A COMPUTER AIDED DESIGN PACKAGE
FOR TIME SERIES ANALYSIS

By

©

ROBERT FRANK MOORE, B. ENG.

A Thesis
Submitted to the School of Graduate Studies
in partial fulfillment of the requirements
for the Degree
MASTER OF ENGINEERING
McMaster University
February 1982

A COMPUTER AIDED DESIGN PACKAGE
FOR TIME SERIES ANALYSIS

MASTER OF ENGINEERING (1982)
(Chemical)

McMASTER UNIVERSITY
Hamilton, Ontario

TITLE: A Computer Aided Design Package For Time Series Analysis

AUTHOR: Robert Frank Moore, B.ENG. (McMaster University)

SUPERVISORS: Dr. J.F. MacGregor, Dr. P.A. Taylor, Dr. J.D. Wright

NUMBER OF PAGES: xii, 142

ABSTRACT

Robert Frank Moore: A Computer Aided Design Package for Time Series Analysis. M.Eng. Thesis, McMaster University, Hamilton Ontario, February 1982.

Control problems often require a reasonable and adequate model for forecasting or control. An interactive computer package is presented which allows the iterative determination of a stochastic and transfer function model for single input/single output experimental data. Note that time series analysis techniques present various quantitative measures of model adequacy for this type of model. User familiarity with time series analysis is essential in the determination of a reasonable, parsimonious model during an interactive session with this package.

The interactive program provides initial diagnostic tests for the data to aid the estimation of parameters for a user selected model form. The Marquardt non-linear least squares procedure is available to improve the initial parameter estimates. Information for diagnostic testing of the model residuals is then displayed. The option for iteration back to the point of model form selection is available.

A multivariable analysis problem may be resolved in a step-wise manner using a non-iterative version of the time series package. This version fits a user selected model to the data and provides diagnostics on the model residuals. The computer package is presently functional on a VAX 11/750 computer. All graphical display characteristics are suitable for a Tektronics 4025 terminal. Programming for the package is in FORTRAN77.

ACKNOWLEDGEMENTS

It is a pleasure to acknowledge my thanks to my supervisors: Dr. J.F. MacGregor, Dr. P.A. Taylor, and Dr. J.D. Wright. Their assistance, guidance and instruction in the past year has been a valuable experience.

I would also like to express my appreciation:

To Dr. D.R. Woods for his valuable contributions to my engineering education.

To my employer, Atomic Energy of Canada, for providing time to complete this project.

To my family, for their patience and support during the past year.

To my father.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOTATION

SUBSCRIPTS

i	Vector or array index
t	indicating variable in time
t+k	indicating variable in time displaced k sampling intervals
0,1,2	component number

VARIABLES

T	Time
U	Control Variable
Y	Output Measurement

MODEL PARAMETERS

B	The backward operator $BU_t = U_{t-1}$
r,s,p,q	Orders of polynomials ω, δ, θ, ϕ respectively
ω, δ, θ, ϕ	Polynomials in B
∇	The difference operator $\nabla = (1-B)$
b	Whole number of periods of delay in model
d	Differencing of series U and Y in model
N_t	Calculated Noise = $Y_{t(observed)} - Y_{t(modelled)}$
A_t	Vector of White Noise (This is model objective.)

CHAPTER 1

INTRODUCTION

1.1 Project Overview

The forecasting and control of a process often requires the use of an adequate model of that process. The time series analysis techniques of Box and Jenkins[1] provide an iterative procedure for the development of stochastic models and transfer functions for discrete equispaced data. An interactive computer package has been written which will sequentially perform the mechanistic aspects of the Box and Jenkins techniques. Tabular data is automatically displayed and the option for interactive graphical display is available to the user. Reasonable familiarity with both the mechanistic and diagnostic aspects of these time series analysis techniques is required for the proper use of the computer package and will be assumed throughout this report.

The remainder of the Introduction will address in general terms the concerns behind the development of this computer package. In chapter 2 the package is presented by the processing of a sample data set. Emphasis is placed on the decisions made by the user interactively, and the data provided by the computer to aid those decisions. In chapter 3 the mainline of the program is compared to the model building scheme outlined in chapter 1.

A breakdown of the package subroutines by function, a cross index of the subroutines, and a tabular summary of subroutine functions with detailed call statements, are presented in chapter 4. Chapter 5 contains a summary of the computer package attributes and possible applications, including a stepwise multivariable analysis scheme. Full program and subroutine listings are in the Appendix.

1.2 Problem Solving, Engineering and the Computer

A five step, general problem solving strategy, which is suitable to a wide range of problems, was derived by modifying Polya's approach.[2]

Define the Problem

Think About It

Plan

Do It

Look Back

Figure 1 A Generalized Problem Solving Strategy

Note that this is a modular, stepwise, iterative procedure which reflects, in general terms, the way we approach problems. Implied within this procedure, via the "Look Back" step, is a judgement or evaluation criterion which allows recognition of a satisfactory solution to the problem.

The computer has often been referred to as a form of artificial intelligence; however, it must be emphasized that computers are not capable of evaluation. A computer program will only achieve and display those results which the programmer wishes to present. The modular nature of the time series computer package and the interactive display and computational speed of modern computers provide ready tools for the rapid presentation of engineering data. The final interpretation of that data is always the responsibility of the engineer using the package.

Each computer possesses computational and language capabilities which are best suited to specific programming structures and practices. Particular emphasis is generally made on the interactive or batch processing capabilities of a computer. From a programming point of view, all of these facets of a computer must be taken into account to fully utilize the resources available and minimize user frustration.

For example, in the design of an interactive computer package the programmer should place emphasis on structured programming techniques[3] and rapid display, computation and numerical analysis routines.

1.3 Interactive Design of Process Models

To facilitate the development of the evaluation and judgement skills which are needed in the analysis of time series data, the computer package incorporates the ability to create time series data for a user selected model. This allows the user to observe the effects of various model parameters on the diagnostic tests which the package performs.

Usually, the control engineer is confronted by a process for which a model is desired. An adequate and reasonable model is often the backbone of the engineering approach to the forecasting or control of a process. A general, iterative model building scheme[1] is outlined in figure 2. Note the similarity to the generalized problem solving strategy previously presented. In terms of time series analysis, this model building scheme indicates a need for parameter estimation techniques and model evaluation criteria.

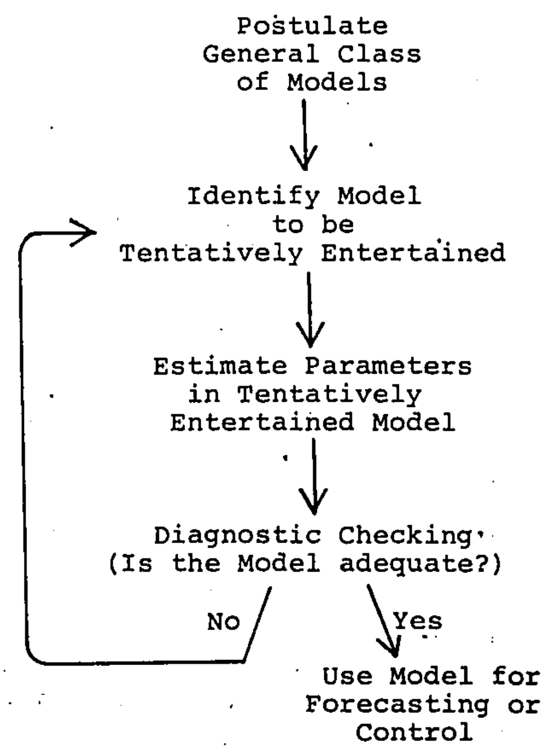


Figure 2 Stages in the Iterative Approach to Model Building

Reference: Box and Jenkins (1976) [1]

1.4 The Computer Aided Design Package

A time series analysis strategy[1] has been implemented on the computer as the mainline of the time series package in such a manner that the iterative model building procedure is executed interactively to determine the form and best parameter estimates of a generalized stochastic and transfer function model described by:

$$Y_t = \frac{\omega_r(B)}{\delta_s(B)} U_{t-b} + \frac{\theta_p(B)}{\phi_q(B)\nabla^d} A_t \qquad (1)$$

Within this model structure there are six integer parameters under user control which set the form of the model. They are:

- b - The number of whole periods of delay in the system.
- d - The degree of differencing of the series needed to obtain stationarity. (Note: $\nabla = 1 - B$)
- p,q,r,s - The orders of the polynomials $\theta, \phi, \omega, \delta$ in the backwards operator (B).

