NOAA Technical Memorandum EDS CEDDA-6

DATA EDITING--SUBROUTINE EDITQ

Donald T. Acheson

Center for Experiment Design
and Data Analysis
Washington, D.C.
June 1975

noaa NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION / Environmental Data Service

NOAA TECHNICAL MEMORANDUMS
Environmental Data Service, CEDDA Series

The Center for Experiment Design and Data Analysis (CEDDA) became part of NOAA's Environmental Data Service in 1972 and was given the responsibility for data management and research activities related to major international scientific field experiments.

Formerly the Barbados Oceanographic and Meteorological Analysis Project (BOMAP), CEDDA is still concerned with analyses of data collected during the Barbados Oceanographic and Meteorological Experiment (BOMEX), conducted in 1969, and will continue to issue publications pertaining to BOMEX as part of the EDS BOMAP series.

NOAA Technical Memorandums in the Environmental Data Service CEDDA series will serve to disseminate information related to the 1972-73 International Field Year for the Great Lakes (IFYGL), the 1974 Global Atmospheric Research Program (GARP) Atlantic Tropical Experiment (GATE), as well as other projects in which CEDDA participates.

Publications listed below are available from the National Technical Information Service, U.S. Department of Commerce, Sills Bldg., 5285 Port Royal Road, Springfield, Va. 22151. Prices on request. Order by accession number (given in parentheses).

NOAA Technical Memorandums

EDS CEDDA-1 Omega Wind-Finding Capabilities: Wallops Island Experiments. Donald T. Acheson, October 1973, 77 pp. (COM-74-10039)

EDS CEDDA-2 Characteristics of the Lower Atmosphere Near Saipan, April 29 to May 16, 1945. Joshua Z. Holland, in press, 1975.

EDS CEDDA-3 IFYGL Physical Data Collection System: Intercomparison Data. Jack Foreman, May 1975, 7 pp.

EDS CEDDA-4 Preliminary Report on Wind Errors Encountered During Automatic Processing of IFYGL LORAN-C Data. J. Sullivan and J. Matejceck, May 1975, 9 pp.

EDS CEDDA-5 Generation of GATE Ship Speed Data by Variational Technique. Jerry Sullivan, June 1975.

NOAA Technical Memorandum EDS CEDDA-6

DATA EDITING--SUBROUTINE EDITQ

Donald T. Acheson

Center for Experiment Design
and Data Analysis
Washington, D.C.
June 1975

UNITED STATES
DEPARTMENT OF COMMERCE
Rogers C. B. Morton, Secretary

NATIONAL OCEANIC AND
ATMOSPHERIC ADMINISTRATION
Robert M. White, Administrator

Environmental Data
Service
Thomas S. Austin, Director

CONTENTS

	Page
Abstract	1
1. Introduction	1
2. Use	2
3. Polynomial Fitting as a Filter	3
4. Operation and Flow	4
Reference	5
Appendix	9

DATA EDITING - SUBROUTINE EDITQ

Center for Experiment Design and Data Analysis
National Oceanic and Atmospheric Administration
Washington, D.C. 20235

Abstract. EDITQ is a FORTRAN subroutine designed to recursively edit, i.e., remove wild or suspicious points from a data sequence. Editing is done by fitting a second-order polynomial $\hat{y}_i = a_2 + a_1x_i + a_2x_i^2$, $i = 1, 2, \ldots, N$ to a paired sequence $\{(y_1, x_1), (y_2, x_2), \ldots, (y_N, x_N)\}$, computing the residual variance

$$\sigma_R^2 = 1/(N - 3) \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \quad ,$$

and forming the dimensionless ratio $R_i^2 = (y_i - \hat{y}_i)^2/\sigma_R^2$ for each point. If R_i^2 exceeds some specified limit, the point (y_i, x_i) is rejected, and the residual variance and polynomial coefficients are recomputed without this data point. The process continues until no further data points are rejected in subsequent passes over the entire sequence. The subroutine returns the original sequence with flagged rejected values of y, the final values of the residual variance and polynomial coefficients, and the covariance matrix of the coefficients.

1. INTRODUCTION

EDITQ is a general-purpose routine designed to edit data, i.e., to remove or flag wild or ridiculous or merely suspicious data values. Data are input as a sequence of paired independent and dependent variables. The principal outputs are the same data with edited data points flagged; the three coefficients of a second-order polynomial fitted to all accepted, but no rejected, data points; the covariance matrix of these coefficients; and the residual variance.

Editing is done as follows:

(1) The dependent variable is least-squares fitted with a second-order polynomial in the independent variable, and the residual variance is calculated.

(2) The absolute value of the deviation of each dependent datum from the fitted polynomial is compared with the square root of the residual variance. If their ratio exceeds a specified limit, the dependent datum is flagged and rejected from further consideration.

(3) When a datum is rejected, the polynomial coefficients and residual variances are recomputed without this data point, and the next dependent datum is examined in the same way. This process continues until no further points are removed, or a fixed number of passes have already been made over the entire data sequence.

