=0
IN IN INELTISNEL/IENEL
KEAU(LN'IK'EKK=20)(ZBUF(I),I=1,IKCMX) Ibirt==AMOD/ADAIA::
ΙΡΟΓΙ-ΜΗΟΟΥΜΡΗΙΗΙ-Ι.ΥΝΟΠΑ/ΤΙ ΔΠΔΙΔΟ-ΔΜΙΝΙ (ΙΤΦΕΓ.ΙΗΕΔΝΕΡΙΦΟΥΥ.ΑΝΜΑΥ)
TRUET-AMORIALIA INCOLO
TEULZ-ANULVHUNINZ- TAUFJ - TRUFJ - TRUFJ
CONTINUE
ADATA1=ADATA1+1BUF2-1BUF1+1
CONTINUE
CALL CLOSE (LN)
RETURN
CALL TEXT(5, ' GTSDAT/GCTSDAT: ERROR READING FILE:
CALL CLOSE (LN)
CALL TEXTI(5, GTSDAT/GCTSDAT:IRCMX EXCEDES MAXZBUF. IRCMX =
CHLL IEALL
ENI

.

8

٠. •

	SUBROUTINE FTSHDR(LN,DFTFIL,IRCMX,TDFTS,BINSZE,SSR,TSSR,
-	, BINFRQ, TDEL, FVEC, IERR)
	BYTE DFTFIL(1)
	DIMENSION FUEC(3)
ں د	
с С	BINSZE = BIN SIZE(HZ)
J	BINFKR = CENTEK FREQUENCY(HZ) OF I'TH BIN
с С	SSR = SFECTRAL SAMFLE RATE
с С	TSSR = TIME SERIES SAMFLE KATE(HZ)
U	IKCMX = COMFLEX RECORD SIZE (254)
υ υ	TIFTS= TOTAL # OF IFT COEFF.S (samples)
U U	TDEL = charnel delay
υ υ	FUEC(3) = FOSITION VECTOR OF SENSOR
IJ	
с С	
ں ا	
ں ا	
	CALL CLOSE(LN)
	OFEN(UNIT=LN,FILE=DFIFIL,STATUS='UNKNOWN',RECL=IRCMX#2,ACCESS='DIRECT',
÷	<pre>PLOCKSIZE=IRCMX#8,ASSOCIATEVARIABLE=IR)</pre>
ပ	
	IR=1
	I E R R = 0
	WRITE(LN'IR, ERR=900)IRCMX, TDFTS, BINSZE, SSR, TSSR, BINFRQ, TDEL, FVEC
	60 10 999
с С	
006	CALL TEXT(5,
	IERR=1) write error
666	CONTINUE
	CALL CLOSE(LN)
	RETURN

والمراجع والمراجع والمراجع والمتحر ومحاج والمراجع والمراجع والمراجع والمراجع والمراجع والمراجع والمراجع والمراجع

.

• •

.

à

.

·..;

à

CALL CLOSE(LN) OFEN(UNIT=LN,FILE=DFTFIL,STATUS='OLD',RECL=IRCMX#2,ACCESS='DIRECT', BLOCKSIZE=IRCMX#8,ASSOCIATEVARIABLE=IR) KEAD(LN'IR,ERR=900)IRCMX,TDFTS,BINSZE,SSR,TSSR,BINFRQ,TDEL,FVEC G0 T0 999 SUBROUTINE GISHDR(LN, DFTFIL, IRCMX, TDFTS, BINSZE, SSR, TSSR, BINFRQ, TMEL, FVEC, IERK) BYTE DFTFIL(1) TSSR = TIME SERIES SAMPLE RATE(HZ) IRCMX = COMFLEX RECORD SIZE (256) TDFTS= TOTAL # OF DFT COEFF.S (samples) R0 = CENTER FREQUENCY(HZ) OF I'TH BIN = SFECTRAL SAMFLE RATE FOSITION VECTOR OF SENSOR CALL TEXT(5, GISHDR: READ ERROR') i read error channel delay BINSZE = BIN SIZE(HZ) DIMENSION PUEC(3) CONTINUE CALL CLOSE(LN) RETURN ENL U ļ FUEC(3) BINFRO IERR=0 [ERR=1 $I \in I$ TUEL SSR . с 600 666 Ċ 0024 0025 0025 0028 0029 0029 0029 0030 0031

4	• • •	· • •
۵ ۵ ۵		
0 0		
ITS1		
IRUT		
VAX-11 FUKIRAM V3.2-37 DISK4USER11CELDKIDGEJTRUTILS1.FOR;2		
TI II -		
X S I J		
2-		
22121		
12:35:20		
28-May-1785 15-0ct-1984		
2 2 2 2 2 2		
AUTHOR		
H I N		
S II		
A IUMMY(AS IS		
1 1		
E S I		
аларана Маларана Алара Аларана Аларана Аларана Аларана Аларана Алара Ала Алара Алара Ала Алара Алара Ала Ала		
SUBROUTINE TRERF THIS SUBROUTINE Return End		
S S S S S S S S S S S S S S S S S S S		
SUBROUT THIS SL FND END		
000		
0000 0000 0000 0000 0000 0000 0000 0000 0000		
\$ \$ \$ \$ \$ \$		

. . . .

• •

. . . .

.