The time series computer package performs various initial diagnostic tests and computations which will allow the user to select reasonable starting values for these six parameters. Furthermore, some calculated results may be used to set initial parameter values within the selected model form. The Yule-Walker equations are an example of the use of the auto correlation function values to set the initial parameter estimates.

To allow improvement of the initial parameter estimates, the Marquardt Compromise Nonlinear Least Squares[4] procedure is provided in the computer package. Reasonable interactive display and computation times were the foremost considerations in the selection of the criterion for parameter convergence offered by this numerical routine. By solving equation 1 for the vector of residuals A_t, the computer may perform diagnostic tests on the "goodness of fit" of the model. Generally, the user stops the iterative process when a model with its best fitting parameters produces a vector of normally distributed random numbers as the residual values (A_t). This "white noise" criterion would usually indicate that a reasonable time series model has been achieved for the set of data under consideration.

CHAPTER 2

A WORKED EXAMPLE IN TIME SERIES ANALYSIS

2.1 The Data

Data for analysis was selected from experimental results collected on a butane hydrogenolysis reactor by A. Jutan.[5] The manipulated or controlled variable was the hydrogen flowrate (cm^3/s) to the reactor. The observed variable was the per cent conversion of butane as measured by an on-line analyzer. Measurements were available every five minutes (300 seconds) from the on-line device. Effectively, the requirements of discrete, equispaced sampling are met by this data. A full table of the data is given in the appendix.

The data was stored on the computer in file REACT.DAT using Fortran format (3F10.4) for the data points given by, (Time, Control, Observation) or (T, U, Y).

2.2 The Computer

A VAX 11/750 is presently in use in the Control Group of the Chemical Engineering Department at McMaster University. File manipulation and structuring are in accordance with standard VAX practices. The entire time series package is written in the VAX compatible Fortran 77 computer language.

The graphical capabilities placed within the time series package utilize the PLOT10 software in use at McMaster. All graphical software is functional on the Tektronics 4025 terminal. This terminal has a "split-screen" capability which creates two windows on the display screen. The top window presents the graphical display, and the bottom window, which is set to 5 horizontal lines in size by the program, presents the tabular data. On a non-graphics terminal only the tabular data display would be observed. Note that "scrolling" of the tabular display is possible on the 4025 terminal, so that the 5 line display size is not a major problem.

2.3 Program Description

The path of the information flow through the time series package depends on the options selected by the user. During execution, the package requires one data file and one subroutine (stored in a separate file). The user must provide these two files or the program will stop with a fatal error indicated. The following outline should clarify the information flow in the general time series package used by the command file MASTER.COM.

Single Input/ Single Output
Data is from either
Experiment or Assignment

Input FILENAME where the data is stored.
Read FILENAME with format (3F10.4)
Max. 1000 points of (T,U,Y)
Note that FILENAME must be in the user's directory.

Input Student Number
↓
If student number = 11 create file PLNT.DAT as known model in format of equation 1 and stop.
↓
Create up to 1000 data points (T,U,Y) using model in PLNT.DAT, and subroutine CONTROL in user supplied file.

Initial Diagnostic Options

Identification of U series
Identification of Y series
Impulse Response of Data
Cross Correlation $U_t * Y_{t+k}$
↓
Set Model form and initial parameter estimates using Subroutine MODEL with IFLAG=1.

Create files BINARY.DAT + TSDATA.DAT
↓
Marquardt Compromise to Improve parameter estimates. This uses Subroutine MODEL with IFLAG=0.
Default order of input of polynomials of equation 1 is $\theta_p, \phi_q, \omega_r, \delta_s$ where

dimensions limit (p+q+r+s).LE. 10.
↓
Identification of Residuals A_t.
↓
If $A_t \approx$ white noise to user satisfaction then stop, otherwise, loop back.

Note that this information flow corresponds to the general time series package used by the command file MASTER.COM. The program version available to command file MASTER1.COM does not offer the"data from Assignment"option or the loop back.

During the execution of the command file MASTER.COM the files BINARY.DAT and TSDATA.DAT will be created in the user's directory. The command file MASTER1.COM requires a user supplied subroutine MODEL and will not create these files.

There are several important package limitations which the user should note. The program has been written for a maximum of 1000 data points and 10 model parameters. Also, the user supplied data file format for the experimental data option must have format (3F10.4) for (T,U,Y). The parameter convergence criterion for the numerical routines have been selected for speed of numerical interactive computation. If numerical ill-conditioning problems arise consult reference 4 for computer library documentation on the behavior of the Marquardt Compromise parameters available for tuning in the mainline call to these routines.

The program display to the computer terminal is created by either print statements, write statements to TAPE6 (which is the output terminal by VAX system default), or by graphical routines. All prompts requiring user interaction were written by write statements to TAPE6. The tabular displays, and numerical analysis routine output are created by print statements. All default Input/Output file assignments are the terminal by definition on the VAX system. If the user requires a printer hardcopy output of a particular run of the time series package, the output of the program may be stored in a file on the user's directory by the use of a logical filename assignment for the local file FOR$PRINT created by the print statements. This output file may then be sent to the printer by regular methods.

In the event of file manipulation problems, the user should consult the VAX user's guide. This document addresses various common problems encountered by the user. Particularly note that a user may create files in his own directory only, although he may execute files stored in another user's directory.

2.3.0 Program Execution Procedure

The VAX file structure allows the creation of command files. These files perform sequences of file manipulation commands. There are two command files which allow execution of the time series package. The user must select the command file which will perform the option desired. Furthermore, the user must provide a fortran coded subroutine in a VAX Fortran-type file which satisfies the requirements of the program package. The entire package is then executed by a single command such as:

$@[MOORE] MASTER [YOURNAME] FILENAME

Here, the file MASTER.COM in directory [MOORE] will be immediately executed where "FILENAME" in directory [YOURNAME] will be termed file "p1" within the command file. Once the package begins execution, all prompts are in plain english and require knowledge of time series analysis techniques. No detailed computer knowledge is necessary.

