Technical Memorandum ERL WPL-19

COMPUTER PROGRAMS FOR MAXIMUM ENTROPY SPECTRAL ANALYSIS OF REAL AND COMPLEX SINGLE-CHANNEL TIME SERIES (WITH MICROFILM PLOTS)

Otto Neall Strand
Jessie M. Young
Russell B. Chadwick

Wave Propagation Laboratory
Boulder, Colorado
February 1977

noaa NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION / Environmental Research Laboratories

NOAA Technical Memorandum ERL WPL-19

COMPUTER PROGRAMS FOR MAXIMUM ENTROPY SPECTRAL ANALYSIS OF REAL AND COMPLEX SINGLE-CHANNEL TIME SERIES (WITH MICROFILM PLOTS),

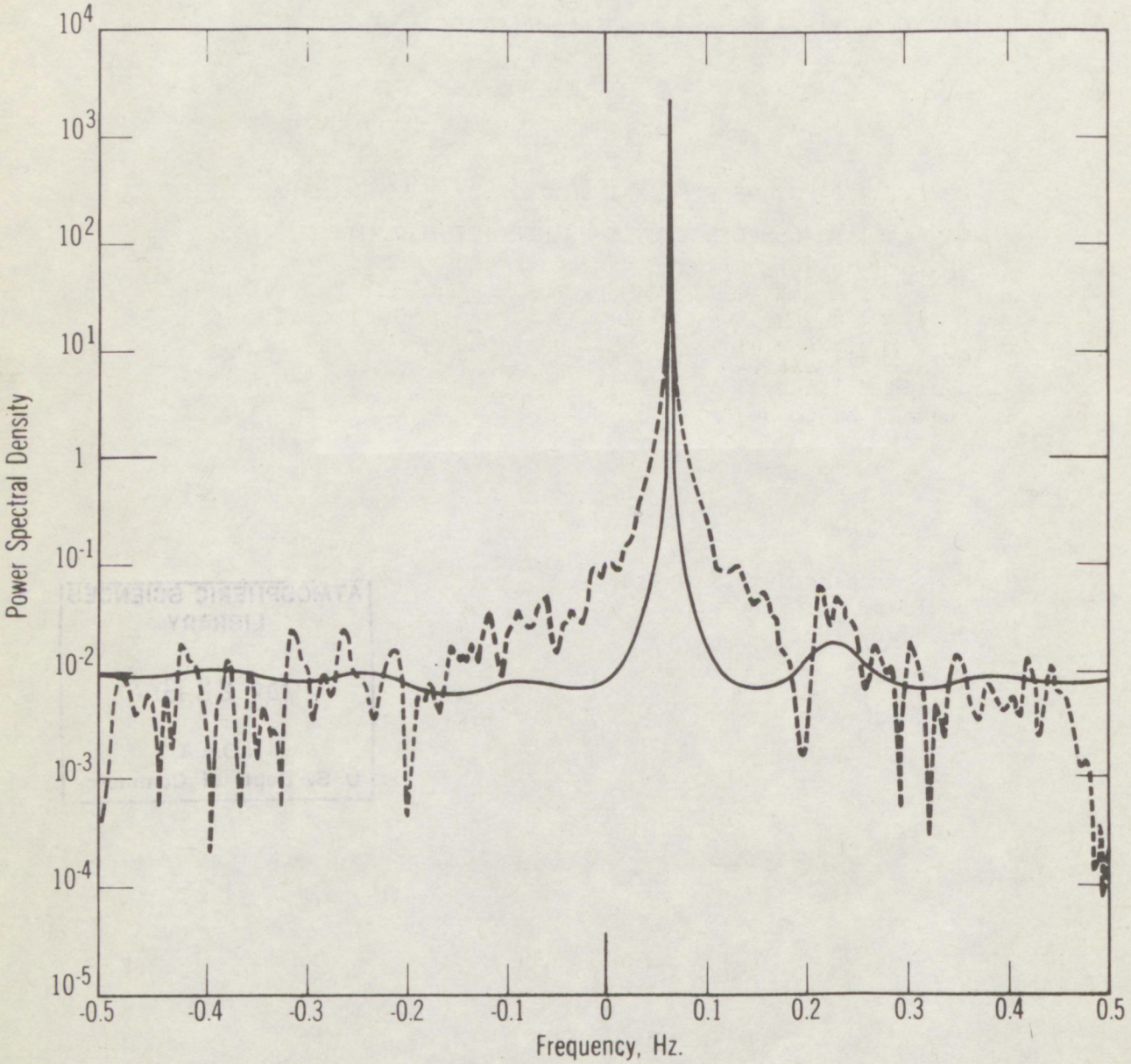
Otto Neall Strand
Jessie M. Young
Russell B. Chadwick

Wave Propagation Laboratory
Boulder, Colorado
February 1977

UNITED STATES
DEPARTMENT OF COMMERCE
Juanita M. Kreps, Secretary

NATIONAL OCEANIC AND
ATMOSPHERIC ADMINISTRATION
Robert M. White, Administrator

Environmental Research
Laboratories
Wilmot N. Hess, Director



Frontispiece. Comparison between Hanning-windowed FFT (dashed) and 7-lag maximum entropy spectra for the nearly monochromatic signal of section 2.

NOTICE

The Environmental Research Laboratories do not approve, recommend, or endorse any proprietary product or proprietary material mentioned in this publication. No reference shall be made to the Environmental Research Laboratories or to this publication furnished by the Environmental Research Laboratories in any advertising or sales promotion which would indicate or imply that the Environmental Research Laboratories approve, recommend, or endorse any proprietary product or proprietary material mentioned herein, or which has as its purpose an intent to cause directly or indirectly the advertised product to be used or purchased because of this Environmental Research Laboratories publication.

CONTENTS

CONTENTS (continued)

COMPUTER PROGRAMS FOR MAXIMUM ENTROPY SPECTRAL ANALYSIS OF REAL AND COMPLEX SINGLE-CHANNEL TIME SERIES (WITH MICROFILM PLOTS)

Otto Neall Strand, Jessie M. Young and Russell B. Chadwick
NOAA/ERL/Wave Propagation Laboratory
Boulder, Colorado 80302

ABSTRACT

We describe various FORTRAN computer programs for maximum entropy spectral analysis of a single real or complex time series. The descriptions are in sufficient detail to permit the methods to be correctly implemented. All programs have been carefully checked out, and are available from the authors on request.

1. INTRODUCTION

This report describes several computer programs written to implement maximum entropy spectral analysis and closely related algorithms. The main objective of these programs is to obtain an estimate of the power spectral density of a single-channel real or complex wide-sense stationary time series. A separate report (Strand, 1976) describes similar programs for multichannel time series. Because of the substantial saving in storage requirements and computing time, the single-channel methods have been implemented separately here instead of being considered as special cases of the multichannel methods. The main purpose of this report is to provide the user with enough information to enable him to implement the computations correctly. Therefore many mathematical details have been intentionally omitted. For these details one may consult Burg (1975), Kanasewich (1973), or Haykin and Kesler (1976). Useful general references on maximum entropy methods are Makhoul (1975), Ulrych and Bishop (1975) and Lacoss (1971). In this report many specific

details regarding the programs are deliberately omitted. If such specific details are desired, the user should consult a printout of the relevant source deck; in some cases COMMENT cards indicate programming options.

All maximum entropy spectral estimation algorithms follow essentially the same procedure. If the given detrended time series data (real or complex) are

$y_1, y_2, \cdots y_{N_d}$, with time increment Δt, then a near-optimum filter in the form

$$y_t + a_{1N}y_{t-1} + \cdots + a_{NN}y_{t-N} = e_N(t) \tag{1.1}$$

is obtained, where N is called the length of the filter (or number of lags), $e_N(t)$ is the output, and the numbers a_{kN} are called filter coefficients. If an appropriate filter (1.1) has been found, then the sequence $e_N(t)$ resembles white noise, and the maximum entropy power spectral density S(f) can be computed by the formula

$$S(f) = \frac{P_N \Delta t}{\left|1 + \sum_{k=1}^{N} a_{kN} z^k\right|^2} \tag{1.2}$$

where z is the complex number defined by

$$z = \exp(-2\pi i f \Delta t), \tag{1.3}$$

the positive number P_N is the power of the output $e_N(t)$, and f is the frequency in Hz, where $-1/(2\Delta t) \leq f \leq 1/(2\Delta t)$.

Criteria for the optimal filter length, N, have generally not been included. One such criterion, due to Akaike (1971), is given by the "final prediction error", FPE(N), where

$$FPE(N) = P_N \left(\frac{N_d + N + 1}{N_d - N - 1} \right) . \qquad (1.4)$$

In (1.4) the power P_N is a decreasing function of N and the quantity in parentheses is increasing. Ideally, one should choose N to minimize (1.4). The incorporation of (1.4), if desired, is generally left to the user; however a criterion essentially equivalent to (1.4) is computed in the program of section 3.2.

It can be shown that the optimal filter of length N satisfies the system

$$\begin{bmatrix} r_0 r_1 & \cdots & r_N \\ r_1^* r_0 r_1 & \cdots & r_{N-1} \\ \cdots & & \cdots \\ r_N^* r_{N-1}^* & \cdots & r_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{1N} \\ \cdot \\ \cdot \\ \cdot \\ a_{NN} \end{bmatrix} = \begin{bmatrix} P_N \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} \qquad (1.5)$$

where the star * denotes complex conjugation, $r_i = E\{y_t y^*_{t-i\Delta t}\}$, i = 0,1,...N, for which the expected value is taken over the ensemble of possible data sets. In our programs we denote r_i by R(I+1) because of constraints on FORTRAN subscripts. All filter computations used in our programs are recursive in nature and approximate the filter of length N in terms of the filter of length N-1. The coefficient $a_{NN} \equiv C_N$ in (1.1) has a special significance in that a_{1N}, a_{2N} ... $a_{N-1,N}$ may all be determined from it and the corresponding optimal filter of length N-1. In fact, the set r_1, r_2, ... r_N may be computed from the set r_0, C_1, ... C_N. Following Burg (1975), we call $a_{NN} \equiv C_N$ the N^{th} reflection coefficient.