		······
£1		•
F ase		
1.FOF		
1115		
-37 EJTRL		
V3.2 K166		
TRAN : LELI		
VAX-11 FOKTRAN V3.2-37 DISN\$USER1:[ELDKIDGEJTRUTILS1.FOR#2		
AX - 11 ISA\$U		
12:32:21	SUBROUTINE DEDIH(D.TI.K) SUBROUTINE DEDIH(D.TI.K) INCLUDE TRENTRL.CHY COMMUNITERNTRL.CHY COMMUNITERNTRL.MEG-INSAMP.ISTRI.CHF.IEND.REF.NSXIF.MAXIT.EFSILON INCLUDE TRENKS/SUG.SIO.0FF.X(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLUDE TRENKS/SUG.SOILO.FFX(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLUDE TRENKS/SUG.SOILO.FFX(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLUDE TRENKS/SUG.SOILO.FFX(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLUDE TRENKS/SUG.SOILO.FFX(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLUDE TRENKS/SUG.SOILO.FFX(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLUDE TRENKS/SUG.SOILO.FFX(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLUDE TRENKS/SUG.SOILO.FFX(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).bB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(31).RHD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(33).RHORD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(33).RHORD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(33).RHORD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(33).RHORD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(33).RHORD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(33).RHORD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(33).RHORD.RHOSD.ZZ(961).ITCOUNT.TS INCLIDE TRENKS/SUG.SOILO.FFX(33).BB(33).RHORD	
	SUBROUTINE UPDTH(D,T1,K) SUBROUTINE UPDTH(D,T1,K) INCLUATINE (CMM) COMMON/TRENTELCM/ COMMON/TRENTELCM/ COMMON/TRUNE.MSAMF.15TET_REF.TENU_REF.NSKIF.MAXIT.EFSILON INCLUATINE (CMM) COMMON/TRUNE.CMM INCLUATIONCOMINAL) (EFCS) INCLUATINEORY, SV(3.0)10) FFX(31). kHOSA 22(961).ITCOU INCLUATINEORY, SV(3.0)10) FFX(31). kHOSA 22(961).ITCOU INCLE.2RMDEG41) THEN INCLE.2RMDEG41) THEN INCLE.2RMDEG41) THEN INCLE.2RMDEG41) THEN INCLE.2RMDEG41) FILS INCLE.2RMDEG41) FILS INCL	
28-May-1985 15-Oct-1984	61) • .	
28-May-1985 15-Oct.1984	MAXI	
άά ∎ Ω	SKIF SKIF	
	ана	
	END_F 31)_F	
	ド	
	SUBROUTINE DFTH(D,T1,K) DTHENSION D(3) NCLUDE TRENTEL,CMN COMMON/TRUNC,NSTATE,C NCHAN,TRUNC,NSTATE,C NCHAN,TRUNC,NSTATE,C NCLUDE TRUNC,NSTATE,C NCLUDE TRUNC,NSV(3,010),F,FX,BB,RHDS0 COMMON/TRUORK/SV(3,010),F,FX(31),FB NCX(31),X(31),COV(4,4),ETCOV NT=K-1 DOUBLE FRECISION 22,DX DOUBLE RECISION 22,DX DOUBLE RECISION 22,DX DOUBLE TRAKI DOUBLE TRUNC,SV(3,010),FF,FX(31),FB NT=K-1 DOUBLE TRUNC,SV(3,010),FF,FX(31),FB NCX(31),X(31),COV(4,4),ETCOV DOUBLE TRECISION 22,DX DOUBLE TRECISION 20,010),FF,FX(31),FB DO(2)=0,000 D(3)=0,000 D(3)=0,000 D(3)=0,000 D(3)=0,000 D(3)=0,000 D(3)=0,000 D(3)=1,00	
	NE DFDTH(D,T1.K) N D(3) TRENTEL,CMN RENTEL/NDEG,NSAMF,T RUNC,NSTATE,C TRUNC,S TRU	
	H H H H H H H H H H H H H H H H H H H	
	INE DFDTH(D TRCNTRL/NDEC TRCNTRL/NDEC TRUNC.NSTATE TRUNC.NSTATE TRUNC.NSTATE TRUNC.NSTATE TRUNC.NSTATE TRUNC.SUC	
	SUBROUTINE DFDTH(D,T1, DIMENSION D(3) INCLUDE 'TRCNTRL_NDEG.NS ,NCHAN, TRUNC,NSTATE,C INCLUDE 'TRUNC,NSTATE,C INCLUDE 'TRUNC,SV(3,011) ,DUBLE FRECISION ZZ,DX DOUBLE FRECISION ZZ,DX DO	
	0001 0005 0005 0005 0005 0001 0011 0011	

.

					an an an an an an an
12				 	••
9 19 19					
FOR\$					
- TS1 -					
RUTI					
2-37 16E J1					
LIRII LIRII	,				
KTRAN I : CEL					
L FOF					
VAX-11 FORTRAN V3.2-37 DISK\$USER1:CELDRIDGEJTRUTILS1.FOR#2					
24					
510					
17:33:20 12:35:21					
14-19 1-19					
28-Мач-1985 15-Oct-1984					
	IISINCDT=COS(X)/X-SIN(X)/X##2				
×	(IS-)				
	< e>				
A D C X C X C X C X C X C X C X C X C X C	S03ª				
INCE ISIO 0)TH NCDT	NCEIT				
FUNCTION DSINCDT(X) DOUBLE FRECISION DSINCDT,X IF(X.EQ.O.DO)THEN DSINCDT=0.DO	191				
ECTIO DELE X EQ	ELSE ENDIF Return End				
	ELSE ENDI Retui END				
1000	000000000000000000000000000000000000000				
	2000 9000 7000 8000				

.