2.3.1 Command File MASTER.COM

This file accesses the iterative, interactive time series package which uses the transfer function and stochastic model of equation 1. Note that instructions are printed for the user as this file executes.

```
$WRITE SYS$OUTPUT "USER SUBROUTINE CONTROL(UNEW,UOLD,YOLD,TIME) "
$WRITE SYS$OUTPUT "MUST EXIST IN FILE ''P1'"
$WRITE SYS$OUTPUT "WHERE DIMENSION UOLD(10),YOLD(10) IS PRESENT"
$FORTRAN 'P1'
$LINK [MOORE]INTER,[MOORE]TIMSER,[MOORE]TCS,[MOORE]PARAMETER,'P1'
$ASSIGN/SUPERVISOR SYS$COMMAND SYS$INPUT
$RUN INTER
$DEL INTER.EXE;*
$EXIT
```

Table 1 Listing of File MASTER.COM

If the user intends to analyze experimental data then Subroutine Control will never be called. However, to avoid non-fatal loader errors in package execution, a simple control subroutine such as the following example, should be provided by the user.

```
Subroutine Control(Unew, Uold, Yold, Time)
Dimension     Uold(10), Yold(10)
Print *,'EXIT ON CALL TO CONTROL'
Call Exit
Return
End
```

If the user intends to generate and analyze time series data by the package assignment option, then the control subroutine acts just like a controller for a known plant model. The plant model must be set up before this option can be used. The plant model is created by the user in file PLNT.DAT by answering student number eleven when prompted for student number. The version number of the PLNT.DAT;# file is used as the problem number. A special short program, given in the appendix, allows the user to view the parameters in the PLNT.DAT file to ensure that the desired model has been set up correctly.

At each Time corresponding to an integer multiple of the sample time input by the user in response to a prompt, the subroutine control is called upon to provide a new control action. Therefore, subroutine control acts just like a process controller. It is possible to use any controller, even a self tuning regulator, in this control subroutine because the process looks to the control subroutine for a new control action before calculating the next observation. Here is a sample of a simple control subroutine, the step test, which will provide the step response of the plant model at TIME=50.

```
      SUBROUTINE CONTROL(U,UOLD,YOLD,TIME)
      DIMENSION UOLD(10),YOLD(10)
      U=0.0
      IF(TIME.GE.50.)U=10.0
      RETURN
      END
```

If the user wishes to establish a model for an Auto Regressive Integrated Moving Average (ARIMA) process, then all control values for all time should be set to zero by the control subroutine or in the experimental data file, depending on the option selected.

2.3.2. Command File MASTER1.COM

This file accesses a subset of the time series package and it accepts a user supplied model subroutine.

```
$WRITE SYS$OUTPUT "USER SUBROUTINE MODEL(NPROB,TH,R,NOB,NPR,IFLAG) "
$WRITE SYS$OUTPUT "MUST EXIST IN FILE ''P1'"
$WRITE SYS$OUTPUT "WHERE DIMENSION R(NOB),TH(NPR) IS PRESENT"
$FORTRAN 'P1'
$LINK [MOORE]INTER1,[MOORE]TCS,[MOORE]PARAMETER,'P1'
$ASSIGN/SUPERVISOR SYS$COMMAND SYS$INPUT
$RUN INTER1
$DEL INTER1.EXE;*
$EXIT
```

Table 2 Listing of File MASTER1.COM

This subset package will process experimental data only. The user supplied model subroutine must satisfy two roles. If the value of IFLAG is less than zero, the subroutine must return the number of parameters in the model (NPR), and preliminary estimates of the model parameters (TH(I), I=1,NPR). If the value of IFLAG is greater than or equal to zero, then then subroutine must return the vector of model residuals (R(I), I=1,NOB). Note that in the case of the general model this meant the solution of equation 1 for the residuals, A_t.

```
      SUBROUTINE MODEL(NPROB,TH,R,NOB,NPR,IFLAG)
      DIMENSION R(NOB),TH(NPR)
      COMMON/DATA/T(1000),U(1000),Y(1000)
      IF(IFLAG.LT.0)THEN
        TH(1)=1.5
        TH(2)=2.0
        NPR=2
      ELSE
        DO 100 K=1,NOB
         R(K)=TH(1)*U(K)-TH(2)*Y(K)
100     CONTINUE
      END IF
      RETURN
      END
```

Table 3   Listing of a Model Subroutine

Note that this subset package is not iterative in nature. One run of the program with the starting values given in Model is achieved each time the MASTER1 file is executed.

2.4 Program Execution for a Sample Data Set

To fit the generalized stochastic and transfer function model to this test data execute the MASTER command file. The control subroutine listed in section 2.3.1 was safely used because the data generation option was not taken. The strategy of Box and Jenkins will be followed in this analysis.

2.4.1 Preliminary Identification

The computer package allows various preliminary identification techniques. For the purposes of demonstration, all of the options available will be shown. In a stepwise manner, the order of techniques is:

1. Plot Control (U) versus Time.
2. Plot Observation (Y) versus Time.
3. Identification of the U series. This consists of the calculation and display of the Auto and Partial Auto Correlation functions for the original, first differenced and second differenced U series.
4. Identification of the Y series.
5. Impulse and Step test calculations relating Y to U.
6. Cross Correlation function of Y(t)*U(t+k).

The results obtained and displayed by the computer are provided on the next few pages.