Two distinct methods of fitting filters are used in our programs. The Burg reflection-coefficient method (Burg, 1975; Kanasewich, 1973) estimates the reflection coefficient C_N by a least-squares fit in which the sum of squares of residuals resulting from "forward" and "backward" filters is mini-

mized. As was noted above, all a_{kN} are then determined by the reflection coefficients, as are the r_k. In contrast, the R-method, sometimes called Yule-Walker estimation (Ulrych and Bishop, 1975) begins by first approximating all r_k that may be needed in the form

$$\tilde{r}_i = \frac{1}{N_d} \sum_{k=1}^{N_d-i} y_{i+k}\, y_k^*, \quad i = 0,1, \ldots, N_{max} \tag{1.6}$$

and then using a recursive method to solve (1.5) for $N \leq N_{max}$. Equation (1.6) defines a biased estimator of r_i; the corresponding unbiased estimator is obtained by replacing N_d by N_d-i in (1.6), i.e.,

$$\hat{r}_i = \frac{1}{N_d-i} \sum_{k=1}^{N_d-i} y_{i+k}\, y_k^*, \quad i = 0,1 \ldots, N_{max} . \tag{1.7}$$

Unfortunately, the estimate $\hat{r}_i$ may provide a coefficient matrix in (1.5) that is not positive definite, with dire consequences akin to obtaining negative variance. Most of our programs have error exits with diagnostic printout in case this occurs. It is a virtue of the Burg reflection-coefficient method that all powers are positive, or equivalently, that all coefficient matrices in (1.5) are positive definite. In the R-method, the use of (1.6) also provides positive power P_N, with some loss of resolution resulting from the bias. We usually use (1.6) with the R-method, but the programming provides the use of (1.7) as an option. As a further option involving the use of (1.7), we present a tapered entropy version of the R-method for real data (section 4). Thus three basic types of programs are presented here with various options. These implement, respectively, the Burg reflection-coefficient method, the R-method, and the tapered-entropy method for real data. These three methods differ only in the process that is used to calculate the filter coefficients; all utilize (1.2) to calculate the spectrum. Source decks are available for all programs described in this report, and all programs and subroutines have been carefully checked out. Because the set of programs described here is being used as a research tool, it is inevitable that some changes will occur. This report describes the programs as they now exist. Most programming is done using a relatively unsophisticated dialect of FORTRAN.

2. COMPLEX BURG REFLECTION-COEFFICIENT PROGRAMS

2.1 Program BURGA

Program BURGA is typical of these programs and will be described in detail. For this program we assume that N_d = NDAT complex values of the time series y_t = YT(I), I = 1, NDAT are given with time interval Δt = DT. We wish to compute and display spectra for filter lengths NSPE, NSP+1,...,NC. (We are thus enabled to "have a look" at spectra for several consecutive filter lengths, if desired.) Simplified flow charts describing BURGA are given in figs. 1 and 2, after which the function of each subroutine is described. A separate flow chart is shown for subroutine CALFIL because all functions except input and detrending are controlled by this subroutine. All subroutines in the program BURGA are explained in detail in order of their occurrence in the source deck.

2.1.1 BURGA, the Main Program

This program reads in the data and calls subroutines as indicated in fig. 1. The data input is described as follows:

NC	Maximum filter length for which spectrum is desired.
NSPE	Minimum filter length for which spectrum is desired.
NDAT	Number of data points.
DT	Time interval, real.
YT(K),K=1,...NDAT	Time series data, complex.

The READ formats can be modified by the user as desired. The present program uses card input.

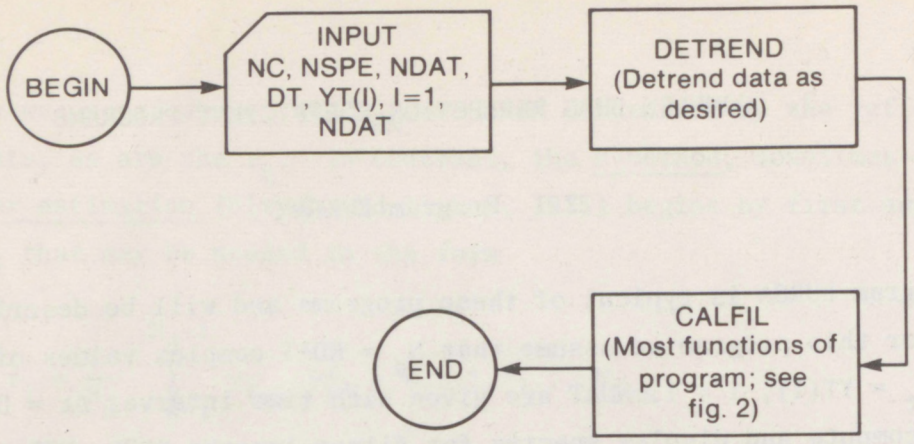


Figure 1. Simplified flow chart for the program BURGA.

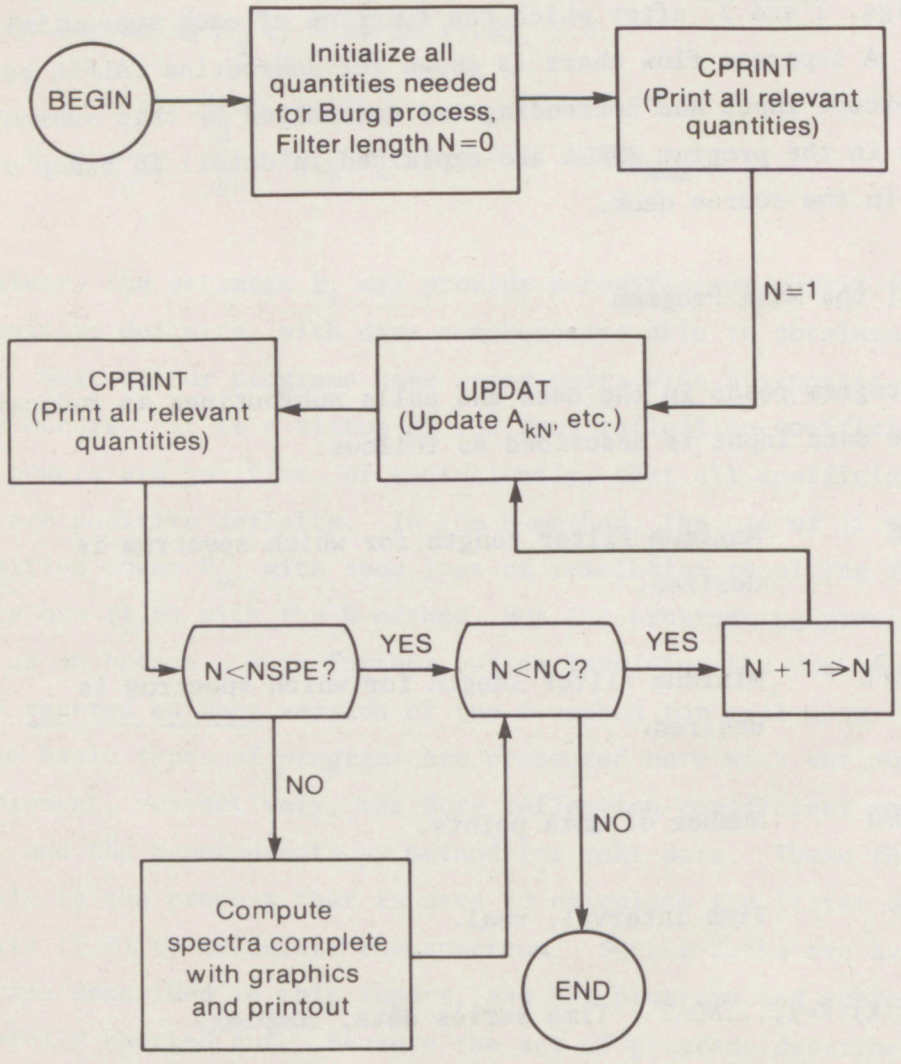


Figure 2. Simplified flow chart for the subroutine CALFIL.

2.1.2 Subroutine DETREND

This subroutine detrends the data in three possible ways. If the control cards ISKIP = 1 and IF(ISKIP.EQ.1)GO TO 30 are placed between COMMENT cards marked with repeated 1's, the data will not be detrended; if these control cards are placed between the 2's, the data will be reduced by subtracting the mean; finally, if these control cards are placed between the 3's a least-squares linear trend will be removed from the data. These options are also indicated by COMMENT cards in the source deck. The printouts of this subroutine are self explanatory.

2.1.3 Subroutine CALFIL

This subroutine performs most of the functions of the program, either by calling other subroutines or by its own statements. A simplified flow chart is given in fig. 2. All graphic subroutines are called from CALFIL except DDEND, which is called as the last step of the main program BURGA. (We refer here to plotting routines that apply to the CDC 6600 at Boulder, Colorado.) CALFIL utilizes two subroutines by J. A. Leise, namely AUTOPLT and MFRAME. A proper card to attach these routines must be included at the beginning of the deck. (Program BURGB and BURGE below have no such requirement, since no graphics are provided.) Computation of the main spectrum (eq. (1.2)) is done by Fast Fourier Transform (FFT), and the statement COMMON/FFDATA/ must be consistent in dimension with the corresponding statement in subroutine FFT. The present version also computes the spectrum near the peak and presents it graphically. This output is useful in checking out line spectra. The computation for this "fine spectrum" is by direct complex arithmetic, i.e., no FFT is involved. This option will probably be of less interest in analyzing actual data, and it is easily removed as indicated by COMMENT cards. All printout of this subroutine is self explanatory, as is the graphical microfilm output, shown in figs. 3 and 4 for the signal