۰,

FUNCTION SINC(X) DOUBLE PRECISION SINC,X If(X.EQ.0.D0)THEN SINC=1.D0 SINC=SIN(X)/X ENDIF Return End ELSE

28-May-1985 17:33:20 15-Oct-1984 12:35:21

Fage 11 VAX-11 FORTRAN V3.2-37 DISK\$USER1!CELDRIDGEJTRUTILS1.FOR#2

.

-

. .

.

•

~

2

<u>_</u>

~

- \

• .

·

...

10 Pase VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELDRIDGEJTRUTILS1.FOR#2 28-May-1985 17:33:20 15-Oct-1984 12:35:21 INCLUDE ' LELUKINGEJIRTIMS.CMN' Common/Irtims/File_ref(32),cf0_ref,bs2_ref,sr_ref,t_ref,f0S_ref(3) .fns_ref.snr_ref,sigsnr_ref ,FILE_RES(32,5),CFQ_RES(5),BSZ_RES(5),SR_RES(5),T_RES(5) ,FOS_RES(355),FNS_RES(5),SNR_RES(5),SIGSNR_RES(5),COH(5),BIAS(5) ,SIGSQCOH(5),ZRESBUF(0:4200),T_BUF syte file_ref,file_res COMPLEX ZRESBUF INCLUDE 'TELDRIDGEJTRCNTRL,CMN' COMMON/TRCNTRL/NDEG,NSAMP,TSTRT_REF,TEND_REF,NSKIP,MAXIT,EPSILON CALL GTSDAT(LN,FILE_RES(1,N),IRCMX,NMIN,NPTS,ZBUF,IERR) DOUBLE FRECISION F1, SAMFN, ARG, SIGMA, SINC, DSINCDT DATA F1/3.141592653589793D0/,LN/21/,IRCMX/256/ TRUNC2=2#TRUNC SIGMA=F1#SK_FES(N) DZXN=DZXN+SINC(AKG)#ZRUF(NN) DUZXNDT=DDZXNDT+DSINCDT(AKG)#ZBUF(NN) SUBROUTINE XN_DXN(TBAR,N,ZXN,DZXNDT) COMFLEX ZXN.FZXNDT,ZBUF(100) COMFLEX#16 FZXN.FDZXNDT ARG= (SAMPN-NS) #PI , NCHAN, TRUNC, NSTATE, C TIXNET=SIGMA*DEZXNET DDZXNDT=(0.,0.) DO 10 NS=NMIN,NMAX SAMPN=TU#SR_RES(N) NMIN=SAMFN-TRUNC+1 NMAX=NMIN+TRUNC2-1 TD=TBAR-T_RES(N) NFTS=NMAX-NMIN+1 I+NN=NN EIZXN=(0,,0,) NXZU=NXZ CONTINUE RE TURN Endi I = NN 10 0013 0015 0015 0015 0016 0017 0019 0019 0024 0025 0025 0021 0022 0023 0028 0030 0032 0033 0034 0035 0027 0036

•

•. •. •

ha

.

ġ

			an and and a state that shall sha	
Ċ,	р.			
1	ຍ ສ ພ			
	.51.F			
Ţ	, FUT IL			
	DGE J1			
	AN V3 Eldrig			
	VAX-11 FORTRAN V3.2-37 DISK\$USER1!CELDKIDGEJTKUTILS1.FOR#2			
	5K \$U2			
F				
	1120			
- 	17:33:20 12:35:21			
	1985			
	28-Мау-1985 15-Осt-1984 ОКS АND			
	28 152 ECTOR			
	NAL			
	SUM			
	28-May- 15-Oct 1.Z1.S2.Z2.N) FES THE VECTOR SUM COMFLEX N-DIMENTIONAL VECTORS AND SCALORS. SCALORS. 524Z2(1)			
	• 52•2 THE VI FLEX LORS.			
	51, Z1 JTES 2 2 2 2 2 2 2 2 3 1, Z1 1 3 2 2 1 3 5 2 5 1 1 5 1 5 1 2 1 5 1 2 1 2 1 2 1 2			
Ľ	0.0(2*) COMF(52.422 22.282 22.282 21.21 21.(1) 21.(1)			
•••	CSMAI JTINE 31421 81421 81421 81421 81421 71 22 71 72 71 727			
	TTINE UBROU S2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2			
	SUBROUTINE CSMAſŀŪ(Z+S1+Z1+S2+Z2+N) THIS SUBROUTINE COMFUTES THE VECTOR Z=S1421+S2422 WHERE 2,21+AND 22 ARE COMFLEX N-DIME S1 AND S2 ARE COMFLEX SCALORS. COMFLEX 2,21,22+S1+S2 D10 10 1=1,N D0 10 1=1,N Z(1)=S14Z1(1)+S24Z2(1) CONTINUE KETURN			
C.L.	200000 ⁹			
Š,	0001 0004 0004 0004 0005 0004 0001 0011 001			
Ľ	55555555555555555555555555555555555555			
L				