Figure 3:
CONTROL VARIABLE VERSUS TIME ORIGINAL SERIES

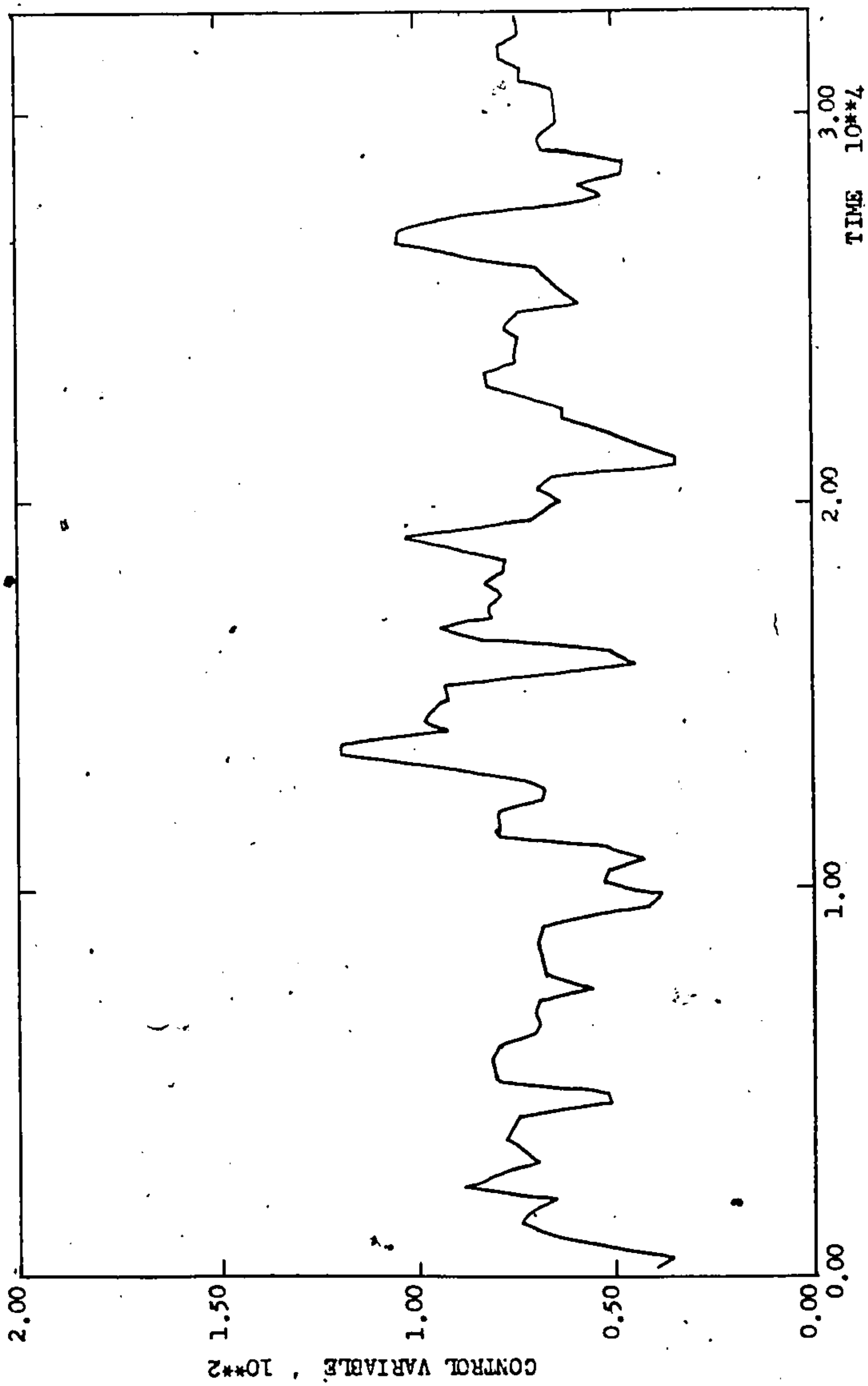
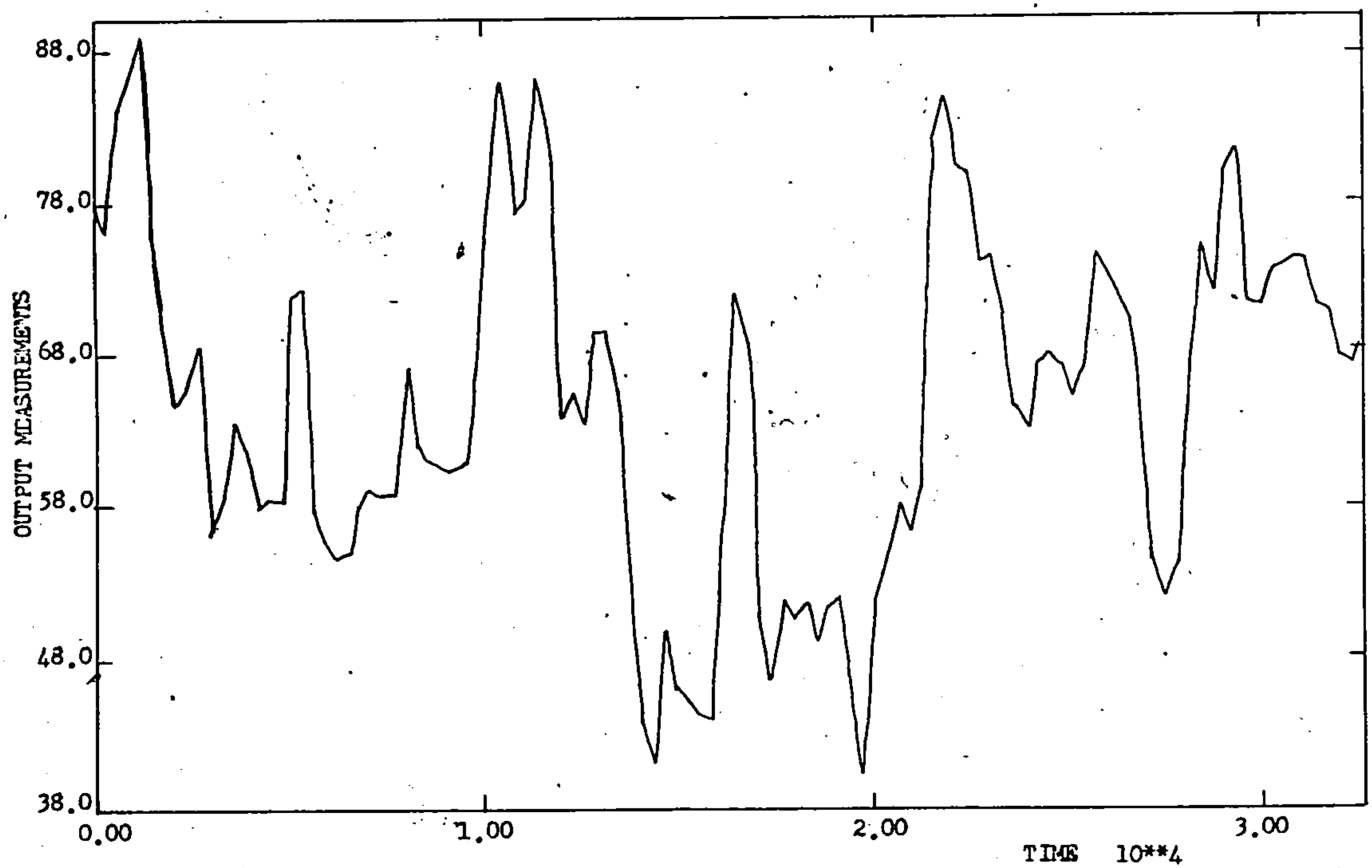


Figure 4:
OUTPUT MEASUREMENTS VERSUS TIME ORIGINAL SERIES



AUTO AND PARTIAL CORRELATIONS OF ORIGINAL SERIES

I	AUTO	PARTIAL
1	0.729	0.729
2	0.372	-0.340
3	0.178	0.152
4	0.087	-0.064
5	0.034	0.005
6	-0.064	-0.175
7	-0.155	-0.017
8	-0.125	0.118
9	-0.105	-0.170
10	-0.111	0.042
11	-0.099	-0.012
12	-0.097	-0.051
13	-0.102	-0.061
14	-0.101	-0.015
15	-0.124	-0.064
16	-0.085	0.086
17	-0.015	-0.005
18	0.000	-0.048
19	-0.056	-0.138
20	-0.063	0.135

APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = 0.192

STANDARD DEVIATION OF SERIES = 0.1673277E+02

CHI-SQUARED STATISTIC = 0.9491E+02 BASED ON 20 DEGREES OF FREEDOM

Table 4 Identification of Control Series

The 95% confidence limits of the correlations are indicated on the plots by the dashed lines. Note that all plots in this report are hand-drawn approximations of the computer display. A hard-copy plotter became available after the completion of this report. Approximate 95% confidence limits plotted are ± 2 * (Number of Observations)**-0.5 .