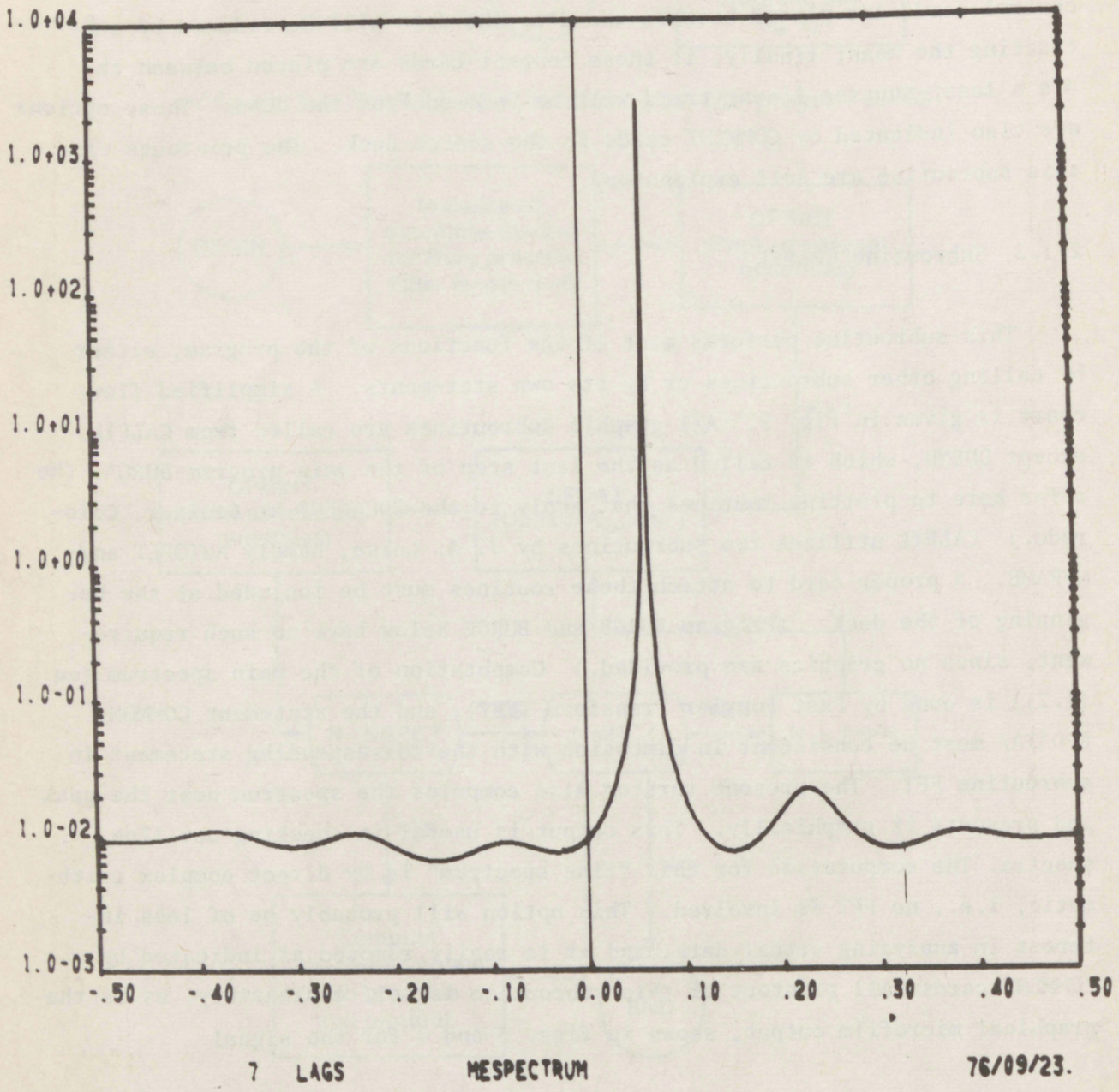


Figure 3. Graphical microfilm output for the program BURGA.

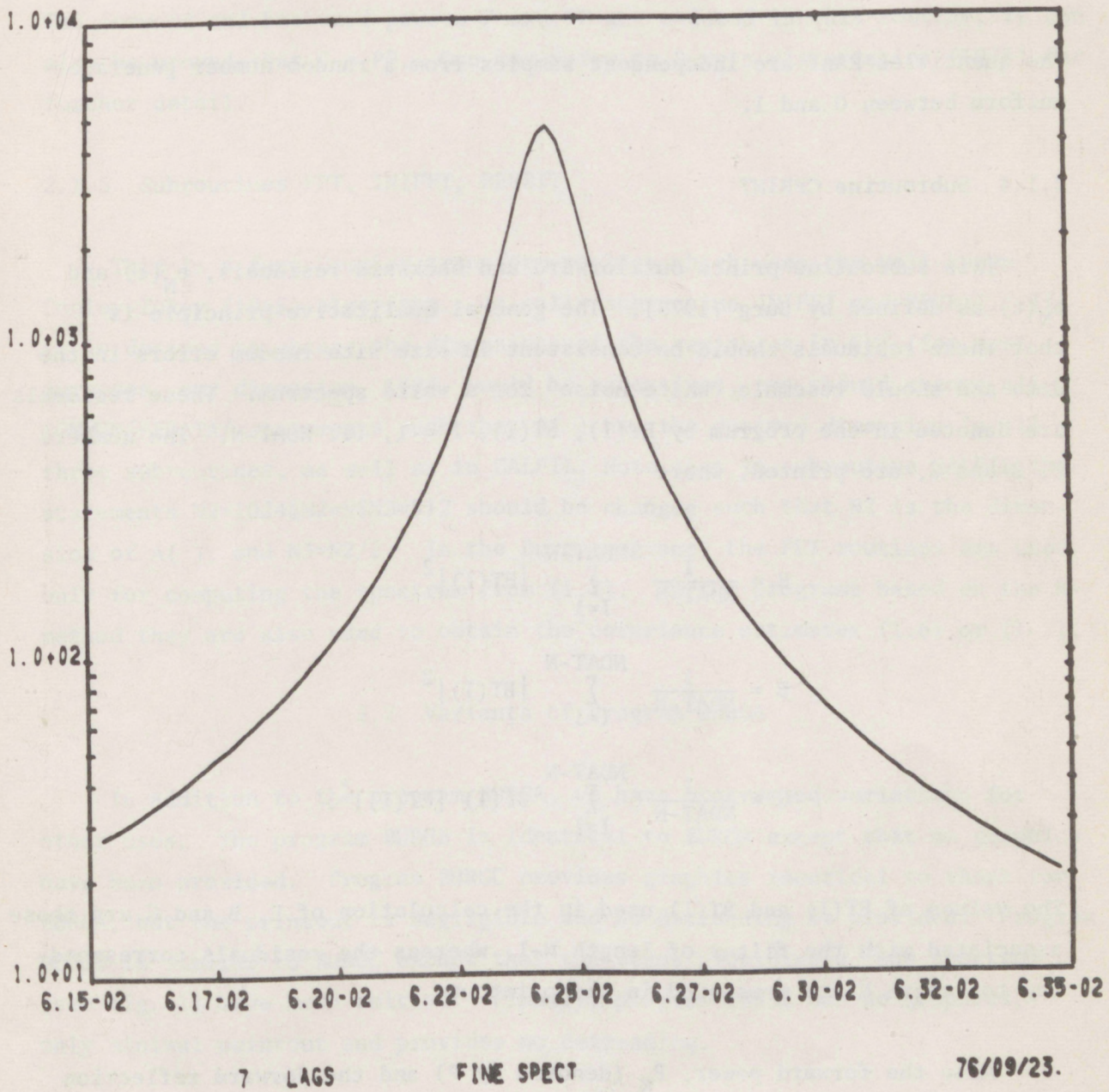


Figure 4. Graphical microfilm output for the program BURGA.

$$\mathrm{Re}(YT(I)) = \cos\left(\frac{2\pi I}{16}\right)+.25(\mathrm{Ranf}-.5);$$

$$\mathrm{Im}(YT(I)) = \sin\left(\frac{2\pi I}{16}\right)+.25(\mathrm{Ranf}-.5); \qquad (2.1)$$

$I = 1,2,\ldots 128;$ DT = 1 sec; NSPE = NC = 7.

The quantities Ranf are independent samples from a random number generator uniform between 0 and 1.

2.1.4 Subroutine CPRINT

This subroutine prints out forward and backward residuals, $e_N(t)$ and $b_N(t)$ as defined by Burg (1975). The general qualitative principle is that these residuals should be consistent in size with random errors in the data and should resemble "white noise" for a valid spectrum. These residuals are denoted in the program by ET(I), BT(I), I = 1, ... NDAT-N. The numbers E, B and G, are printed, where

$$E = \frac{1}{\mathrm{NDAT-N}} \sum_{I=1}^{\mathrm{NDAT-N}} |ET(I)|^2$$

$$B = \frac{1}{\mathrm{NDAT-N}} \sum_{I=1}^{\mathrm{NDAT-N}} |BT(I)|^2$$

$$G = \frac{1}{\mathrm{NDAT-N}} \sum_{I=1}^{\mathrm{NDAT-N}} BT(I)\cdot[ET(I)]^{*}.$$

The values of ET(I) and BT(I) used in the calculation of E, B and G are those associated with the filter of length N-1, whereas the residuals corresponding to length N are presented in the printout.

Also the forward power, P_N (denoted by P) and the forward reflection coefficient, $a_{NN} \equiv$ CN, are printed out. In the printouts, real and imaginary parts are calculated and then printed with real FORMAT statements.

2.1.5 Subroutine UPDAT

This subroutine calculates forward and backward reflection coefficients, CN and CB, respectively, and updates the filter coefficients a_{kN} (denoted by CF(K), K = 1,2, ... N in the program). The quantities CB(K) are backward filter coefficients and can be shown to be complex conjugates of the CF(K). The forward and backward powers P and PP are updated in this routine; it can also be proved that P = PP. One may refer to Burg's dissertation (1975) for further detail.

2.1.6 Subroutines FFT, INIFFT, REVBIT

This is a fast Fourier transform package which uses the well-known Cooley-Tukey (1965) algorithm. It calls subroutine INIFFT and REVBIT. If it is desired to change the dimensions of the variables in FFT (for most purposes, our dimension, 1024, would be excessive), one should change all COMMON/FFDATA/ statements such that A( ) has the desired dimension in all three subroutines, as well as in CALFIL. Note that in subroutine CALFIL the statements N2=1024$M2=9$N3=512 should be changed such that N2 is the dimension of A( ), and N3=N2/2. In the Burg programs, the FFT routines are used only for computing the spectrum from (1.2). In the programs based on the R-method they are also used to obtain the covariance estimates (1.6) or (1.7).

2.2 Variants of Program BURGA

In addition to the program BURGA, we have programmed variations for other uses. The program BURGB is identical to BURGA except that no graphics have been provided. Program BURGC provides graphics identical to those for BURGA, but the printout is negligible and no detrending is provided. Program BURGD is similar to BURGC except that the steps providing a fine spectrum (see fig. 1) have been omitted. Finally, program BURGE has no graphics, only minimal printout and provides no detrending.

3. PROGRAMS THAT IMPLEMENT THE R-METHOD FOR COMPLEX DATA; BURG REFLECTION-COEFFICIENT SUBROUTINE FOR REAL DATA

Although the programs described in this section and those of sections 2 and 4 were written by different programmers, they give equivalent results. The program SPEC implements the R-method for complex data by using the recursive procedure usually attributed to Durbin (1973), and the program MSPEC applies the Burg process as in BURGA to BURGE of section 2, except that the signal y_t is assumed real.

3.1 Program SPEC

In this program the covariances $\tilde{r}_k$, k = 1,N, (see (1.6)) are computed by FFT from the complex time-series data Z(I), I = 1,NN. Subroutine RGR then calculates reflection coefficients C_k, filter coefficients a_{kN}, and error powers P_N. The maximum entropy spectrum S(f) is computed by subroutine MEM, in which (1.2) is evaluated directly without using the FFT subroutine. It is also possible (with only slight modification) to calculate the maximum likelihood spectrum by using subroutines MLM and CMINV; the method used is that indicated by Lacoss (1971). A simplified overall flow chart describing SPEC is given in fig. 5, after which the function of each subroutine is described.