				•••••	A 19 19 19 19				•	•
	60									
	-									
	FOR									
Í										
	user									
	VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELURINGE]TRUTILS1.FOR;2									
	33									
e , i	17:33:20									
		52								
	28-May-1985 15-0ct-1984	AND								
	ar	0 S1								
	156-1	VECTDRS AND								
ÌÍ	İ	TORS								
	• •	SUBROUTINE RSMADD(F,S1,F1,S2,F2,N) THIS SUBROUTINE COMPUTES THE VECTOR SUM F=S1#F1+S2#F2 WHERE F,F1,AND F2 ARE REAL N-DIMENSIONAL ARE REAL SCALORS								
	-	DF 5 NSIC								
	L	P2.N VECT DIME	-							
		S21	2(I							
	• •	, F1, ES 1 REAL	2(1) S2#F							
	_	P,S1 HFUT ARE ARE	9.(1) +(1)							
	-	1 CO	F1 (
•		RSM4 111NE AND AND	(1),)=S1							
	•. •.	SURROUTINE RSMADD(F,S1,F1,S2,F2,N) THIS SUBROUTINE COMFUTES THE VECTO F=S1#F1+S2#F2 WHERE F,F1,AND F2 ARE REAL N-DIMEN ARE REAL SCALORS	DIMENSION F(1),F1(1),F2(1) DO 10 I=1,N F(I)=S1#F1(I)+S2#F2(I) CONTINUE							
		COUT S SUI S E F	INNI:	RE LUKN END						
		SUBF THIS WHEF	DO 1 CON1							
2	•									
		000000	2 2							
										:
	•	0001 0003 0004 0004 0005	0008	0013						
ند.		_								
ľ										
									•	•

~ Pase VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELDRIDGE]TRUTILS1.FOR;2 28-May-1985 17:33:20 15-0ct-1984 12:35:21 THIS FUNCTION COMPUTES THE COMFLEX DOT FRODUCT BETWEEN THE N-DIMENSIONAL COMFLEX VECTORS Z1 AND Z2 COMPLEX FUNCTION CEDT#16(21,22,N) COMFLEX Z1,Z2 DIMENSION Z1(1),Z2(1) CDDT=(0.,0.) DO 10 I=1,N r 0000 Ļ

CPOT=CPOT+Z1(I)*CONJG(Z2(I))

CONTINUE Return End

10

- 	Pade 6	-				-		-	
	VAX-11 FORTKAN V3.2-37 DISN\$USER1:EELDRIDGEJTRUTILS1.FOR;2								
	28-May-1985 17:33:20 15-Oct-1984 12:35:21	FUNCTION REDT(V1,V2,N)	THIS FUNCTION COMPUTES THE DOT FRODUCT BETWEEN THE REAL '	N-RIJENGTONNE AFCIONO AT 1995 AF	DIMENSION V1(1),V2(1)		Rb0T=Rb0T+V1(1)*V2(1)	CONTINUE Return	END
Ë		L L		ა ს				10	
		0001	5000	0002	9000	000	0000	0010	0012
ΪĊ.									

•

4 . ._

_

		• · · · ·	× • •
Ê	۵		
	យ ហ ល ជ.		
	版 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3		
	. F0		
	I CARACTER I CARACTERIZA		
	VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELDRIDGE]TKUTILS1.FOR#2		
	2.5 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7		
	E F C F C F C F C F C F C F C F C F C F		
	11 \$U\$		
	AX- ISK		
r.	24		
F	81		
	17:33:20		
	88 86		
	28-May-1985 15-Oct-1984 LY - Si IS A		
<u> </u>	28 28 3 21		
	ALDK MULTIFL VECTORS AND		
	ji s M		
	ECT		
	C A C		
	EX 12 LE		
	CO FI		
	Z1, SIG		
	S1. Merkan		
	(Z, FER -D1 -D1 -D1		
-	90 2 2 7 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2		
	28 SURROUTINE CVSCM(Z,S1,Z1,N) THIS SUBROUTINE FERFORMS A COMFLEX SCALOR MULTIFLY Z=S1*Z1 WHERE Z,Z1 ARE N-DIMENSIONAL COMZLEX VECTORS AND S COMFLEX SCALOR. COMFLEX SCALOR. COMFLEX Z,Z1,S1 DIMENSION Z(1),Z1(1) DO 10 1=1,N DO 10 1=1,N DO 10 1=1,N CONTINUE FETURN CONTINUE FETURN		
N.	SURROUT THIS SU WHERE 2 COMFLEX COMFLEX DIMENSI DO 10 1 CONTINU END END		
Ċ			
C	000000 9		
	0001 0000 0000 0000 0001 0001 0001 000		
Ľ			