Figure 5:
AUTO CORRELATION VERSUS LAG, K ORIGINAL SERIES CONTROL VARIABLE

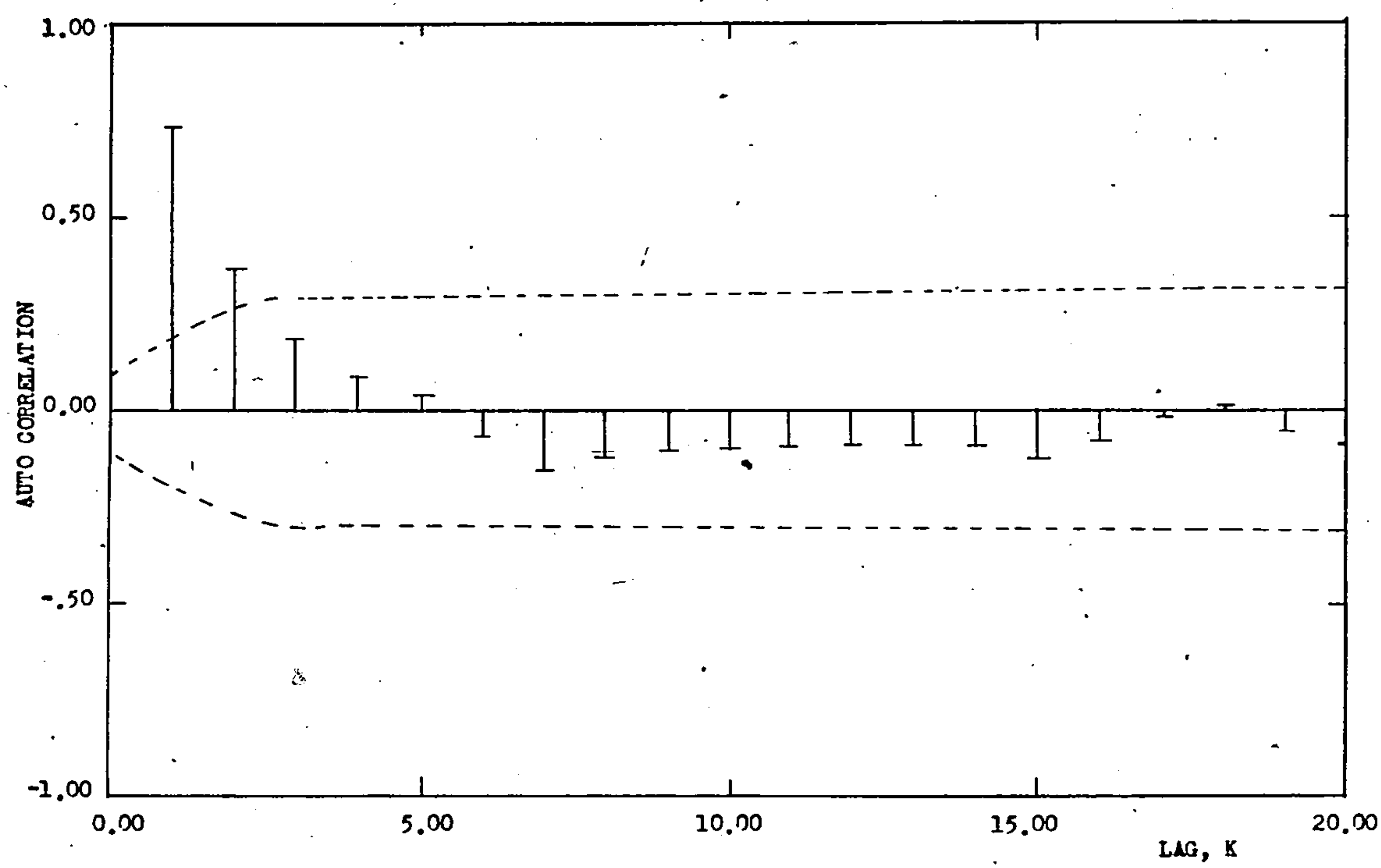
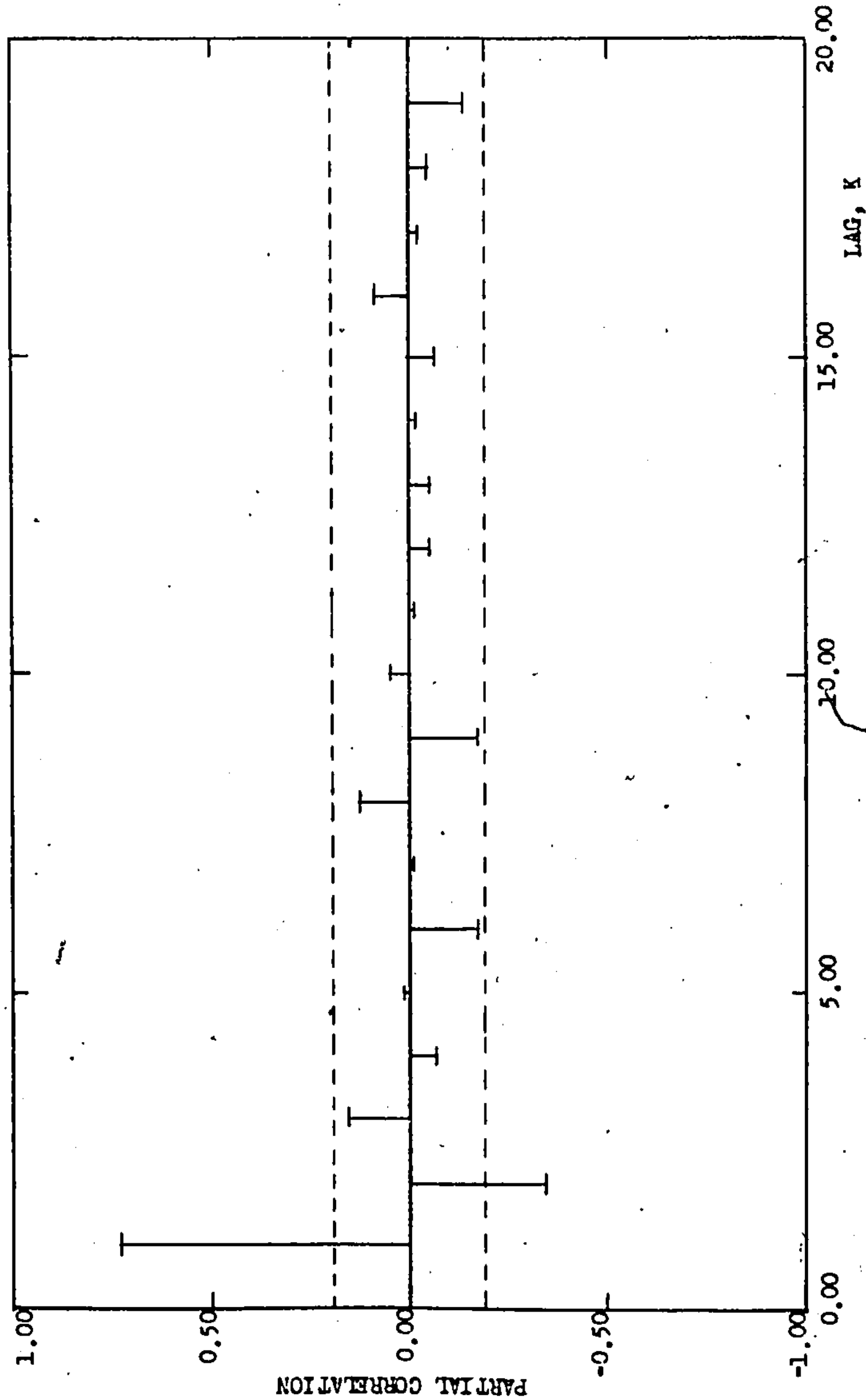


Figure 6:
PARTIAL CORRELATION VERSUS LAG, K ORIGINAL SERIES CONTROL VARIABLE



AUTO AND PARTIAL CORRELATIONS OF FIRST DIFFERENCES OF SERIES

I	AUTO	PARTIAL
1	0.159	0.159
2	-0.289	-0.322
3	-0.171	-0.068
4	-0.062	-0.124
5	0.080	0.045
6	-0.022	-0.127
7	-0.188	-0.175
8	0.006	0.032
9	0.026	-0.121
10	-0.027	-0.068
11	0.029	-0.016
12	0.010	-0.029
13	-0.008	-0.053
14	-0.001	-0.048
15	-0.124	-0.156
16	-0.001	-0.005
17	0.104	-0.019
18	0.136	0.111
19	-0.105	-0.197
20	-0.023	0.134

APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = 0.192

STANDARD DEVIATION OF SERIES = 0.1201823E+02

CHI-SQUARED STATISTIC = 0.2799E+02 BASED ON 20 DEGREES OF FREEDOM

Table 5 Identification of Control Series

Figure 7:
AUTO CORRELATION VERSUS LAG, K FIRST DIF SERIES CONTROL VARIABLE

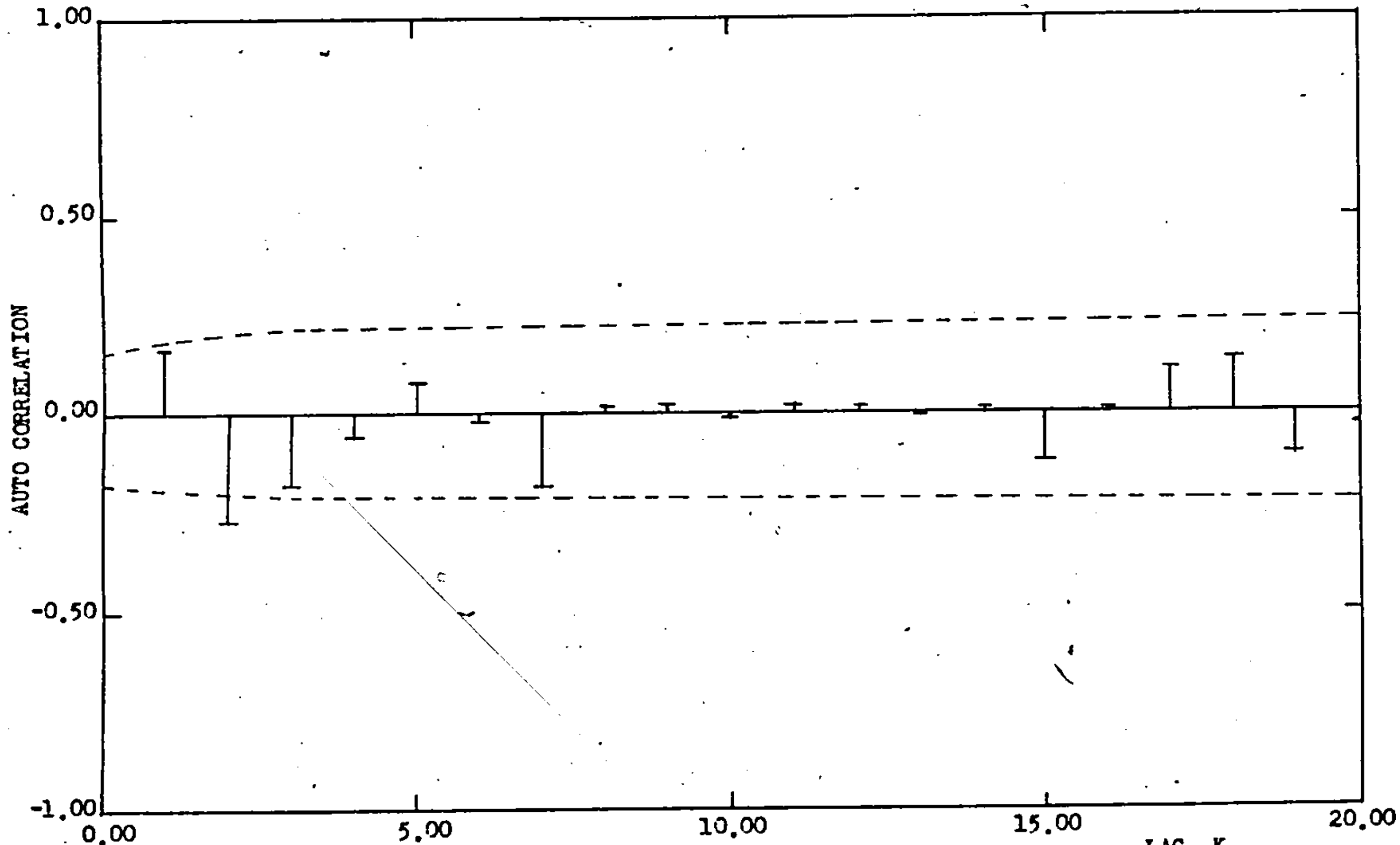
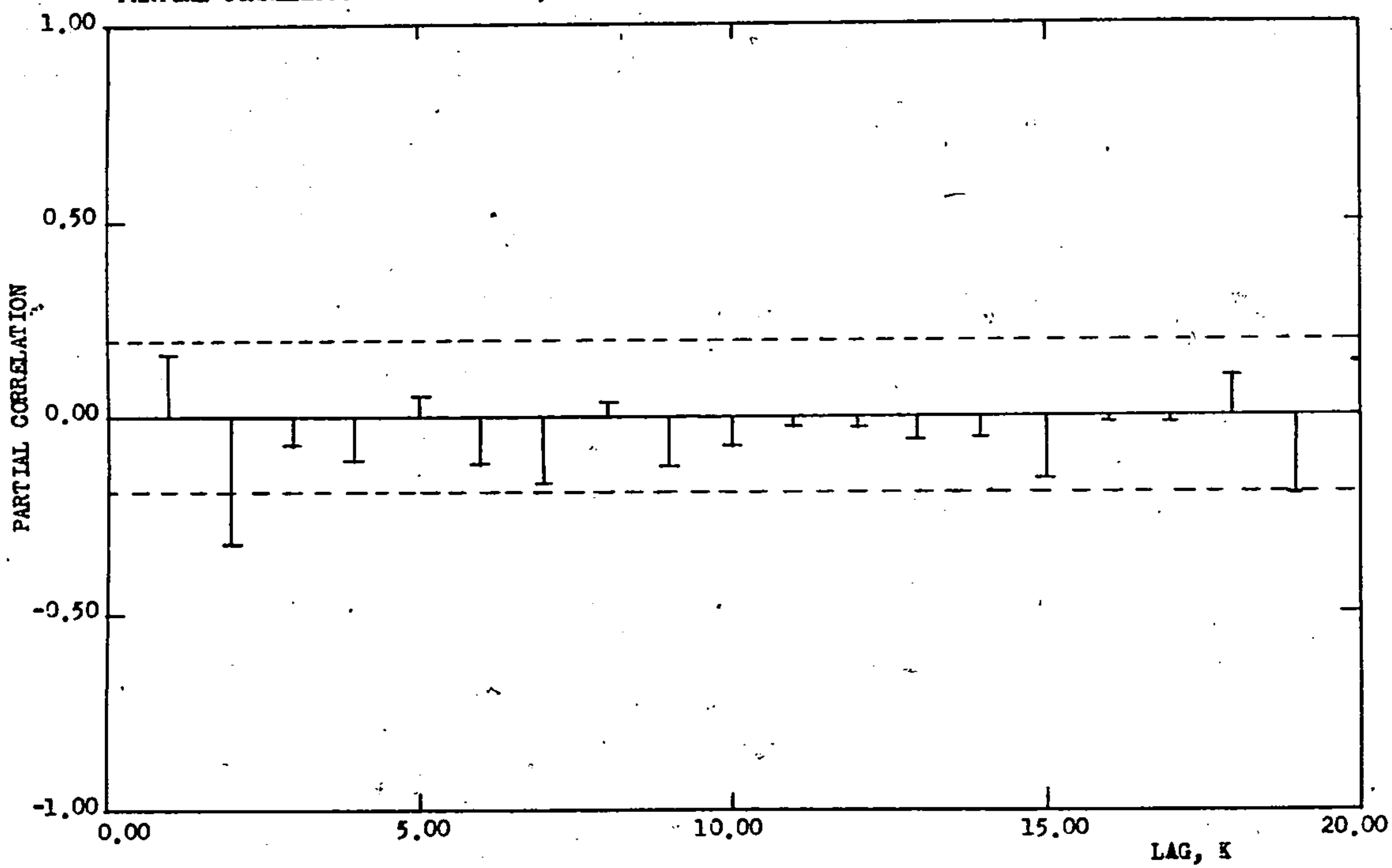