The principal advantages of EDITQ are:

(1) The accept/reject criterion for editing described in step (2) above automatically accomodates both very noisy and very clean data.

(2) EDITQ is recursive. Once a data point, no matter how wild, is rejected, it no longer figures in any further computations and cannot obscure other data points with much smaller error content but still worthy of rejection.

(3) The polynomial coefficients returned provide a simple means to compute replacement values for those rejected in the process.

The principal restrictions on the use of EDITQ are:

(1) If we view the dependent data as describing an underlying physical process more or less obscured by noise, it must be possible to approximate the underlying physical process by a second-order polynomial in the independent variable over the range of the data sequence. In other words, the modeling error must be significantly less than the residual variance returned. For example, a rapidly varying dependent variable, say surface solar radiation, could only be edited effectively if the sample rate were very high to allow EDITQ to operate on a sequence with a sufficient number of points to do useful editing but sufficiently short in duration for a second-order polynomial to be a good approximation to the real physical process.

(2) The independent variable is not edited. Unpredictable results occur when this variable contains errors.

(3) There is no physics in EDITQ. It has utterly no capability for editing consistently ridiculous data, say a sequence of absolute zeroes in temperature as a function of time. EDITQ is adept at editing outliers in a statistical sense, but must be preceded by a physical credibility window.

2. USE

With reference to the listing in the appendix to this discussion, users must input the two-dimensional array DATA, its dimensions in storage LENGTH and IMAX, the index IND at the independent variable and the index I of the dependent variable, the maximum number of points MAXOUT that may be rejected and also the dimension of array NLOST, the value BOUND that must be more negative than the negative of the absolute value of any acceptable dependent datum, and the accept/reject ratio SDLIM.

The routine returns the array DATA in which all values of the independent data and all accepted dependent data values are as input, while those rejected have the value DATA ($J_{rejected}$, I) + 2 x BOUND; the number of points rejected NCORR; the indices of rejected points NLOST (k), k = 1, 2, ... , NCORR; the final polynomial coefficients AØ, A1, and A2; the covariance matrix COVAR of these three coefficients; the square root of the residual variance STD, and the maximum deviation DEVMAX of any data point from the polynomial current at the time DEVMAX was found.

Appropriate choices of some of these parameters lend considerable flexibility to the use of EDITQ, e.g.,

(1) SDLIM - If SDLIM is greater than $[\text{LENGTH-3}]^{1/2}$, no editing will take place, and the routine serves simply to least-squares fit the data and return the residual variance, polynomial coefficients, and their covariance matrix.

(2) BOUND - If attention is paid to the number of significant digits that can be represented in the computer being used, original data values of rejected points can be recovered by addressing the array NLOST for the indices J_{rejected} = NLOST (k), k = 1, 2, ..., NCORR and subtracting 2 x BOUND from the returned values in DATA. If preliminary editing has been done, say simple checking for physically impossible values, and points failing this test are assigned values less than BOUND, EDITQ will not consider them further.

3. POLYNOMIAL FITTING AS A FILTER

Since a least-squares fitted second-order polynomial is, in general, a smooth estimator of noisy data, it is suitable to compare this estimate with simple arithmetic averaging.

Define time-centered simple arithmetic averaging as

$$\overline{y}_N(o) = 1/(2N+1) \sum_{j=-N}^{N} y_j \ , \ y_j = y(t_j) \quad , \tag{1}$$

where t_j is time centered (without loss of generality) at $t_o = 0$. The analogous second-order least-squares fitted estimate is

$$\hat{y}_N(o) = a_0 + a_1 t_0 + a_2 t_0^{\,2} = a_0 \quad ,$$

$$a_0 = (\Sigma\, t_j^{\,4}\ \Sigma\, y_j - \Sigma\, t_j^{\,2}\ \Sigma\, t_j^{\,2} y_j)/[(2N+1)\ \Sigma\, t_j^{\,4} - (\Sigma\, t_j^{\,2})^2] \quad , \tag{2}$$

all sums being j = -N, ..., N.

The transfer function of a filter is defined as the ratio of the Fourier transform F of output to that of the input,

$$\overline{Y}_N(\omega) = F(\overline{y}_N)/F(y) \quad ,$$

$$\hat{Y}_N(\omega) = F(\hat{y}_N)/F(y) \quad . \tag{3}$$

Since both smoothing operations defined in (1) and (2) are linear, their transfer functions are easily written down for the time-centered, equispaced sampling case ($t_j = j\Delta t$),

$$\overline{Y}_N(\omega) = (1/2N + 1)(1 + 2 \sum_{j=1}^{N} \cos \omega j\Delta t)$$

$$= (1/2N + 1) \sin [(2N + 1) \omega\Delta t/2]/\sin (\omega\Delta t/2) \quad , \tag{4}$$

$$\hat{Y}_N(\omega) = \Big([(3N^2 + 3N - 1)/S](1 + 2 \sum_{j=1}^{N} \cos \omega j\Delta t)$$

$$- 2 \sum_{j=1}^{N} j^2 \cos \omega j\Delta t\Big)/[(2N + 1)(3N^2 + 3N - 1)/S - N(N + 1)(2N + 1)/3] \ . \tag{5}$$

Eqs. (4) and (5) are plotted in figure 1 for N = 6, 12, and 24.