3.1.1 SPEC, the Main Program

This program reads the data, calls subroutines, and outputs data as required. The inputs are described as follows.

N	Number of filter coefficients to be calculated
K	Number of positive frequencies desired; here the total number of frequencies is 2K+1 and K is not related to FFT dimensions.

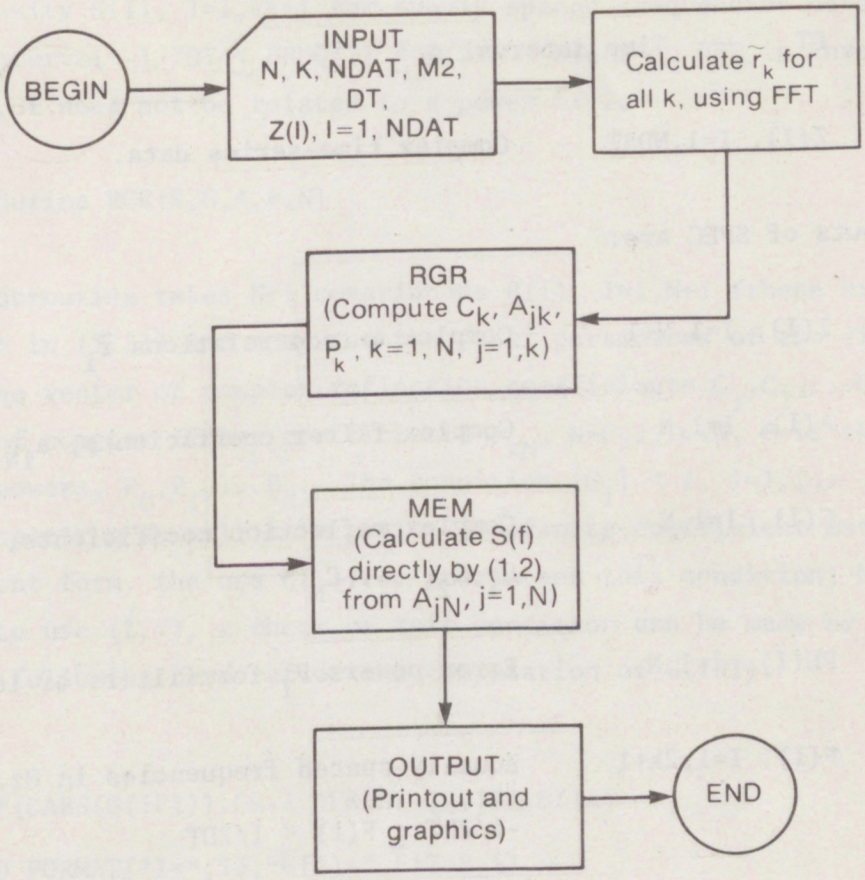


Figure 5. Simplified flow chart for the program SPEC.

NDAT — Number of data points

M2 — Power of 2 involved in FFT; the number of FFT frequencies used in calculating the r_k is N2=2**M2

DT — Time interval Δt

Z(I), I=1,NDAT — Complex time-series data.

The outputs of SPEC are:

Z(I), I=1,N+1 — Complex autocorrelations $\tilde{r}_i$

A(I), I=1,N — Complex filter coefficients, a_{iN}

G(I), I=1,N — Complex reflection coefficients, $C_1, C_2, \ldots C_N$

PE(I), I=1,N — Error powers P_I for filters of length I

F(I), I=1,2K+1 — Equally-spaced frequencies in Hz., where $-1/2DT \leq F(I) \leq 1/2DT$

SPECT(I), I=1,2K+1 — Power spectral density S(f) at frequency F(I).

This main program also utilizes the subroutine AUTOPLT, as mentioned in section 2.1.3.

3.1.2 Subroutines FFT, INIFFT, REVBIT

These subroutines are identical to those mentioned in section 2.1.6.

3.1.3 Subroutine MEM(A,FREQ,S,P,N,K)

This subroutine takes N filter coefficients A(I), I=1,N and the N^{th} power, P_N, as inputs and uses equation (1.2) for a direct computation of the spectral density S(I), I=1,2K+1 for evenly-spaced frequencies covering the frequency interval -1/2DT $\leq$ FREQ(I) $\leq$ 1/2DT. Since no FFT is involved in the computation, K need not be related to a power of 2.

3.1.4 Subroutine RGR(R,G,A,P,N)

This subroutine takes N+1 covariances R(I), I=1,N+1 (these are r_i, i=0,...N in (1.5)) and computes the other parameters of the filter. Here G denotes the vector of complex reflection coefficients $C_1,C_2,...C_N$, A is the vector of complex filter coefficients a_{kN}, k=0,1,...N, P is the vector of real error powers, $P_0,P_1,...P_N$. The condition $|C_i| < 1$, i=1,2,...N assures a positive spectrum as well as a positive definite coefficient matrix in (1.5). In our present form, the use of (1.6) guarantees this condition; however, if one wishes to use (1.7), a check of this condition can be made by the addition of the following cards after the computation of G(IP1):

```
IF(CABS(G(IP1)).GE.1.)PRINT 10,IP1,G(IA)
10 FORMAT(*I=*,I3,*G(I)=*,F12.8,1).
```

In addition, for Akaike's error criterion, a quantity approximately equal to 10 $\log_{10}$(FPE(N)), eq. (1.4), may be computed by adding the following cards:

C=1. (as the first executable statement)
E(IP1)=4.343*(ALOG(PE(IP1))+(2.*IP1)/N2) after the computation of PE(IP1), dimensioning E and passing it and N2 through the argument list. The approximation here is adequate when N2 is large and IP1/N2 << 1.

3.1.5 Subroutine MLM(R,FREQ,S,N,K)

This subroutine takes the vector R of N autocovariances, N and K as input and generates 2K+1 frequencies FREQ and the corresponding spectral densities S as computed by the maximum likelihood method (Lacoss, 1971). MLM calls CMINV, a complex matrix inversion routine. No graphical or printed output is provided; steps generating such output should be provided in the calling routine.

3.1.6 Subroutine CMINV

This is a complex matrix inversion subroutine that is required only if MLM is used.

3.2 Program MSPEC

This program implements the Burg reflection-coefficient method in a manner similar to that employed by the programs of section 2, except that the signal is assumed real. However, MSPEC has some options different from those of the other programs. A criterion equivalent to FPE(N), eq. (1.4), is computed and output if desired, and the overall output obtained is controlled by a "switch", as indicated below. A simplified flow chart describing MSPEC is given in fig. 6, after which the function of each subroutine is described.

3.2.1 MSPEC, the Main Program

This program reads the data, calls subroutines, and outputs data as required. The inputs are described as follows:

N	Number of data points
L	Number of filter coefficients to be calculated
MP	Power of 2 for use in FFT, where $NS=2^{(MP-1)}$ frequencies are desired for the spectrum

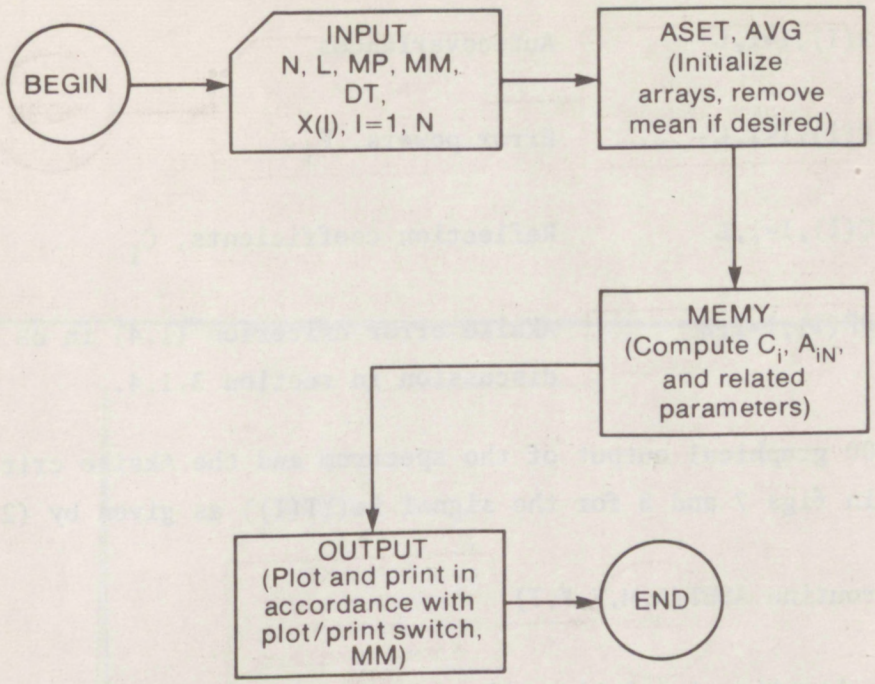


Figure 6. Simplified flow chart for the program MSPEC.

DT — Time interval Δt

MM — Plot/print switch
- MM=-1 plot only
- MM=0 print only
- MM=1 print and plot

X(I),I=1,N — Real time-series data.

The outputs of MSPEC are:

F(I),I=1,NS — Frequencies, evenly spaced from 0 to 1/(2DT)

S(I),I=1,NS — Spectral densities

A(I),I=1,L — Filter coefficients

R(I),I=1,L	Autocovariances
P(I),I=1,L	Error powers, P_i
C(I),I=1,L	Reflection coefficients, C_i
PE(I),I=1,L	Akaike error criterion (1.4) in dB. (See discussion in section 3.1.4.

The CDC 6600 graphical output of the spectrum and the Akaike criterion in dB are shown in figs 7 and 8 for the signal Re(YT(I)) as given by (2.1).

3.2.2 Subroutine ASET(A,M,V,K,T)

This subroutine initializes the vector A(I), I=1,M to zero, determines the mean of A or determines the mean of A and subtracts it from each A(I). For the latter outcome, this routine is entered as AVG(A,M,V,K,T).

3.2.3 Subroutine MEMY(X,A,P,NDAT,L1,R,C,PE)

This subroutine computes the filter parameters recursively up to filter length L1 from the real input time-series data X(I), I=1,NDAT. The parameters A(I),P(I),R(I),C(I) and PE(I) are as defined in the description of MSPEC (sec. 3.2.1) above. The subroutine returns the value L_m of L1 for which the Akaike criterion PE achieves its minimum within the range $1 \leq L_m \leq L1$. If $L_m < L1$, then it can be inferred that the best statistical representation of the process involves a filter of length $L_m < L1$. However, the value of L_m has no effect on the computation. Subroutine MEMY is available as a separate subroutine that can be used as the main element in constructing other maximum-entropy programs. It is concise and is easily called by other subroutines.

3.2.4 Subroutines FFT,INIFFT,REVBIT

These subroutines are identical to those mentioned in section 2.1.6.