تعقينا an Tarata <u>مت</u> ---G 19 Pase DISK#USER1: [ELDRIDGE]TRUTILS1.FOR;2 VAX-11 FORTRAN V3.2-37 28-May-1985 17:33:20 15-Oct-1984 12:35:21 SUBROUT INE TEXTI(5, CURRENT VALUE OF MAXBUF = ',MAXBUF) TEXT(5, IF YOU INCREASE MAXBUF,MAKE SURE YOU INCREASE THE') TEXT(5, CUMENSION OF BUF ACCORDINGLY.') OFEN(UNIT=LN,FILE=DFTFIL,STATUS='OLD',FECL=IRCMX,ACCESS='DIRECT', BLOCKSIZE=IRCMX#4,ASSOCIATEVARIABLE=IR) THIS SUBROUTINE OBTAINS REAL DATA FROM FILES GENERATED BY SUBROU PLTSDAT. THE FROGRAM RETURNS WITH THE FIRST NSAMF VALUES OF THE REAL ARRAY B LOADED FROM THE NSAMF REAL SAMFLES STARTING WITH THE ANMIN'TH SAMFLE IN THE FILE. THE FIRST SAMFLE IN THE FILE IS NUMBERED O TO THE USER CALL TEXTI(5, GRTSDAT:IRCMX EXCEDES MAXBUF. IRCMX = ',IRCMX) FER RECORD SUBROUTINE GRISDAT(LN,DFTFIL, IRCMX,NMIN,NSAMF,B,IERR) THIS SUBROUTINE ASSUMES IRCMX REAL DATA SAMPLES IBUF1=AMOD(ADATA1-1,,RCMX)+1 ADATA2≈AMIN1((IREC-IMEADER)‡RCMX,ANMAX) 0 READ(LN'IR,ERR=20)(BUF(I),I=1,IRCMX) CALL TEXT(5, ' GRISDAT: ERROR READING FILE: IBUF2=AMOD(ADATA2-1.,RCMX)+1 ADATA1=ADATA1+IBUF2-IBUF1+1 IEREC=(ANMAX-1.)/RCMX+IHEADER+1 E(N1)=BUF(I) Ċ IF(IRCMX.GT.MAXBUF)G0 T0 30 DATA IHEADER/1/,MAXBUF/256/ 5 I=IBUF1, IBUF2 I SREC=NMIN/RCMX+IHEADER+1 DIMENSION B(1)+BUF(256) N1=N1+ IREC=ISREC, IEREC Ľ; ANMAX=NMIN+NSAMF CONTINUE CALL CLOSE(LN) CALL CLOSE(LN) CALL CLOSE (LN) BYTE DFTFIL(1) ANMIN=NNIN+1 ALATA1=ANMIN 2 RCMX#IRCMX (R=ISREC CONTINUE RETURN **RETURN** IERR=0 IEKK=1 01 00 CALL CALL N1=0 CALL STOF END C = 2 = 2 = 2 t 10 80 0£ c 00000000000 ç ŝ 0017 0036 0037 0038 0049 0001 0002 0003 0004 0005 0005 0005 0008 0010 0011 0012 0014 0016 0020 0023 0026 0029 0030 0032 0034 0039 0040 0042 0043 0044 0045 0046 0047 0048 6000 0013 0019 0022 0024 0027 0028 0035 0041 0051 0052 0025 0031 Ļ

F'age VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELDRIDGEJTRUTILS1.FOR#2 CALL CLOSE(LN) OFEN(UNIT=LN,FILE=DFTFIL,STATUS='UNKNOWN',FECL=IRCMX,ACCESS='DIRECT', 28-May-1985 17:33:20 15-Oct-1984 12:35:21 TEXTI(5, CURRENT VALUE OF MAXBUF = '+MAXBUF) TEXT(5, IF YOU INCREASE MAXBUF,MAKE SURE YOU INCREASE THE') TEXT(5, DIMENSION OF BUF ACCORDINGLY,') CALL TEXTI(5, PRTSDAT:IRCMX EXCEDES MAXBUF. IRCMX = ',IRCMX) THIS SUPROUTINE WRITES REAL DATA TO FILES COMPATIBLE WITH Grisdat. The frogram writes the first nsamp values OF THE REAL ARRAY B INTO NSAMF REAL SAMFLES STARTING WITH THE ANMIN'TH SAMFLE IN THE FILE THIS SUBROUTINE ASSUMES IRCMX REAL SAMFLES FER RECORD. SUBROUTINE FRISDAT(LN.DFIFIL,IRCMX,NMIN,NSAMF,B,IERR) ADATA2=AMIN1((IREC-IHEADER)#RCMX,ANMAX) WRITE(LN'IR,ERR=20)(BUF(I),I=1,IRCMX) PLOCKSIZE=IRCMX#4,ASSOCIATEVARIABLE=IR) READ(LN'IR,ERR=3)(BUF(I),I=1,IRCMX) TEXT(5, ' PRTSDAT: ERROR WRITING FILE: IF(IBUF2-IBUF1+1.EQ.IRCMX)GO TO IBUF2=AMOD(ADATA2-1.,RCMX)+1 I BUF 1=AMOD(ADATA1-1.,RCMX)+1 ADATA1=ADATA1+IBUF2-IBUF1+1 [EREC=(ANMAX-1.)/RCMX+IHEADER+1 BUF(I) = B(NI)[F(IRCMX.GT.MAXBUF)G0 T0 30 DATA IHEADER/1/,MAXBUF/256/ 00 5 I=IBUF1, IBUF2 [SREC=NMIN/RCMX+IHEADER+1 N1=N1+1 DIMENSION B(1), BUF (256) 00 10 IKEC=ISREC, IEREC ANMAX=NMIN+NSAMP CONTINUE IR=IREC **DFTFIL(1)** CLOSE (LN) CLOSE (LN) **1+NIWN=NIMNA** ALATA1=ANMIN RCMX=IRCMX **IR=ISKEC** CONTINUE IERR=0 RETURN RETURN ERR=1 BYTE CALL CALL CALL CALL STOF N1 = 0 **UN** 0 E 10 20 C പ 000 **U** 00 C n Ċ m 0015 0016 0017 0018 0018 0013 00200 0023 0026 0027 0020 0035 0038 0003 0004 0005 0005 0005 0008 0000 0014 0032 0036 0039 00400 0042 0048 0011 0021 0024 0025 0028 4500 0045 0049 0200 0052 0053 0054 0001 0012 0037 0041 5400 0044 0046 0047 1500 0051

_

r,

| | | | | |

ê

(.