It is evident from figure 1 that the shape of the transfer functions for either simple averaging or least-squares polynomial fitting do not change appreciably as N is increased. As N is decreased, this is no longer true. In fact, for N = 1, the polynomial transfer function $\hat{Y}_N(\omega)$ equals 1, as can be seen from (5). This is not surprising since a second-order polynomial exactly fits three data points.

The major change in either transfer function with varying N occurs when Δt, the data sampling interval, is kept constant. If we define the simple averaging filter "bandwidth" as $\omega' = \pi$ in figure 1, then the "bandwidth" = $2\pi/(2N + 1)\Delta t$ decreases rapidly with increasing N as one expects for any averaging or smoothing filter. From figure 1 it is also easy to compare the effects of simple averaging and second-order polynomial fitting. Since the side-lobe structure of both filters is essentially the same, we need only be concerned with the bandwidths. The polynomial filter bandwidth is about 1.75 times that of simple averaging for the same number of points, so comparability is obtained when $(2N + 1)_{\text{polynomial}} \cong 1.75\ (2N + 1)_{\text{simple averaging}}$.

4. OPERATION AND FLOW

EDITQ is shown in figure 2. The basic equations are given below.

The least-squares coefficients of a second-order polynomial $y = a_0 + a_1x + a_2x^2$ are

$$a_0 = [S_y(S_{x2}S_{x4} - S_{x3}{}^2) + S_{xy}(S_{x3}S_{x2} - S_xS_{x4}) + S_{x2y}(S_xS_{x3} - S_{x2}{}^2)]/D \ , \tag{6}$$

$$a_1 = [L(S_{xy}S_{x4} - S_{x2y}S_{x3}) + S_x(S_{x2y}S_{x2} - S_yS_{x4}) + S_{x2}(S_yS_{x3} - S_{xy}S_{x2})]/D \ , \tag{7}$$

$$a_2 = [L(S_{x2}S_{x2y} - S_{x3}S_{xy}) + S_x(S_{x3}S_y - S_xS_{x2y}) + S_{x2}(S_xS_{xy} - S_{x2}S_y)]/D \quad , \quad (8)$$

$$D = [L(S_{x2}S_{x4} - S_{x3}^2) + S_x(S_{x3}S_{x2} - S_xS_{x4}) + S_{x2}(S_xS_{x3} - S_{x2}^2)] \quad , \quad (9)$$

where

$$S_x = \sum_{j=1}^{L} x_j \quad ,$$

$$S_{x2} = \sum_{j=1}^{L} x_j^2 \quad ,$$

$$S_{x2y} = \sum_{j=1}^{L} x_j^2 y_j \quad , \text{etc.}$$

The residual variance σ^2 is given by

$$\sigma^2 = 1/(L - 3) \sum_{j=1}^{L} (y_j - a_0 - a_1x_j - a_2x_j^2)^2 \quad ,$$

but can be expanded and summed and more efficiently written as

$$\sigma^2 = (S_{y2} - 2a_0S_y - 2a_1S_{xy} - 2a_2S_{x2y} + La_0^2 + 2a_0a_1S_x + 2a_0a_1S_{x2} + a_1^2S_{x2} + 2a_1a_2S_{x3} + a_2^2S_{x4})/(L - 3) \quad . \quad (10)$$

The covariance matrix C_{ov} of a_0, a_1, and a_2, is given by (Jenkins and Watts, 1968)

$$C_{ov} = \begin{pmatrix} L & S_x & S_{x2} \\ S_x & S_{x2} & S_{x3} \\ S_{x2} & S_{x3} & S_{x4} \end{pmatrix}^{-1} \sigma^2 \quad . \quad (11)$$

In (11), the element C_{ov} (1, 1) is the estimated error variance in the coefficient a_0, C_{ov} (1, 2) is the estimated error covariance between a_0 and a_1, and so forth.

REFERENCE

Jenkins, A.M., and D.G. Watts, Spectral Analysis and Its Applications, Holden-Day, San Francisco, 1968, 525 pp.