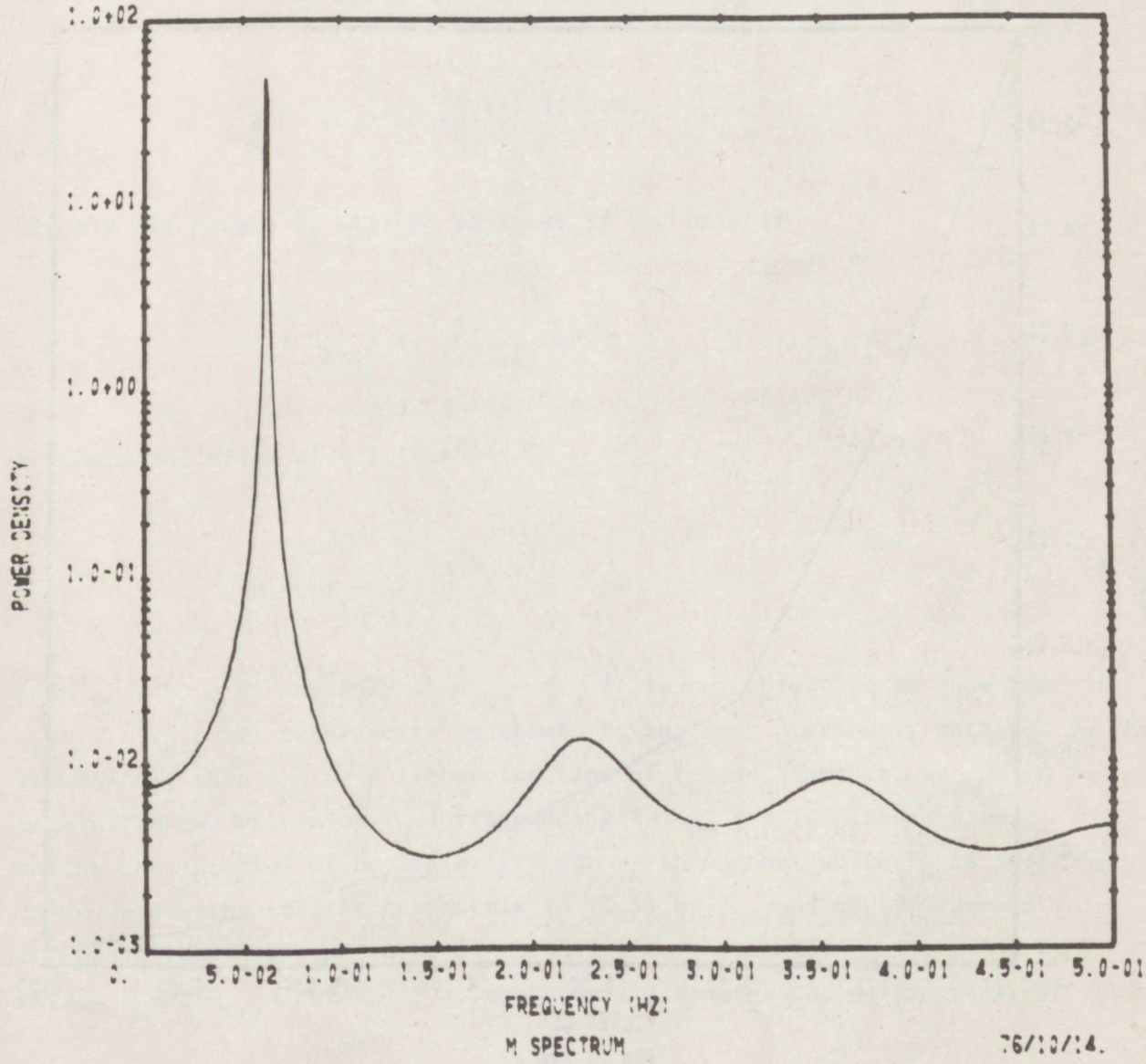


Figure 7. Graphical microfilm output for the program MSPEC, N=7.

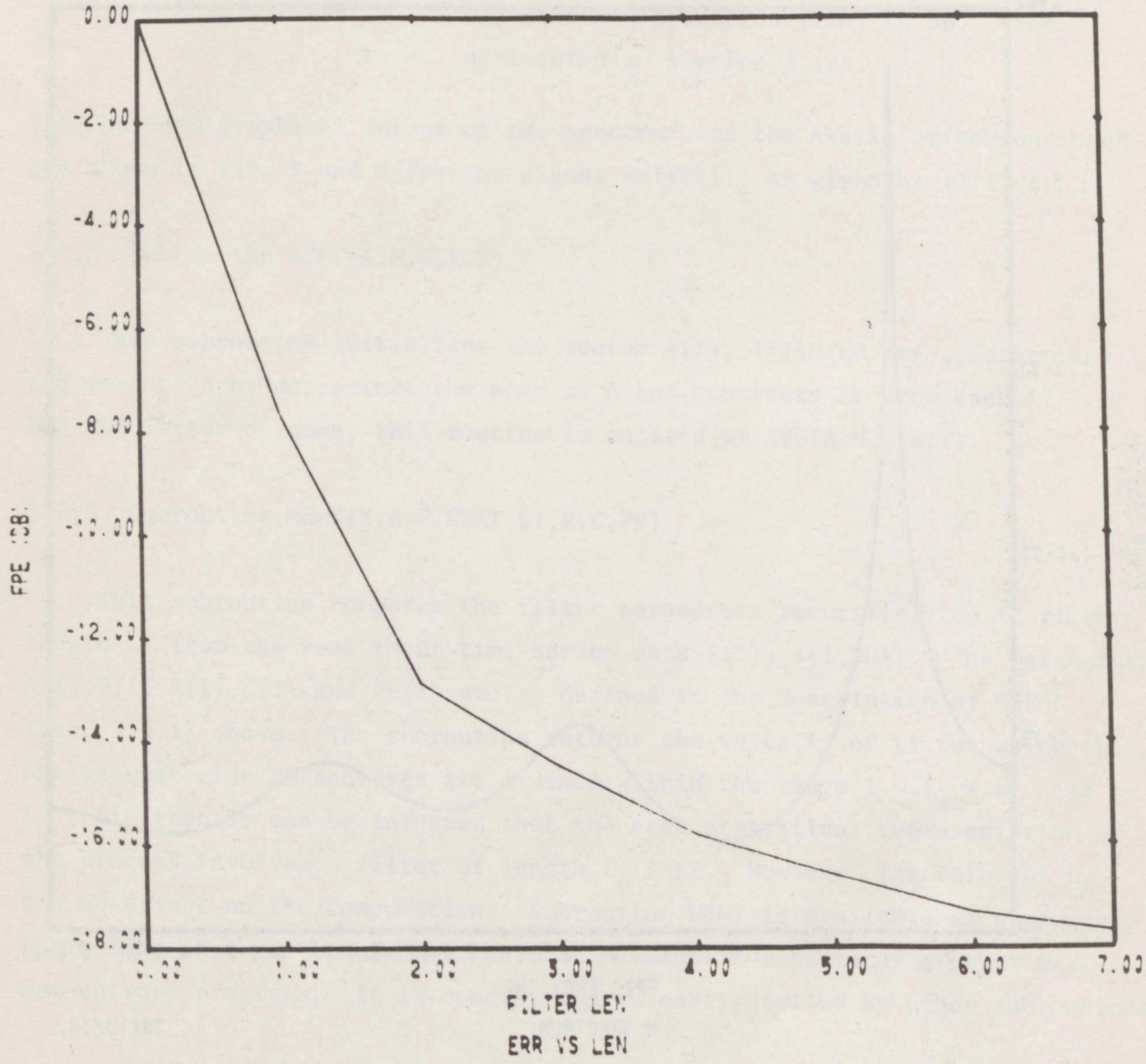


Figure 8. Graphical microfilm output for the program MSPEC.

4. TAPERED ENTROPY PROGRAMS FOR REAL DATA

4.1 Theoretical Considerations

We present sufficient theory here to enable the user to understand the concept of tapering the entropy. It is known (Burg, 1975; Makhoul, 1975; Durbin, 1973) that for a real time series

$$P_k = P_{k-1}(1-a_{kk}^2), k=1,2,\ldots N. \tag{4.1}$$

Clearly all powers P_k will be positive if and only if

$$-1 < a_{kk} < 1, \quad k=1,2,\ldots N. \tag{4.2}$$

Furthermore, if $|a_{kk}| < 1$, then

$$P_0 \geq P_1 \geq \ldots \geq P_N \geq 0 \tag{4.3}$$

with $P_k = P_{k-1}$ if and only if $a_{kk} = 0$. It is our objective to show that if $r_0, r_1, \ldots r_{N-1}$ are regarded as constant (having been previously obtained in the recursion), then a_{NN} is a linear function of r_N and, further, we desire to obtain the two values of r_N corresponding to $a_{NN} = \pm 1$. These values define the interval of admissibility for r_N throughout which P_N is positive. If we denote the coefficient matrix in (1.5) by R_N, and apply Cramer's rule to obtain the first component of the solution vector $(1, a_{1N}, \ldots a_{NN})^T$, we obtain $P_N \det(R_{N-1})/\det(R_N) = 1$, where det( ) denotes the determinant, so that

$$P_N = \det(R_N)/\det(R_{N-1}), \quad N=1,2,\ldots \quad . \tag{4.4}$$

It follows from (4.4) and the leading principal minor theorem (Mirsky, 1961), that R_N is positive definite if and only if $P_0, P_1, \ldots P_N$ are all positive. If we regard $r_0, r_1, \ldots r_{N-1}$ as constants, then $\det(R_N)$ (and consequently P_N) is a quadratic function of r_N (see (1.5)). If $a_{NN} = 0$, then the first N equations of (1.5) reduce to a corresponding system (1.5) with N replaced by N-1, for which the unique solution is $(1, a_{1,N-1}, \cdots a_{N-1,N-1})^T$. By the (N+1)st equation of (1.5), the value of r_N for which $a_{NN} = 0$ is r_N^C, where

$$r_N^C = - \sum_{k=1}^{N-1} a_{k,N-1} r_{N-k}. \tag{4.5}$$

This expression gives the value of r_N corresponding to maximal power, for which $P_N = P_{N-1}$. By applying Cramer's rule to (1.5) to obtain the last component, a_{NN}, of the solution vector, it is found that a_{NN} contains r_N linearly, and in fact that

$$a_{NN} = \frac{(-1)^N P_N}{\det(R_N)} \begin{vmatrix} r_1 r_0 & \cdots & r_{N-2} \\ r_2 r_1 & \cdots & r_{N-3} \\ \cdots & & \cdots \\ r_N r_{N-1} & \cdots & r_1 \end{vmatrix}$$

$$= \frac{(-1)^N P_N}{\det(R_N)} \left\{ \begin{vmatrix} 0 & r_0 & \cdots & r_{N-2} \\ 0 & r_1 & \cdots & r_{N-3} \\ \cdot & & & \\ \cdot & \cdots & & \cdots \\ \cdot & & & \\ 0 & r_{N-2} & \cdots & r_0 \\ r_N & r_{N-1} & \cdots & r_1 \end{vmatrix} + \begin{vmatrix} r_1 & r_0 & \cdots & r_{N-2} \\ r_2 & r_1 & \cdots & r_{N-3} \\ \cdots & & & \cdots \\ r_{N-1} & r_{N-2} & \cdots & r_0 \\ 0 & r_{N-1} & \cdots & r_1 \end{vmatrix} \right\}$$

$$= - \frac{P_N}{\det(R_N)} \left[r_N \det(R_{N-2}) + C \right] = - \frac{1}{P_{N-1}} \left[r_N + \frac{C}{\det(R_{N-2})} \right]. \tag{4.6}$$