Figure 8:
PARTIAL CORRELATION VERSUS LAG, K FIRST DIF SERIES CONTROL VARIABLE



AUTO AND PARTIAL CORRELATIONS OF SECOND DIFFERENCES OF SERIES

I	AUTO	PARTIAL
1	-0.230	-0.230
2	-0.337	-0.412
3	0.004	-0.255
4	-0.021	-0.333
5	0.145	-0.114
6	0.043	-0.075
7	-0.220	-0.262
8	0.103	-0.078
9	0.046	-0.130
10	-0.064	-0.159
11	0.044	-0.126
12	0.000	-0.088
13	-0.019	-0.092
14	0.082	0.007
15	-0.141	-0.141
16	0.006	-0.109
17	0.043	-0.204
18	0.162	0.106
19	-0.193	-0.224
20	0.005	-0.029

APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = 0.193

STANDARD DEVIATION OF SERIES = 0.1565213E+02

CHI-SQUARED STATISTIC = 0.4026E+02 BASED ON 20 DEGREES OF FREEDOM

Table 6 Identification of Control Series

Figure 9:
AUTO CORRELATION VERSUS LAG, K SECOND DIF SERIES CONTROL VARIABLE

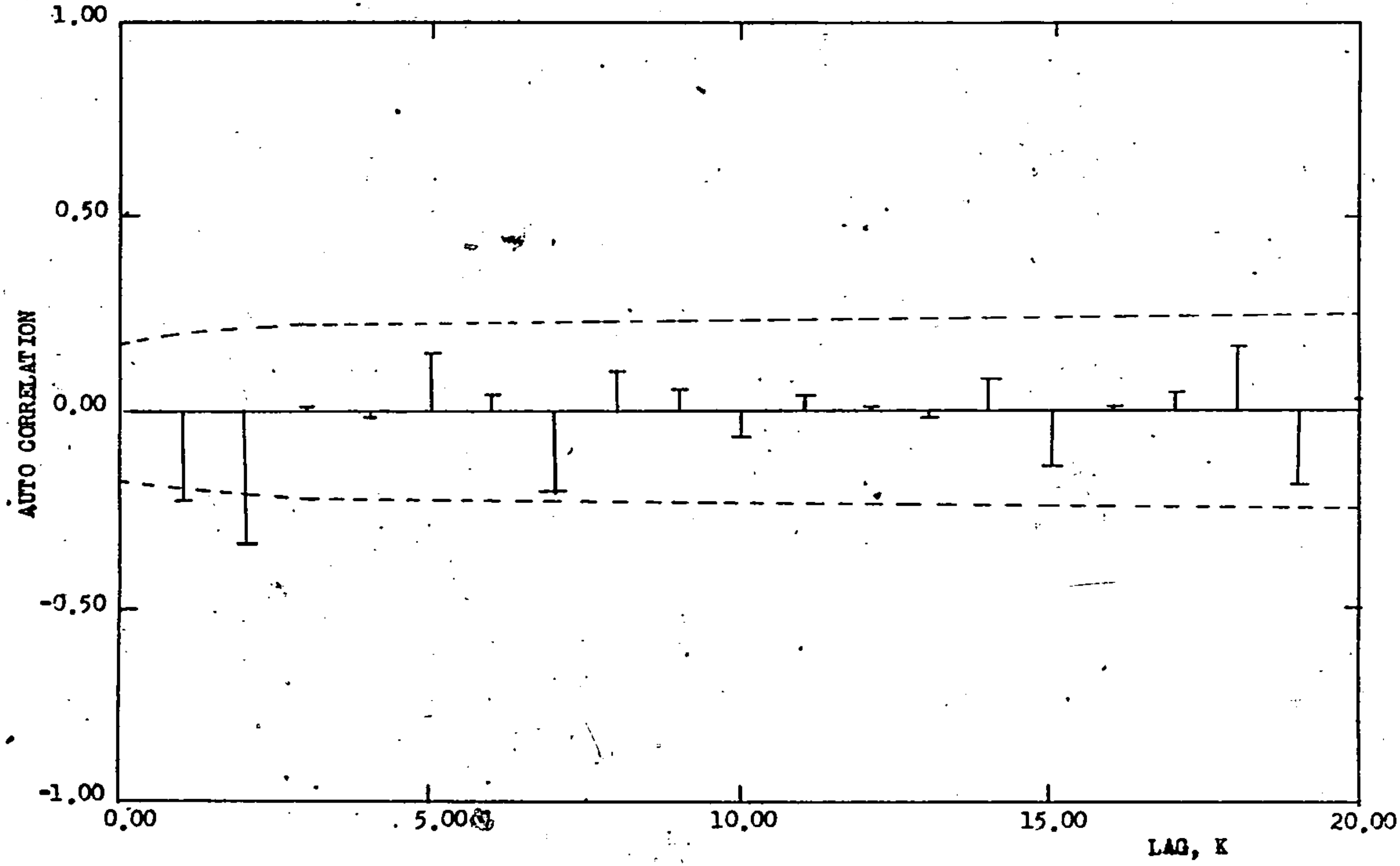
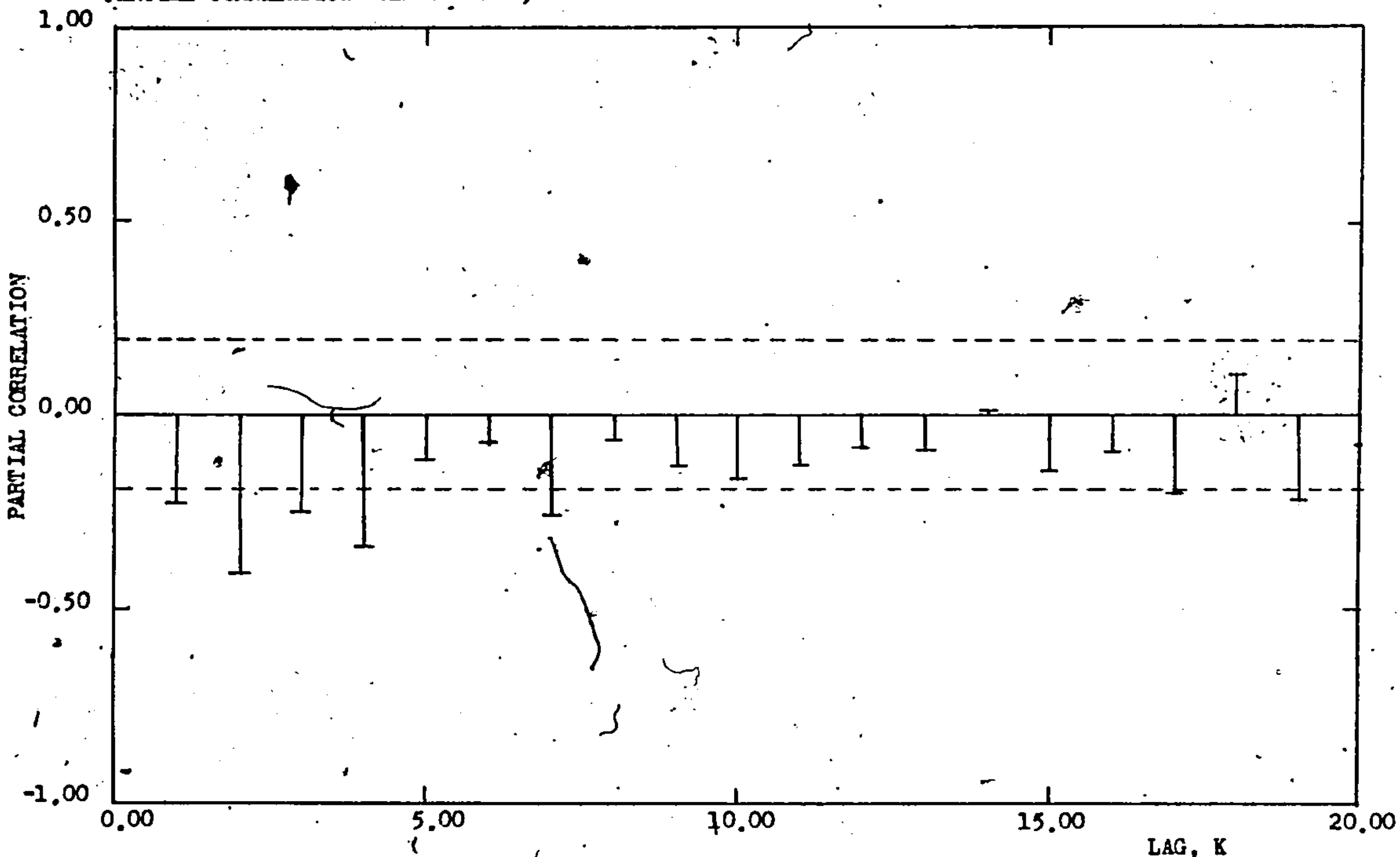