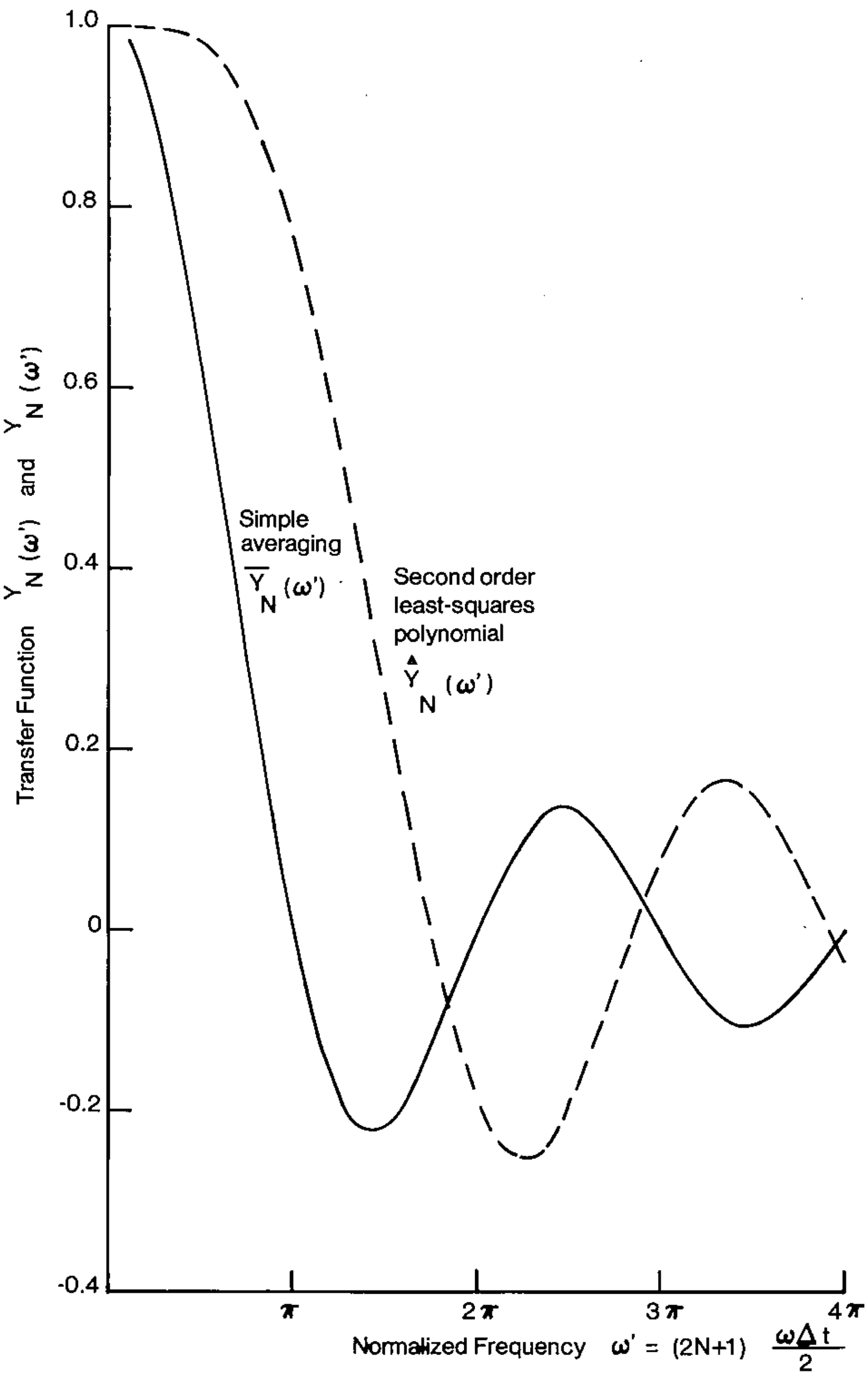

Figure 1.--Plot of $\overline{Y}_N(\omega')$ and $\hat{Y}_N(\omega')$ for N = 6, 12, and 24. Note that the differences between N = 6, 12, and 24 are too small to show clearly on this plot. As N increases, there is a slight suppression of the side lobes.