Ë

20

•__

-1 21 Page VAX-11 FORTRAN V3.2-37 DISK\$USER11FELDRIDGEJTRUTILS1.FOR#2 D 1 ÷ 28-May-1985 17:33:20 15-0ct-1984 12:35:21 ł THIS SURROUTINE FERFORMS COMPLEX HETRODYNING ON THE INFUT COMPLEX SERIES CONTAINED IN THE ARRAY Z. FHIS=SIGN(1.,ANMIN#FHIO)#AMDD(ABS(ANMIN#FHIO),1.) 25=CEXF(CMFLX(0.,TWOFI#FHIS)) ر ۲ FHI0=SIGN(1.,FK0)#AMOD(ABS(FR0)/SSR,1.) SUBROUTINE HET(Z+ANMIN+NSAMF+FKQ+SSR) IMFLICIT COMFLEX (Z) ZW=CEXF(CMFLX(0,,TWDFI*PHIO)) DATA TWOFI/6.283185307/ <u>(</u> IF(FRQ.EQ.0.)RETURN (() 10 N=1.NSAMF 2(N)=2(N)#25 DIMENSION Z(1) MZ*SZ=SZ CONTINUE RETURN END COMMAND RUALIFIERS E 10 0000 7 0019 0012 0013 0014 0015 0016 0018 0011 Ľ

FORTRAN /NOOBJ/LIS/SHOW=(INCLUDE,NOMAF) TRDRIV1,TRUTILS1

/WARNINGS /NOD_LINES /NOCROSS_REFERENCE /CHECK= (NOBOUNDS, OVERFLOW, NOUNDERFLOW) /DEBUG=(NOSYNBOLS,TRACEBACK) /STANDARD=(NOSYNTAX,NOSOURCE_FORM) /SHOW=(NOFREFROCESSOR,INCLUDE,NOMAF) /F77 /NOG_FLOATING /14 /OFTIMIZE

/NOMACHINE_CODE /CONTINUATIONS=19

COMFILATION STATISTICS

Run Time: 18.35 seconds Elarsed Time: 39.47 seconds Fade Faults: 23 Dynamic Memory: 164 rades

ļ.	-		、 、 ·	` • •	
Ę					
_	ی بو ب				
	•				
N	06 # 1				
	L . 				
	-37 E JQ/				
ן <u>ז</u>	3.2				
	L L L L L				
E	0KTK K1:C				
	.1 F(
	VAX-11 FORTRAN V3.2-37 Diskauseri:[Eldridgejûan.for;14				
	26				
•	0 4 1) 0				
	: 2 4 1 : : 5 6 : :				
÷.	28-May-1985 17:41:05 5-Jul-1984 14:56:40				
	198.				
	- 1 I-1				
	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0				
ļ	SUBROUTIME GAR(LUN,ITEXT,ANS) BYTE ITEXT(1),IZER DATA IZER/O/ DO 10 1=1,60 If(ITEXT(1),EG,0)GD TO 20 If(ITEXT(1),EG,0)GD TO 20 If(ITEXT(1),EG,0)GD TO 20 MRITE(LUN,30)(ITEXT(11),11=1,1-1),(IZER,I1=1,60) FDRMAT(GA1*0) READ(LUN,40,ERR=5)ANS FORMAT(G15.0) READ(LUN,40,ERR=5)ANS FORMAT(G15.0)				
	1=1				
	E R				
	• (12				
5	-1)				
(=1.				
	ХТ, А 20 7, 11				
	, ITE) TO) TO NNS				
Ĺ	ZER 260)60 1611 1621				
	SUBROUTINE GAR(LUN,ITEXT,ANS) BYTE ITEXT(1),IZER DATA IZER/O/ DO 10 1=1,60 If(ITEXT(1),EG.0)GO TO 20 If(ITEXT(1),EG.0)GO TO 20 URITE(LUN,30)(ITEXT(11),I1=1, FORMAT(60A1,§) READ(LUN,40,ERR=5)ANS FORMAT(G15.0) RETURN				
	11110 11110 11110 11110 11110 1110 111				
	00111 1110 1110 1121 1210 1210 1210 121				
	BUBR BATE DO DO DO DO DO DO DO DO DO DO DO DO DO				
		-			
	10 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0				
	6 888 6				
Ĕ.	0001 0000 0003 0005 0005 0005 0005 0005	12			
		0			
Ę					
-					