For N=1 we obtain $a_{11} = -r_1/P_0$ with $r_1^C = 0$. In (4.6), which we have derived using (4.4), C denotes a function constant with respect to r_N. Setting $a_{NN} = 0$ shows that $r_N^C = - C/\det(R_{N-2})$. It follows from (4.6) that $|a_{NN}| < 1$, and consequently $P_N > 0$ if and only if

$$r_N^C - P_{N-1} < r_N < r_N^C + P_{N-1} . \tag{4.7}$$

For a given set of $r_0, r_1, \ldots r_{N-1}$ defining a positive definite matrix R_{N-1} we call a value of r_N admissible if it gives rise to positive power P_N, and the interval defined by (4.7) is called the interval of admissibility for r_N. Because this interval has length $2P_{N-1}$, moving r_{N-1} closer to r_{N-1}^C will widen the interval of admissibility for r_N, since it increases P_{N-1}.

We outline a tapering procedure for use with $\hat{r}_i$ as given by (1.7). The procedure is not particularly valuable with biased estimates $\tilde{r}_i$ as in (1.6), but such an option is provided in the program. The standard maximum entropy procedure, using the R-method with the $\hat{r}_i$, fits a filter by (1.5), thereby forcing a maximum entropy or optimal extension (von den Bos (1971)) for which $r_{N+1}=r_{N+1}^C$, $r_{N+2}=r_{N+2}^C$, ... This extension is a consequence of terminating the filter at length N as in (1.1). The procedure, in effect, treats the given data $\hat{r}_0, \hat{r}_1, \ldots \hat{r}_N$ as error free and assumes that $\hat{r}_{N+1}, \hat{r}_{N+2}, \ldots$ are so erroneous as to be worthless. This treatment is unrealistic because the sampling errors in the $\hat{r}_N$ tend to become larger relative to the interval of admissibility as N is increased and the increase in relative error is gradual rather than abrupt. This effect is caused mainly by the decrease in P_{N-1} as filters of increasing length are fitted to the data, resulting in a narrower admissibility interval (4.7). We may take account of this type of error behavior by the following device. For each N we replace $\hat{r}_N$ by a convex linear combination or weighted mean given by

$$r_N^{(\alpha)} = \alpha \hat{r}_N + (1-\alpha) r_N^C, \quad 0<\alpha<1 \tag{4.8}$$

$$N=1,2, \ldots .$$

The implementation of (4.8) is recursive; we replace $\hat{r}_1$ by $r_1^{(\alpha)}$, compute the filter of length 1, then replace $\hat{r}_2$ by the resulting $r_2^{(\alpha)}$ (calculated using r_2^C as derived from $r_1^{(\alpha)}$), ... etc. The parameter α may be tapered as indicated in fig. 9. As an example, α may be given by a curve such as the solid curve, in which it is assumed that $\hat{r}_1$ is nearly free from error relative to the admissibility interval (-1,1), and that $\hat{r}_{15}$ and subsequent $\hat{r}_N$ have sufficiently large errors as to be completely worthless relative to the intervals $(r_N^C - P_{N-1}, r_N^C + P_{N-1})$, $N \geq 15$. Note that the conventional maximum entropy R-process would assume α as given by the dashed step function in fig. 9. Our tapered entropy programs implement a linear taper as given by the dotted line in fig. 9. Other tapers are easily implemented by alterations in subroutine TAPER, described in section 4.2.7.

In fig. 10, typical admissibility intervals are shown for wind velocity data in which a space coordinate plays the role of time. The original unbiased covariances (given by (1.7)) are shown as circles, and the α-modified covariances $r_k^{(\alpha)}$ are shown as crosses. Note that the admissibility interval is widened by the taper, and that the $r_k^{(\alpha)}$ remain admissible

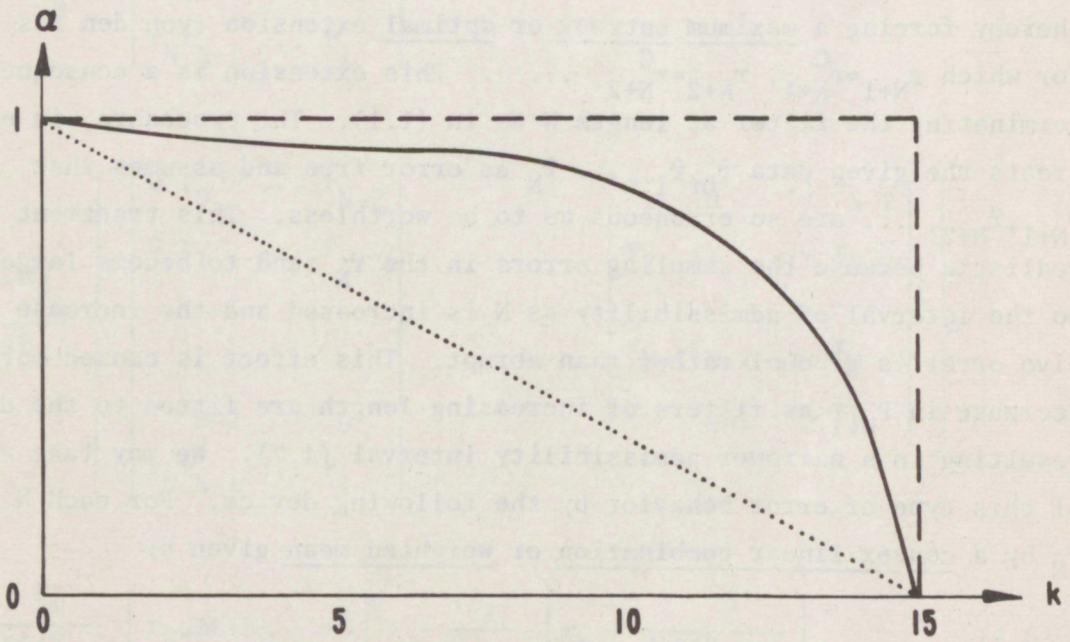


Figure 9. Curves relating α to the number k of filter coefficients.

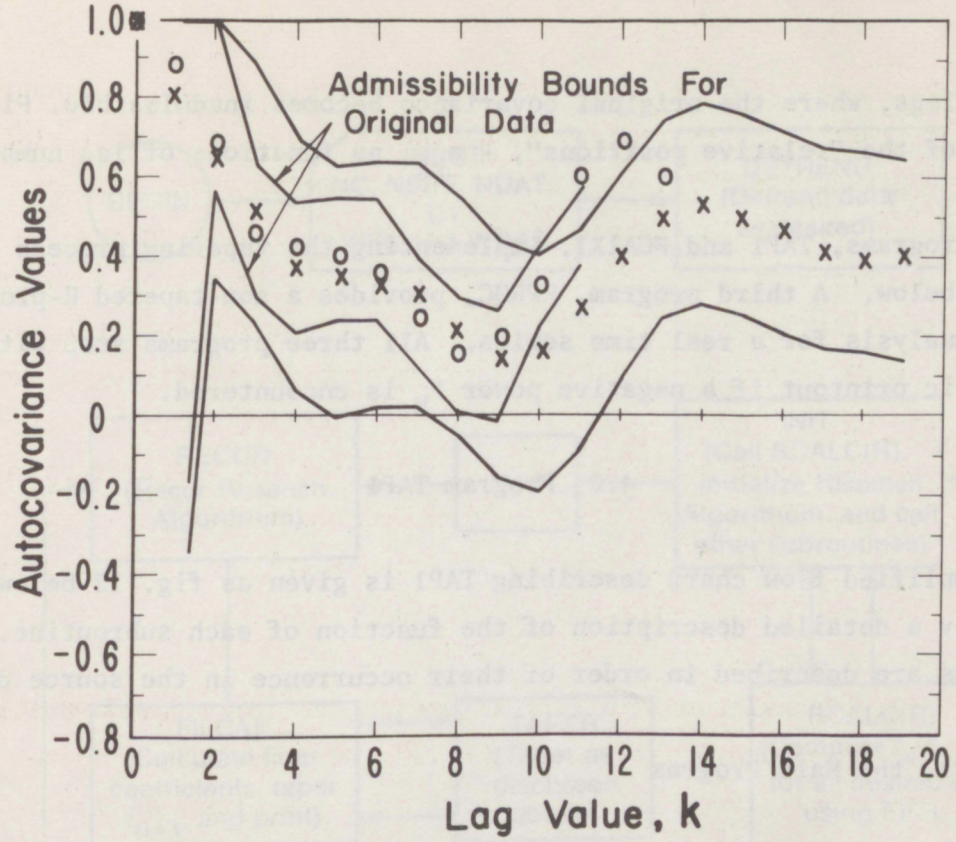


Figure 10. Autocovariances and admissibility intervals.