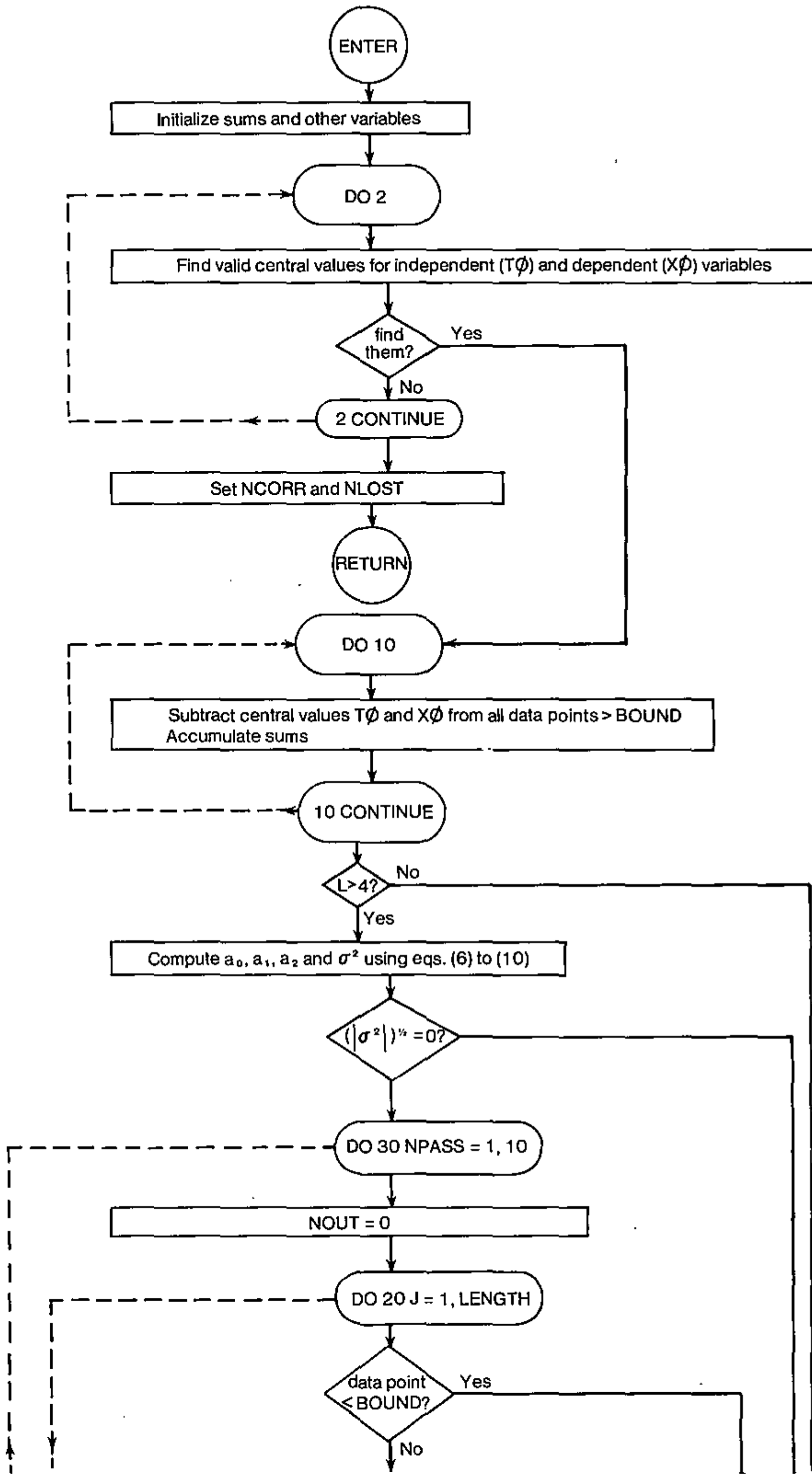

Figure 2.--Flow chart.