	2		
ĺ	ب م ه		
·. •	μ. 		
Ì	4		
Ì	0K 10		
Į	A. NAG		
2	• 2 - 3] 106 - 10		
	N N N N N		
?	VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELDRIDGE]QAN.FOR;14		
!	1 FO		
Ì	1 - X 4(
Î	17:41:05 14:56:40		
	141 741 75		
	989 984 98		
	28-May-1985 5-Jul-1984		
	5 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8		
P	1.60		
1	- 11		
1	IZER		
	4S) =1 • I -		
	XT.AI		
	, ITE 0 TO 1(1) ANS		
Ľ	SUBROUTINE GAX(LUN,ITEXT,ANS) BYTE ITEXT(1),IZER DATA IZEK/0/ DO 10 1=1.60 IF(ITEXT(1).EQ.0)GD TO 20 IF(ITEXT(1).EQ.0)GD TO 20 URITE(LUN.30)(ITEXT(11).11=1.1-1),(IZER,11=1.60) FORMAT(60A1.\$) FORMAT(60A1.\$) FORMAT(60A1.\$) FORMAT(26) FORMAT(26) REAULN		
	C(1): C(1):		
•	UTINE IZEXI IZEXI IZEXI IZEXI IZEXI IZEXI IZEXI ICEON		
	SUBROUTIN BYTE ITE) DATA IZEF DO 10 1=1 IF(ITEXT(URITE(LUN) FORMAT(6C READ(LUN) FORMAT(6C READ(LUN) FORMAT(20 RETURN		
	0 & C C H C 3 C K L K U		
Ė	40 000 4		
	0001 0003 0003 0004 0006 0006 0007 0006 0007 0006 0001 0011 001		
•			
ļ,			

.

. · ·

۰.

Sector			
cu cu			`•
Û			
9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	v		
÷14			
VAX-11 FORTRAN V3.2-37 DISK\$USER1:[EELDKIDGEJQAN.FOR;14			
ZAN.			
2-3) GE 1(
V3. DRID			
RAN LELI			
0RT 5R1:			
11 F \$USE			
JAX JISK			
10 10 10			
17:41:05 14:56:40			
28-May-1985 5-Jul-1984			
28-1 28-1			
	SUBRDUTIME GAI(LUN,ITEXT,IANS) BYTE ITEXT(1),IZER DATA IZER/O/ D0 10 1=1.60 If(ITEXT(1).EQ.0)GD T0 20 CONTINUE WRITE(LUN,30)(ITEXT(11),11=1.1-1),(IZER,11=1.60) FORMAT(60A1.\$) FORMAT(60A1.\$) FORMAT(60A1.\$) FORMAT(115) FORMA		
	=		
	ER, 1		
	21)		
	- 1 -		
	NS)		
	T, If		
,	TD TO NNS		
	SUBROUTINE GAI(LUN.ITEXT,IANS) BYTE ITEXT(1),IZER DATA IZER/O/ DD 10 1=1,60 IF(ITEXT(1).EQ.0)GD TD 20 CONTINUE URITE(LUN.30)(ITEXT(11),11=1,I FORMAT(60A1.\$) FORMAT(60A1.\$) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115) FORMAT(115)		
), IC EQ.0 (IT ERR= ERR=		
	50 110 100 100 100 100 100 100 100 100 1		
1 7	SUBROUTINE GAI BYTE ITEXT(1), UATA IZER/O/ DO 10 1=1,60 IF(ITEXT(1),EQ CONTINUE WRITE(LUN,30)(FORMAT(60A1,\$) FORMAT(115) FORMAT(115) RETURN END		
i.	2 2 2 2 H 0 2 F 2 F 2 F 2 F 2 F 2 F 2 F 2 F 2 F 2		
)			
1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		
•	H O M < D < D < D < O = O		
•	0001 0005 0005 0005 0005 0005 0005 0005		

20-10-1000 Internation Internatio Internation Internation Internati	}14 Fase			
28-Mav-1985 J141165 SUBFOUTINE GAALLWHITEXTIANS SUBFOUTINE GAALLWHITEXTIANS AT I TEXT(1):IIIA AT I TEXT(1):IIIIA AT I TEXT(1):IIIIIA AT I TEXT(1):IIIIIA AT I TEXT(1):IIIIIA AT I TEXT(1):IIIIIA AT I TEXT(1):IIIIIA AT I TEXT(1):IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII	J3.2-37 &IDGEJQAN.FOR			
SE-Mar-1985 171105 SUBFOUTINE GAA(LUN-ITEXT)ANS) SUBFOUTINE GAA(LUN-ITEXT)-18UF (40) MAIA IEX/O DI 01 151/0 IF(ITEX(1).EG.0)BD T0 20 IF(ITEX(1).EG.0)BD T0 20 IF(ITEX(1).EG.0)BD T0 20 RETECTIONE WITE (LOADIA) RETECTIONE RETECTIONE RETECTIONE DI 05 151.0CR DI 05 151.0CR DI 05 151.0CR RETECTIONE EDMAT(0,001) DI 05 151.0CR RETECTIONE RE	AX-11 FORTRAN			
Servas-1985 S-JJ-1984 BYTE IEKT(1): IZEK: IANS(1): IBUF (40) DI 01 01 11:00 IF(IEXT(1): ER.0)6D T0 20 CONTINE WITF(LUW: 30:(ITEXT(1): 11=1.1-1):(IZEK: 11=1.60) FRITF(LUW: 30:(ITEXT(1): 11=1.1-1):(IZER: 11=1.1-1):(IZER: 11=1.60) FRITF(LUW: 30:(ITEXT(1): 11=1.1-1):(IZER: 11=1.1-1				
		((LUN,ITEXT,IANS) ZER,IANS(1),IBUF(40) .0)60 TO 20 TEXT(11),I1=1,I~1),(IZER,I1=1,60) R=5)NCHR,(IBUF(NC),NC=1,NCHR) 	·	
		SUBROUTINE GAA BYTE ITEXT(1) DATA IZEK/0/ IQ IO I=1.60 IF(ITEXT(1).ER CONTINUE WRITE(LUN.30)(1) FORMAT(60A1.\$) FORMAT(60A1.\$) FORMAT(0,40A1) IANS(NCHR41)=0 DO 50 I=1.NCHR FORMAT(D)=IBUF(1) RETURN RETURN		

٠.