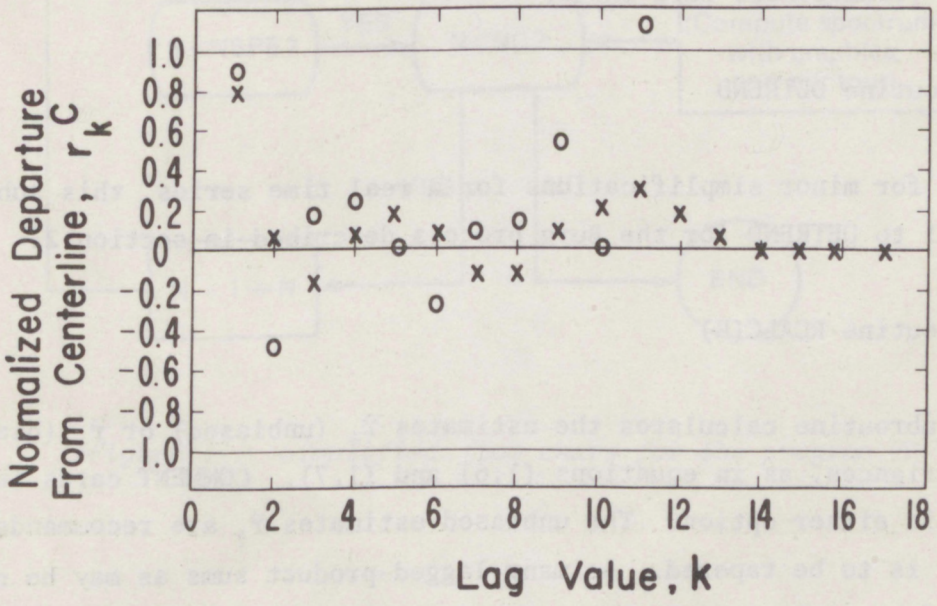


Figure 11. Relative positions of autocovariances r_k.

beyond 11 lags, where the original covariance becomes inadmissible. Figure 11 is a plot of the "relative positions", $-a_{kk}$, as functions of lag number, k.

Two programs, TAP1 and FCA1X1, implementing the tapering process are described below. A third program, RPROC, provides a non-tapered R-process spectral analysis for a real time series. All three programs stop with a diagnostic printout if a negative power P_N is encountered.

4.2 Program TAP1

A simplified flow chart describing TAP1 is given as fig. 12 below, followed by a detailed description of the function of each subroutine. The subroutines are described in order of their occurrence in the source deck.

4.2.1 TAP1, the Main Program

This program reads in the data, as indicated in the block labeled INPUT in fig. 12, and calls the other subroutines. The data NC,NSPE,NDAT,DT and YT(I),I=1,NDAT have the same meaning as in BURGA except that YT(I) is real. Our present version uses card input.

4.2.2 Subroutine DETREND

Except for minor simplifications for a real time series, this subroutine is identical to DETREND for the Burg process described in section 2.

4.2.3 Subroutine RCALC(R)

This subroutine calculates the estimates $\hat{r}_i$ (unbiased) or $\tilde{r}_i$ (biased) of the covariances, as in equations (1.6) and (1.7). COMMENT cards indicate how to obtain either option. The unbiased estimates $\hat{r}_i$ are recommended if the entropy is to be tapered. As many lagged-product sums as may be needed are obtained by a single FFT followed by a multiplication in the frequency domain and an inverse FFT. The number, NN, of $\hat{r}_i$ to be calculated, should

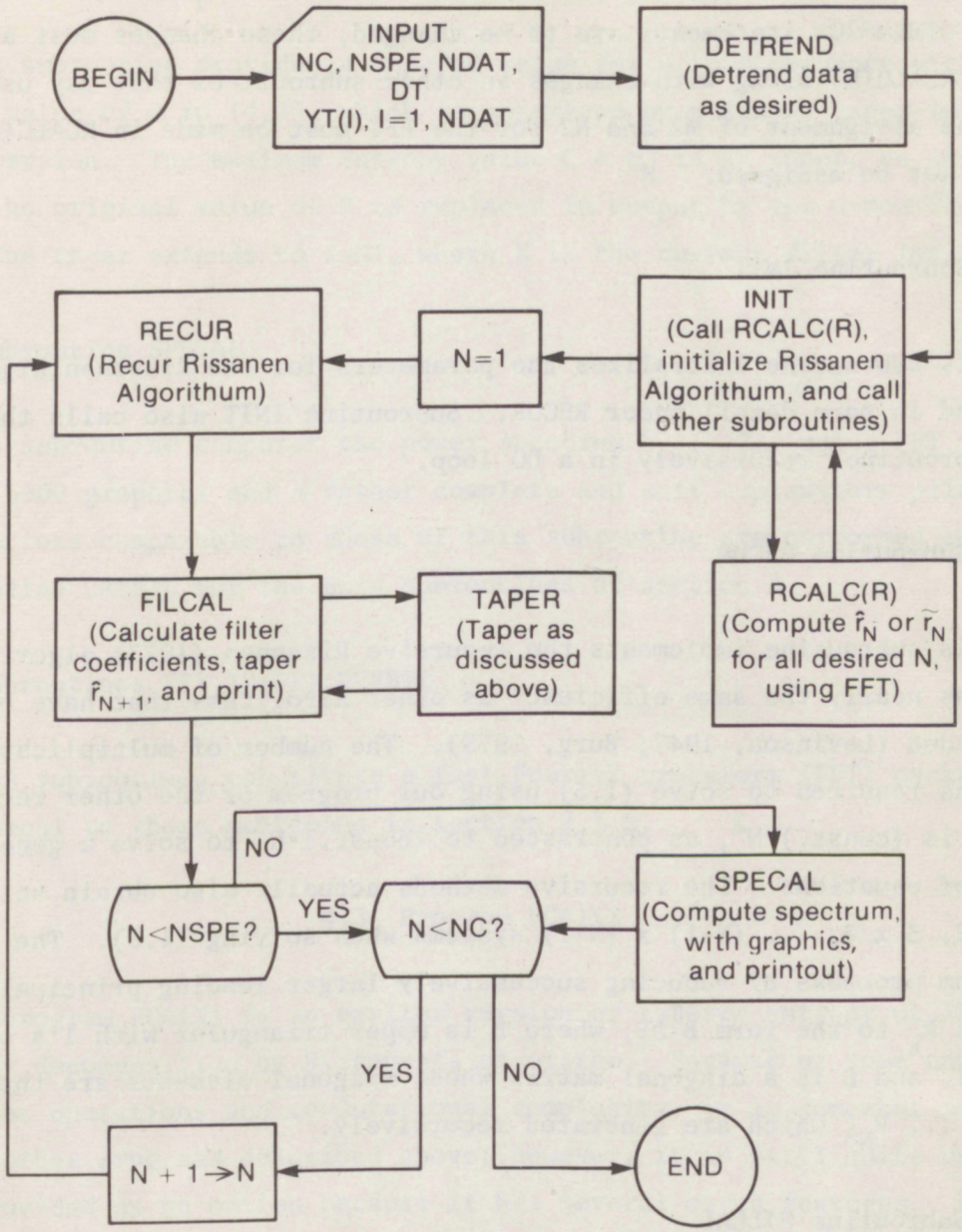


Figure 12. Simplified flow chart for the program TAP1.

be set equal to the DIMENSION of R. Note that R(I+1)=r_i in eq. (1.5). If the FFT DIMENSION statements are to be changed, these changes must also be made in RCALC(R) along with changes in other subroutines that may use the FFT. The assignment of M2 and N2 for the FFT must be made in RCALC(R); N3 need not be assigned.

4.2.4 Subroutine INIT

This subroutine initializes the parameters for the Rissanen algorithm, described in more detail under RECUR. Subroutine INIT also calls the other main subroutines recursively in a DO loop.

4.2.5 Subroutine RECUR

This subroutine implements the recursive Rissanen (1973) algorithm, which has nearly the same efficiency as other algorithms that have been recommended (Levinson, 1947; Burg, 1975). The number of multiplications and/or divisions required to solve (1.5) using our program or the other recursive methods is (const.)$\cdot N^2$, as contrasted to (const.)$\cdot N^3$ to solve a general N x N system of equations. The recursive methods actually also obtain solutions of 2 x 2, 3 x 3, ... (N-1) x (N-1) systems when solving (1.5). The Rissanen algorithm proceeds by reducing successively larger leading principal minor matrices R_k to the form B^TDB, where B is upper triangular with 1's on the diagonal, and D is a diagonal matrix whose diagonal elements are the powers P_0, P_1, ... P_N, which are generated recursively.

4.2.6 Subroutine FILCAL

A formula for updating the filter coefficients a_{kN} (which is easily obtained from Rissanen's B^TDB decomposition) has been implemented in this subroutine. A feature to be noted here is that the coefficients after the first have been defined as the negatives of the a_{kN} in (1.1) and (1.5). Thus for K1>1, -H2(K1)=$a_{K1-1,N}$ and H2(1)=1. This subroutine calls TAPER and provides a tapered-entropy value of R(K+1). Some clarification is given in the printouts.

4.2.7 Subroutine TAPER(A,C,R,K,N1)

This subroutine provides a tapered value for R in accordance with (4.8). A is the value of α in (4.8), which is generated by a linear taper in the present version. The maximum entropy value C = r_N^C is an input, as are R,K and N1. The original value of R is replaced in output by the α-modified value. The taper extends to K=N1, where K is the current filter length.

4.2.8 Subroutine SPECAL

This subroutine computes the power spectrum by (1.2), using FFT, and provides 6600 graphics and a rather complete and self explanatory printout. The operations comparable to those of this subroutine are performed as part of subroutine CALFIL for the Burg subroutines of section 2.

4.2.9 Subroutines FFT,INIFFT,REVBIT

These subroutines constitute a fast Fourier transform (FFT) package, and are identical to those mentioned in section 2.1.6.

4.3 Program FCA1X1

The program FCA1X1 is an earlier version of tapered entropy utilizing a Cholesky decomposition by Rissanen's algorithm. Because of some unnecessary square root operations and computational complexity, it is somewhat more bulky than the other programs described above; however, it is still quite useable, and is provided as an option because it has several extra features. First, its detrending routine provides for winnowing the data to operate on every KDDth data point and/or a segment of the original data. Second, it computes and presents both numerical and graphical outputs of the interval of admissibility as defined by (4.7), and third, it provides a convenient page plot of the spectrum in dB below the maximum. This page plot requires no unusual systems routines and is received without delay along with the numerical printout. A simplified flow chart describing FCA1X1 is given as fig. 13 below, followed by a detailed description of the function of each