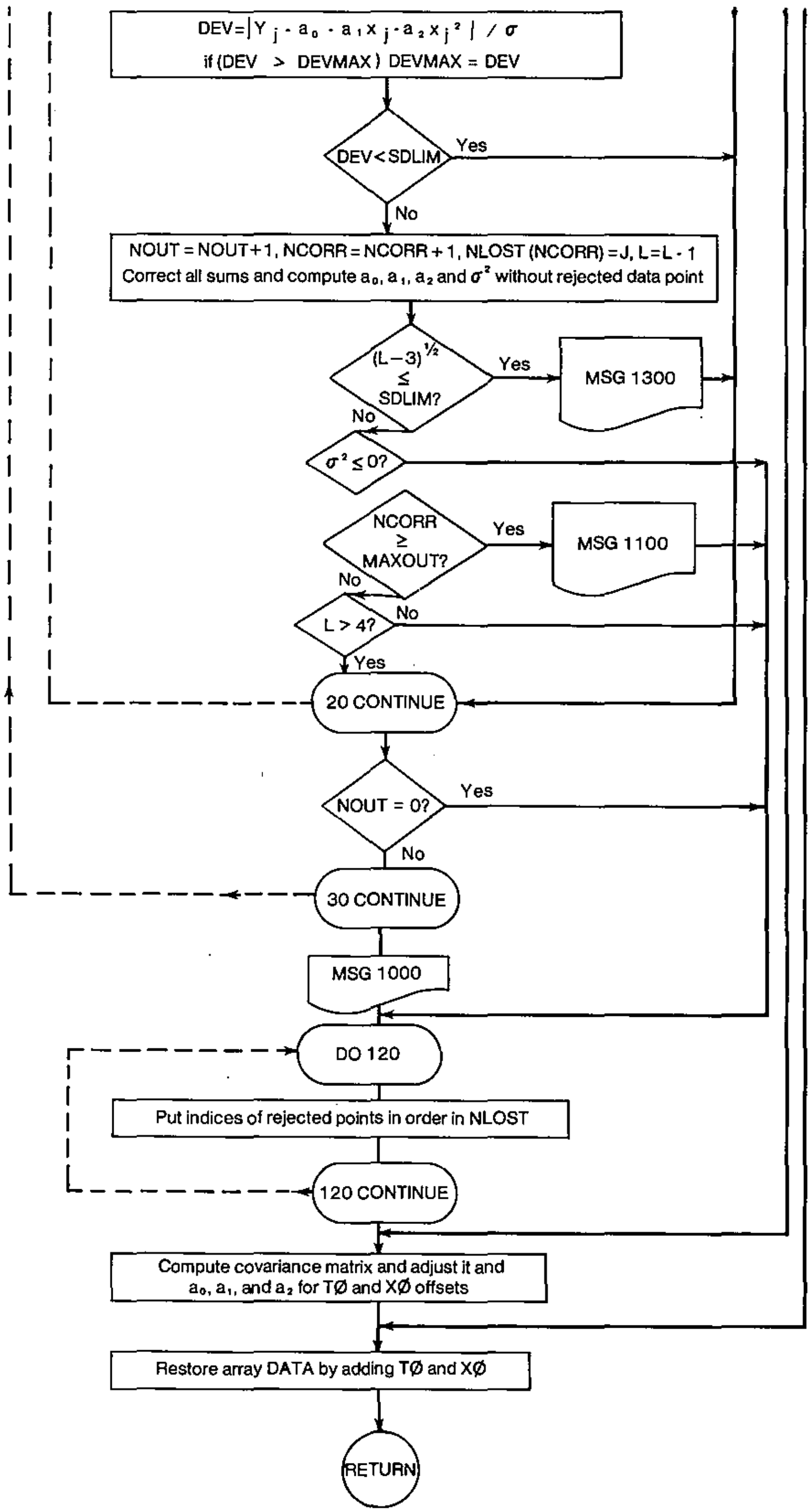

Figure 2.--Flow chart (continued).