رم ا			
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1			
<u>u</u>			
VAX-11 FORTRAN V3.2-37 DISK\$USER1:EELDRIDGEJQAN.FOR:14			
28-May-1985 17:41:05 5-Jul-1984 14:56:40			
985 17 984 14			
May-19	۵		
28- 2+	(IZER,I1=I,60),ANS		
	11=1,¢		
	IZER,		
	<u>.</u>		
	SUBROUTINE TEXTR(LUN,ITEXT,ANS) BYTE ITEXT(1),IZER Data IZER/0/ 10 10 1=1,60 f(TEXT(1),EQ.0)GD TD 20 continue continue write(LUN,30)(ITEXT([1),1]=1,I-1), format(60a1,G12.4) return return		
	SUBROUTINE TEXTR(LUN,ITEX BYTE ITEXT(1),IZER Data IZER/O/ DO 10 1=1,60 TF(ITEXT(1),EQ.0)GO TO 20 Continue Continue Write(Lun,30)(ITEXT(11),I Format(60A1,G12,4) Return Return		
	ZER ZER 0)GD 0)GD .4)		
	TEXT (1),1 0/).EQ.].612].612		
	UTINE 11EXT 12ER/ 12ER/ 1=1, 1=1, 1=1, 1=1, 1=1, 1=1, 1=1, 1=1		
	SUBROUTIME TEXTR(LUM,ITEXT,ANS) BYTE ITEXT(1),IZER Data IZER/0/ D0 10 1=1,60 IF(TEXT(1),EQ.0)GO TO 20 CONTINUE WRITE(LUM,30)(ITEXT(11),11=1,I- FORMAT(60A1,G12,4) RETURN END		
	90 00 10 00 10 00		
	0001 0002 0000 0000 0000 0000 0000 0000		

۰.

4	
а Б С	
VAX-11 FORTRAN V3.2-37 DISK\$USER1:[ELDRIDGE]QAN.FOR;14	
28-May-1985 17:41:05 5-Jul-1984 14:56:40	St
28 5),(IZEK,I1=1,60),A
	SUBROUTIME TEXTK(LUN.ITEXT.ANS) BATE ITEXT(1).IZER Data IZEK/0/ D0 10 1=1.60 IF(ITEXT(1).ED.0)60 T0 20 CONTINUE MATTE(LUN.30)(ITEXT(11).11=1,1-1).(IZER.11=1.60).ANS FORMT(60A1.28.8) RETURN RETURN END
	SUBROUTINE TEXTX(LL BYTE ITEXT(1),IZER DATA IZEK/O/ DO 10 I=1,60 IF(ITEXT(1),EQ.0)60 CONTINUE WRITE(LUN,30)(ITEXI FORMAT(60A1,Z8,8)) RETURN END END
	000 000 000
	00010000000000000000000000000000000000

.

LT I UN II WHNF JOHT UNJOHI I UJONI USTA ALIGATES PART TAN A .

. . · · ۰.

- - - ł۵
- SUBROUTINE TEXT(LUN,ITEXT) BYTE ITEXT(1),IZER DATA IZER/O/ D0 10 1=1,72 IF(ITEXT(1),E0.0)G0 T0 20 CONTINUE WRITE(LUN,30)(ITEXT(11),11=1,1-1),(IZER,I1=1,72) FORMAT(72A1) RETURN END 0 0 0 0 0

<u> 2011 - El Constantino de Constantino de Constantino de Constantino de Constantino de Constantino de Constanti</u>

____ 7

SUBRDUTIME TEXTA(LUN.ITEXT.IANS) BYTE ITEXT(1).IZER.IANS(20) DATA IZEK/0/ TO 10 1=1.52 IF(ITEXT(1).EQ.0)GD TD 20 CONTINUE CONTINUE MATT(221) FORMAT(72A1) FORMAT(72A1) KETURN ENT 900 m ŝ

and a state of the second s

÷ 2 `_

.
 0002
 BYTE ITEXT(1).IZER

 0003
 BATA IZER/0/

 0004
 5
 IO 10 I=1.60

 0005
 IF(ITEXT(1).EQ.0)GO TO 20

 0006
 10
 CONTINUE

 00005
 10
 UNITNUE

 00006
 10
 CONTINUE

 00005
 10
 URITE(LUM.30)(ITEXT(11).11=1.1-1).(IZER.11=1.60).1ANS

 00008
 30
 FORMAT(6001.16)

 0009
 30
 FORMAT(6001.16)

 0010
 END
 END

.....

FILMED

END

9-85

DTIC