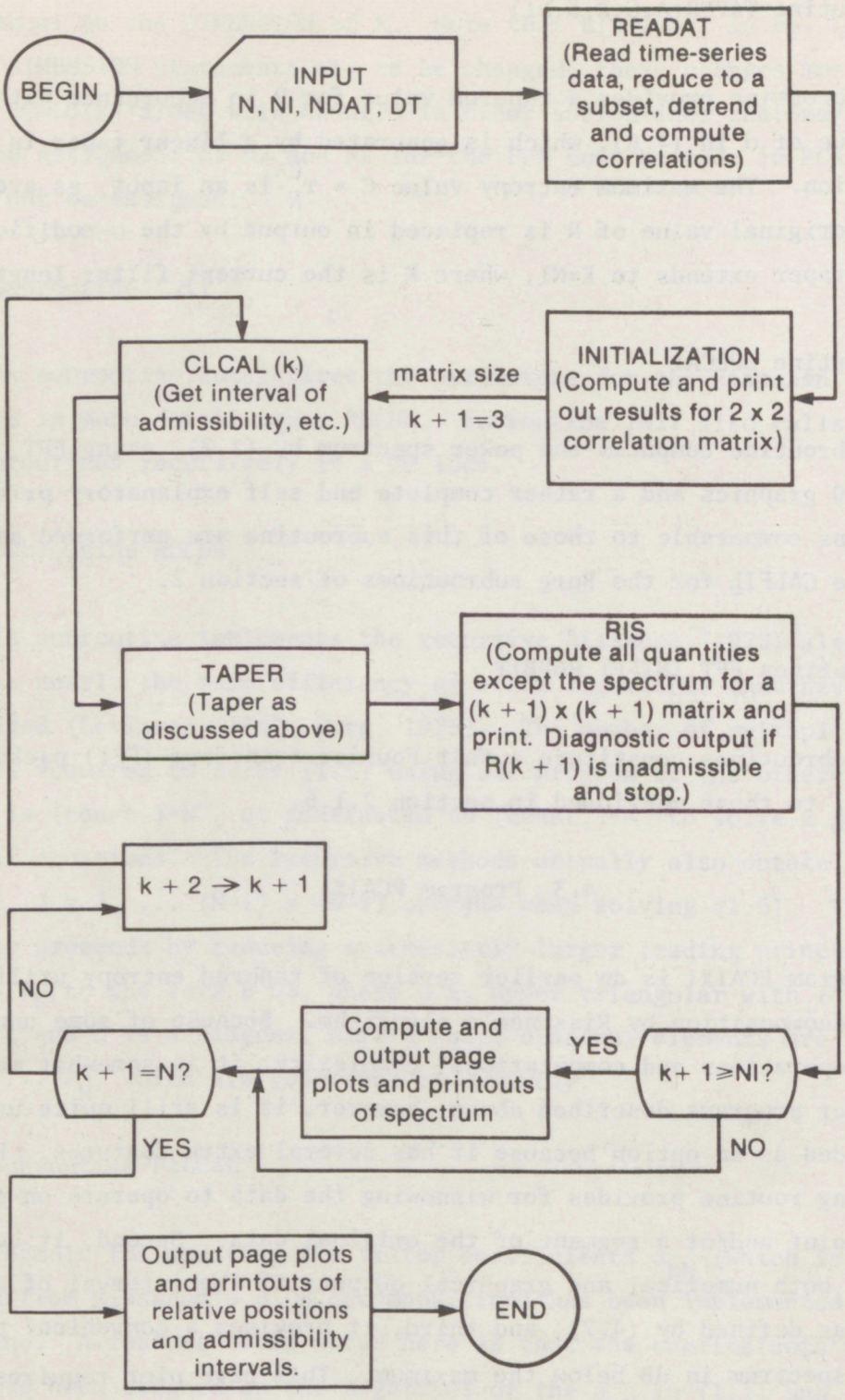


Figure 13. Simplified flow chart for the program FCA1X1.

subroutine. The subroutines are described in order of their occurrence in the source deck.

4.3.1 FCA1X1, the Main Program

This program calls the various subroutines as required for the recursive application of the tapered R-method. The program computes filters of length 1,2, ... N1, ... N. A full spectral-density output, consisting of numerical data and page plots is provided for filters of length N1, N1+1, ..., N. Here N1 and N have the same meaning as NSPE and NC in the Burg programs of section 2. If one wishes to use values of the covariances as inputs instead of the time series, he may replace the statement CALL READAT(R) by a series of statements that cause the covariances to be read in as R(1), R(2), ... R(N). The filter of length 1 (2 x 2 coefficient matrix in (1.5)) is handled as a special case, and all quantities required to get the Rissanen algorithm started are initialized in this main program. The evaluation of the spectrum by calling FFT routines is also done here. Some further information is given on COMMENT cards.

4.3.2 Subroutine RIS(K)

This subroutine recurs the filter coefficients by applying the Rissanen (1973) fast Cholesky algorithm. An upper-triangular matrix CHOL(I,J) is generated such that $(CHOL)^T(CHOL)=R_k$. The powers POW(I) are found as CHOL(I,I)**2. If POW(I) is positive, its square root is taken and the algorithm proceeds normally. If it is negative, $\hat{r}_k$ is inadmissible and the program stops with a diagnostic printout. The particulars relating the inadmissible value of $\hat{r}_k$ to the interval of admissibility (4.7) are printed out in this case, and a page plot of the sequence of a_{kk} is presented.

4.3.3 Subroutine CLCAL(K)

This subroutine computes r_k^C by a method equivalent to (4.5) and computes the interval of admissibility as defined by (4.7). In the printouts, r_k^C is

called the center-line value of r_k. A 2 x 2 covariance matrix is handled as a simple special case.

4.3.4 Subroutine TAPER(A,C,R,K,N1)

This is the same subroutine that is used in the program TAP1, section 4.2.7.

4.3.5 Subroutine READAT(R)

This subroutine reads in NDAT values of the time series YT(I),I=1,2,..., NDAT, where NDAT $\leq$ 1024. It then reduces the data to a subset consisting of every KDDth data point for a given subsegment, if desired. Then NDAT and the time interval DT are readjusted for the data actually used. COMMENT cards detail exactly how to obtain a spectral estimate based on a desired subset of the original data. The data remaining after this winnowing are then detrended by a linear regression. The two statements JSKIP = 1 and IF(JSK1P.EQ.1) GO TO 300 can be located in various places to obtain no detrending, subtraction of the mean, or full linear detrending. Lagged-product sums are computed from the detrended data by FFT. Either biased estimates $\tilde{r}_i$ or unbiased estimates $\hat{r}_i$ can be obtained, as indicated by COMMENT cards.

4.3.6 Subroutine PRPLOT

This subroutine provides a handy page plot of the admissibility intervals in one application and the set of relative positions in the other.

4.3.7 Subroutine DBPLOT

This subroutine provides a handy page plot of the spectrum in DB relative to the maximum. It provides a dynamic range of 50 dB. Any values smaller than -50 dB are plotted as -50 dB.

4.3.8 Subroutines FFT, INIFFT, and REVBIT

These subroutines constitute a fast Fourier transform (FFT) package, and are identical to those mentioned in section 2.1.6.

5. SUMMARY

The programs described in this report are summarized in table 1 below. In the checkout of all full programs we implemented eq (2.1), punched out the values YT(I), I=1,2,...128, with a format ±X.XXX, and then modified the programs to accept the resulting cards as input. Thus the inputs can be considered to be exact to the single-precision accuracy of the CDC 6600, about 11 decimal places. Complete computer source decks with this checkout data and graphical and/or printed output can be provided. If it is desired to implement one of these programs on a limited-accuracy computer, comparisons with the numerical printout supplied will give an estimate of the accuracy of the results. All programs listed in table 1 are available from the authors on request.

6. ACKNOWLEDGMENTS

The authors gratefully acknowledge the contribution of W. B. Sweezy of the Wave Propagation Laboratory, who supplied the fast Fourier transform routines, and the contribution of J. A. Leise of CIRES, who supplied the 6600 microfilm plotting routines.

Table 1. List of Computer Programs for Single-channel Maximum Entropy Spectral Analysis

Program Name	Type of data	Method	Approximate source deck thickness[1]	Approximate octal core storage[2]	FFT dimension[3]	Time series dimension[3]	Remarks
BURGA	Complex	Burg	3"	47 700	1024	1024	Complete printout and 6600 graphics[4].
BURGB	Complex	Burg	3"	23 300	512	512	Complete printout, no graphics.
BURGC	Complex	Burg	2"	42 400	1024	1024	Minimal printout, 6600 graphics.
BURGD	Complex	Burg	2"	42 300	1024	1024	Similar to BURGC but has no fine spectrum.
BURGE	Complex	Burg	1 3/4"	36 400	1024	1024	Minimal printout, no graphics.
MEMY	Real	Burg	3/8"	10 000	--	600	Subroutine only.
MSPEC	Real	Burg	1 3/4"	33 000	1024	600	Program implementation of MEMY.
SPEC	Complex	R-method	2"	31 500	1024	1024	Minimal printout, 6600 graphics.
RPROC	Real	R-method	2 7/8"	24 000	512	512	Complete printout, 6600 graphics.
TAP1	Real	Tapered Entropy	3"	24 200	512	512	Complete printout, 6600 graphics.
FCA1X1	Real	Tapered Entropy	4 1/2"	55 000	1024	1024	Complete printout, page plots.

NOTES:

1. Exclusive of data.

2. Storage depends largely on FFT dimension, Time series dimension, and data type. Includes declared storage and storage for program; may vary with different compilers.

3. In the Burg programs, the time series dimension need not be the same as the FFT dimension.

4. All CDC 6600 graphics generate microfilm.

7. REFERENCES

Akaike, H., 1971: Autoregressive model fitting for control. *Ann. Statist. Math.*, 23, 163-180.

Burg, J. P., 1975: Maximum entropy spectral analysis. Ph.D. dissertation, Dept. of Geophysics, Stanford University.

Cooley, J. W. and J. W. Tukey, 1965: An algorithm for the machine calculation of complex Fourier series *Math. Comp.*, 19, 297-301.

Durbin, J., 1973: The fitting of time-series models. *Rev. Inst. Int. Statist.*, 28, no. 3, 233-243.

Haykin, S. and S. Kesler, 1976: The complex form of the maximum entropy method for spectral estimation. *Proc. IEEE*, 64, 822-823.

Kanasewich, E. R., 1973: *Time Sequence Analysis in Geophysics*, University of Alberta Press, Edmonton, Canada.

Lacoss, R. T., 1971: Data adaptive spectral analysis methods. *Geophysics*, 36, no. 4, 661-675.

Levinson, N., 1947: The Wiener RMS (root mean square) error criterion in filter design and prediction. *J. Math. Phys.*, 25, no. 4, pp. 261-278.

Makhoul, J., 1975: Linear prediction: A tutorial review. *Proc. IEEE*, 63, no. 4, 561-579.

Mirsky, L., 1961: *An Introduction to Linear Algebra*. Oxford, Clarendon Press.

Rissanen, J., 1973: Algorithms for triangular decomposition of block Hankel and Toeplitz matrices with application to factoring matrix polynomials. *Math. Comp.*, 27, no. 121, 147-154.

Strand, O. N., 1977: Computer programs for maximum entropy spectral analysis of real and complex multichannel time series. *NOAA Tech. Report*, in preparation.

Ulrych, T. J. and T. N. Bishop, 1975: Maximum entropy spectral analysis and autoregressive decomposition. *Rev. Geophys. and Space Phys.*, 13, no. 1, 183-200.

van den Bos, A., 1971: Alternative interpretation of maximum entropy spectral analysis. *IEEE Trans. Inf. Theory, IT-17*, 493-494.