APPENDIX

```
SUBROUTINE  EDITQ     TRACE                          CDC 6600 FTN V3.0-324  OPT=0  04

      SUBROUTINE EDITQ(DATA,LENGTH,IMAX,IND,I,MAXOUT,BOUND,SDLIM,NLOST,
     1                 STD,NCORR,A0,A1,A2,COVAR,DEVMAX)
C
C     PROGRAMMED BY DT ACHESON, CEDDA, APRIL 1972
C
C     EDITQ IS DESIGNED AS A COMPUTATIONALLY FAST AND EFFICIENT MEANS
C     OF FLAGGING SUSPICIOUSLY LARGE OR SMALL VALUES IN A SERIES OF DATA
C     THE DATA SERIES IS FITTED WITH A LEAST SQUARES FIT OF SECOND ORDER
C     POLYNOMIALS UNDER THE ASSUMPTION THAT THE PROGRAMMER LIMITS THE
C     LENGTH OF THE DATA SERIES TO REGIONS SUFFICIENTLY SMALL SO THAT
C     SECOND ORDER POLYNOMIALS ARE LOCALLY A GOOD APPROXIMATION TO THE
C     SHAPE OF THE CURVE.
C
C     INPUT
C
C       DATA(J,I), J=1,LENGTH IS THE UNEDITED DATA SERIES, I=1,IMAX
C       I          =THE INDEX OF THE VARIABLE TO BE EDITED.
C       MAXOUT     =THE MAXIMUM NUMBER OF DATAPOINTS WHICH MIGHT BE
C                   FLAGGED.
C       BOUND      =A NEGATIVE VALUE SMALLER THAN ANY ACCEPTABLE DATA
C                   VALUE AND LARGER IN ABSOLUTE VALUE THAN ANY
C                   ACCEPTEBLE DATA VALUE.
C       SDLIM      =THE LIMIT, IN NUMBER OF STANDARD DEVIATIONS, WITHIN
C                   WHICH THE DATA IS CONSIDERED ACCEPTABLE.
C       IND        =THE INDEX OF THE INDEPENDENT VARIABLE.
C
C     OUTPUT
C
C       DATA(J,I)  = THE ARRAY OF DATA WITH FLAGGED POINTS HAVING THE
C                   VALUE INPUT DATA(I,J)+2.0*BOUND
C       NLOST(M), M=1,MAXOUT IS THE ARRAY OF INDICES J (IN INCREASING
C                   ORDER) OF FLAGGED DATA.
C       STD        =THE RESIDUAL STANDARD DEVIATION OF THE DATA WITH THE
C                   FLAGGED POINTS EXCLUDED.
C       NCORR      =NUMBER OF POINTS FLAGGED.
C       A0,A1,A2   =THE LEAST SQUARES POLYNOMIAL COEFFICIENT OF THE DATA
C                   SERIES AFTER EDITING.
C       COVAR(N,N)=THE COVARIANCE MATRIX OF A0, A1, AND A2
C       N          =NUMBER OF POLYNOMIAL COEFFICIENTS
C
C     SUBROUTINES CALLED - NONE
C
C     REFERENCE - JENKINS AND WATTS
C
      DIMENSION DATA(LENGTH,IMAX),NLOST(MAXOUT),COVAR(3,3)
      COMMON /FILES/ ICARD,IFLTHD,MSCARD,IBSWCAL,IBOOM,MET0,NAV0,ISHIPN,
     1               IPRINT,MET1,NAV1,ITEMP,IARCHV
      SUMX=0.0
      SUMXSQ=0.0
      SUMX3=0.0
      SUMX4=0.0
      SUMY=0.0
      SUMYSQ=0.0
      SUMXY=0.0
      SUMX2Y=0.0
```

```
                  DEVMAX=0.0
                  RL=0.0
                  NCORR=0
                  A0=BOUND
                  A1=BOUND
                  A2=BOUND
                  IHALFL=LENGTH/2
                  DO 2 I0=1,IHALFL
                  I1=IHALFL+I0-1
                  X0=DATA(I1,I)
                  T0=DATA(I1,IND)
                  IF(X0.GT.BOUND.AND.T0.GT.BOUND) GO TO 4
                  I1=IHALFL-I0+1
                  X0=DATA(I1,I)
                  T0=DATA(I1,IND)
                  IF(X0.GT.BOUND.AND.T0.GT.BOUND) GO TO 4
                2 CONTINUE
                  NCORR=LENGTH
                  DO 3 J=1,MAXOUT
                  NLOST(J)=J
                3 CONTINUE
                  RETURN
                4 CONTINUE
                  DO 10 J=1,LENGTH
                  DATA(J,IND)=DATA(J,IND)-T0
                  IF(DATA(J,I).GT.BOUND) GO TO 5
                  NCORR=NCORR+1
                  NLOST(NCORR)=J
                  GO TO 10
                5 CONTINUE
                  DATA(J,I)=DATA(J,I)-X0
                  RJ=DATA(J,IND)
                  SUMX=SUMX+RJ
                  SUMXSQ=SUMXSQ+RJ*RJ
                  SUMX3=SUMX3+RJ*RJ*RJ
                  SUMX4=SUMX4+RJ*RJ*RJ*RJ
                  DATAJ=DATA(J,I)
                  SUMY=SUMY+DATAJ
                  SUMYSQ=SUMYSQ+DATAJ*DATAJ
                  SUMXY=SUMXY+RJ*DATAJ
                  SUMX2Y=SUMX2Y+RJ*RJ*DATAJ
                  RL=RL+1.0
               10 CONTINUE
                  IF(RL.LE.4.1) GO TO 127
                  AVG=SUMY/RL
                  DENOM=RL*(SUMXSQ*SUMX4-SUMX3*SUMX3)
                 1      +SUMX*(SUMX3*SUMXSQ-SUMX*SUMX4)
                 2      +SUMXSQ*(SUMX*SUMX3-SUMXSQ*SUMXSQ)
                  A0=(SUMY*(SUMXSQ*SUMX4-SUMX3*SUMX3)
                 1   +SUMXY*(SUMX3*SUMXSQ-SUMX*SUMX4)
                 2   +SUMX2Y*(SUMX*SUMX3-SUMXSQ*SUMXSQ))/DENOM
                  A1=(RL*(SUMXY*SUMX4-SUMX2Y*SUMX3)
                 1   +SUMX*(SUMX2Y*SUMXSQ-SUMY*SUMX4)
                 2   +SUMXSQ*(SUMY*SUMX3-SUMXY*SUMXSQ))/DENOM
                  A2=(RL*(SUMXSQ*SUMX2Y-SUMX3*SUMXY)
```

```
      1    +SUMX*(SUMX3*SUMY-SUMX*SUMX2Y)
      2    +SUMXSQ*(SUMX*SUMXY-SUMXSQ*SUMY))/DENOM
       VDV=(SUMYSQ-2.0*A0*SUMY-2.0*A1*SUMXY-2.0*A2*SUMX2Y+RL*A0*A0+2.*A0*
      1      A1*SUMX+2.0*A0*A2*SUMXSQ+A1*A1*SUMXSQ+2.0*A1*A2*SUMX3+A2*
      2      A2*SUMX4)/(RL-3.0)
       VDV=ABS(VDV)
       STD=SQRT(VDV)
       IF(STD.EQ.0.0) GO TO 125
       DO 30 NPASS=1,10
       NOUT=0
       DO 20 J=1,LENGTH
       IF(DATA(J,I).LT.(BOUND+0.1)) GO TO 20
       RJ=DATA(J,IND)
       DEV=ABS((DATA(J,I)-A0-A1*RJ-A2*RJ*RJ)/STD)
       IF(DEV.GT.DEVMAX) DEVMAX=DEV
       IF(DEV.LE.SDLIM) GO TO 20
       NOUT=NOUT+1
       NCORR=NCORR+1
       NLOST(NCORR)=J
       RL=RL-1.0
       SUMX=SUMX-RJ
       SUMXSQ=SUMXSQ-RJ*RJ
       SUMX3=SUMX3-RJ*RJ*RJ
       SUMX4=SUMX4-RJ*RJ*RJ*RJ
       DATAJ=DATA(J,I)
       DATA(J,I)=DATA(J,I)+X0+2.0*BOUND
       SUMY=SUMY-DATAJ
       SUMYSQ=SUMYSQ-DATAJ*DATAJ
       SUMXY=SUMXY-RJ*DATAJ
       SUMX2Y=SUMX2Y-RJ*RJ*DATAJ
       AVG=SUMY/RL
       DENOM=RL*(SUMXSQ*SUMX4-SUMX3*SUMX3)
      1      +SUMX*(SUMX3*SUMXSQ-SUMX*SUMX4)
      2      +SUMXSQ*(SUMX*SUMX3-SUMXSQ*SUMXSQ)
       A0=(SUMY*(SUMXSQ*SUMX4-SUMX3*SUMX3)
      1    +SUMXY*(SUMX3*SUMXSQ-SUMX*SUMX4)
      2    +SUMX2Y*(SUMX*SUMX3-SUMXSQ*SUMXSQ))/DENOM
       A1=(RL*(SUMXY*SUMX4-SUMX2Y*SUMX3)
      1    +SUMX*(SUMX2Y*SUMXSQ-SUMY*SUMX4)
      2    +SUMXSQ*(SUMY*SUMX3-SUMXY*SUMXSQ))/DENOM
       A2=(RL*(SUMXSQ*SUMX2Y-SUMX3*SUMXY)
      1    +SUMX*(SUMX3*SUMY-SUMX*SUMX2Y)
      2    +SUMXSQ*(SUMX*SUMXY-SUMXSQ*SUMY))/DENOM
       VDV=(SUMYSQ-2.0*A0*SUMY-2.0*A1*SUMXY-2.0*A2*SUMX2Y+RL*A0*A0+2.*A0*
      1      A1*SUMX+2.0*A0*A2*SUMXSQ+A1*A1*SUMXSQ+2.0*A1*A2*SUMX3+A2*
      2      A2*SUMX4)/(RL-3.0)
       STD=SQRT(ABS(VDV))
       IF(SQRT(RL-3.0).LE.SDLIM) WRITE(IPRINT,1300)
       IF(VDV.LE.0.0) GO TO 100
       IF(NCORR.GE.MAXOUT) GO TO 40
       IF(RL.LE.4.1) GO TO 100
    20 CONTINUE
       IF(NOUT.EQ.0) GO TO 100
    30 CONTINUE
       WRITE(IPRINT,1000)I
```

```
     1000 FORMAT(1H ,21H EDITING OF VARIABLE ,I2,28H EXHAUSTED BEFORE COMPLE
         1TION/)
          GO TO 100
       40 WRITE(IPRINT,1100) I,MAXOUT
     1100 FORMAT(1H ,21H EDITING OF VARIABLE ,I2,36H STOPPED BECAUSE THE MAX
         1IMUM NUMBER ,I4,18H HAVE BEEN REMOVED/)
     1300 FORMAT(1H0,86H WARNING - YOU HAVE TOO FEW POINTS TO EDIT EFFECTIVE
         1LY AT THE SPECIFIED VALUE OF SDLIM//)
      100 CONTINUE
          KSP=NCORR
          IF(KSP.LE.1) GO TO 125
          JSP=KSP-1
          DO 120 J=1,JSP
          KST=J+1
          DO 110 K=KST,KSP
          IF(NLOST(J).LT.NLOST(K)) GO TO 110
          NHOLD=NLOST(K)
          NLOST(K)=NLOST(J)
          NLOST(J)=NHOLD
      110 CONTINUE
      120 CONTINUE
      125 CONTINUE
          A0=X0+A0-A1*T0+A2*T0*T0
          A1=A1-2.0*A2*T0
          A2=A2
          COA0A0=(SUMXSQ*SUMX4-SUMX3*SUMX3)*VDV/DENOM
          COA0A1=(SUMX3*SUMXSQ-SUMX*SUMX4)*VDV/DENOM
          COA0A2=(SUMX*SUMX3-SUMXSQ*SUMXSQ)*VDV/DENOM
          COA1A1=(RL*SUMX4-SUMXSQ*SUMXSQ)*VDV/DENOM
          COA1A2=(SUMX*SUMXSQ-RL*SUMX3)*VDV/DENOM
          COA2A2=(RL*SUMXSQ-SUMX*SUMX)*VDV/DENOM
          COVAR(1,1)=COA0A0-2.*T0*COA0A1+2.*T0*T0*COA0A2+T0*T0*COA1A1
         1-2.*T0*T0*T0*COA1A2+T0*T0*T0*T0*COA2A2
          COVAR(1,2)=COA0A1-2.*T0*COA0A2-T0*COA1A1+3.*T0*T0*COA1A2
         1-2.*T0*T0*T0*COA2A2
          COVAR(1,3)=COA0A2-T0*COA1A2+T0*T0*COA2A2
          COVAR(2,2)=COA1A1-4.*T0*COA1A2+4.*T0*T0*COA2A2
          COVAR(2,3)=COA1A2-2.*T0*COA2A2
          COVAR(3,3)=COA2A2
          COVAR(2,1)=COVAR(1,2)
          COVAR(3,1)=COVAR(1,3)
          COVAR(3,2)=COVAR(2,3)
      127 CONTINUE
          DO 130 J=1,LENGTH
          DATA(J,IND)=DATA(J,IND)+T0
          IF(DATA(J,I).LE.BOUND) GO TO 130
          DATA(J,I)=DATA(J,I)+X0
      130 CONTINUE
          RETURN
          END
```