Figure 10:
PARTIAL CORRELATION VERSUS LAG, K SECOND DIF SERIES CONTROL VARIABLE



AUTO AND PARTIAL CORRELATIONS OF ORIGINAL SERIES

I	AUTO	PARTIAL
1	0.834	0.834
2	0.597	-0.319
3	0.447	0.203
4	0.341	-0.092
5	0.256	0.031
6	0.153	-0.156
7	0.064	0.046
8	0.038	0.073
9	0.018	-0.095
10	-0.007	0.035
11	-0.031	-0.044
12	-0.065	-0.054
13	-0.091	-0.022
14	-0.103	0.000
15	-0.119	-0.045
16	-0.085	0.176
17	-0.056	-0.145
18	-0.076	-0.038
19	-0.115	-0.090
20	-0.105	0.175

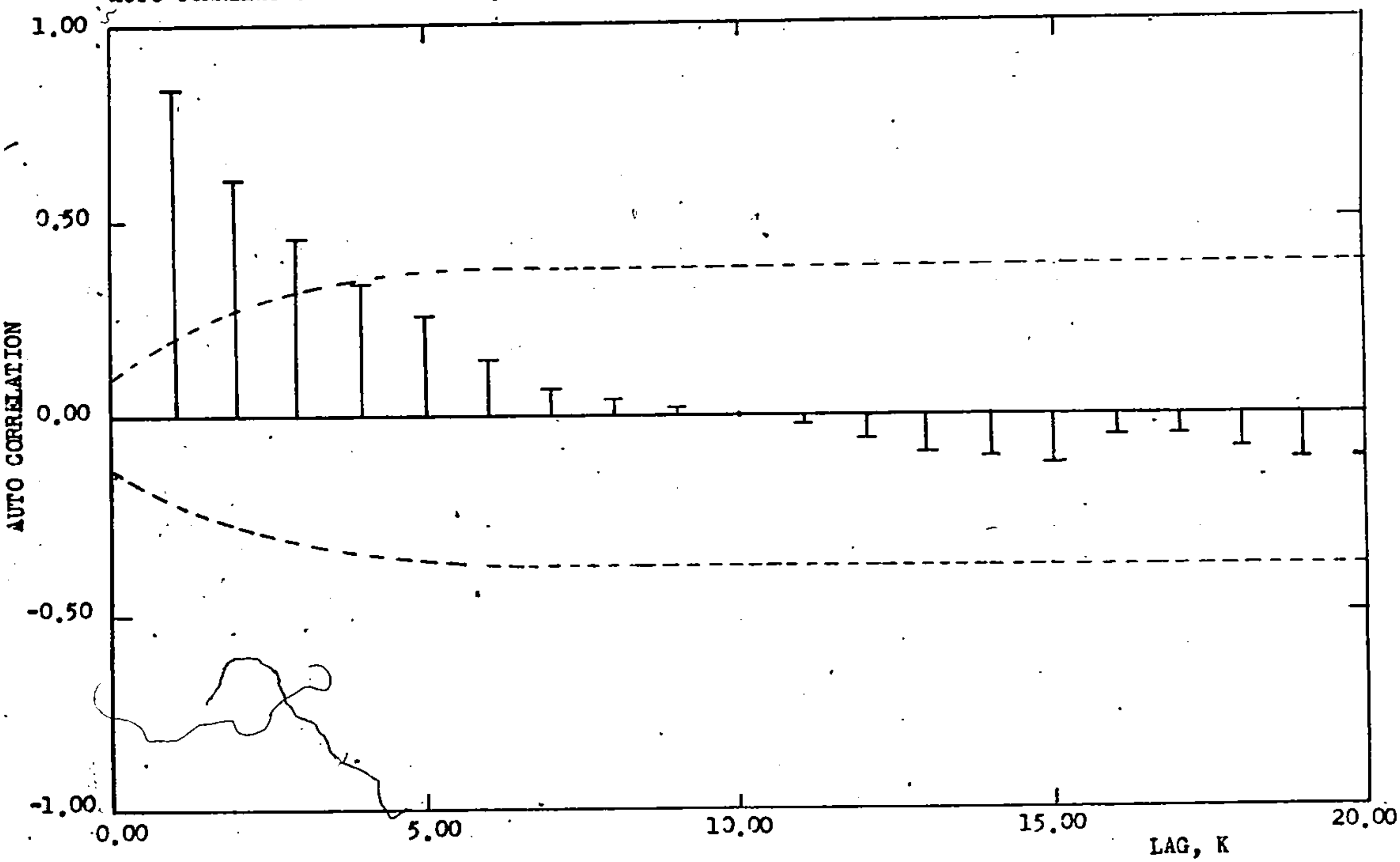
APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = 0.192

STANDARD DEVIATION OF SERIES = 0.1126568E+02

CHI-SQUARED STATISTIC = 0.1726E+03 BASED ON 20 DEGREES OF FREEDOM

Table 7 Identification of Output Measurement Series

Figure 11:
AUTO CORRELATION VERSUS LAG, K ORIGINAL SERIES OUTPUT MEASUREMENTS



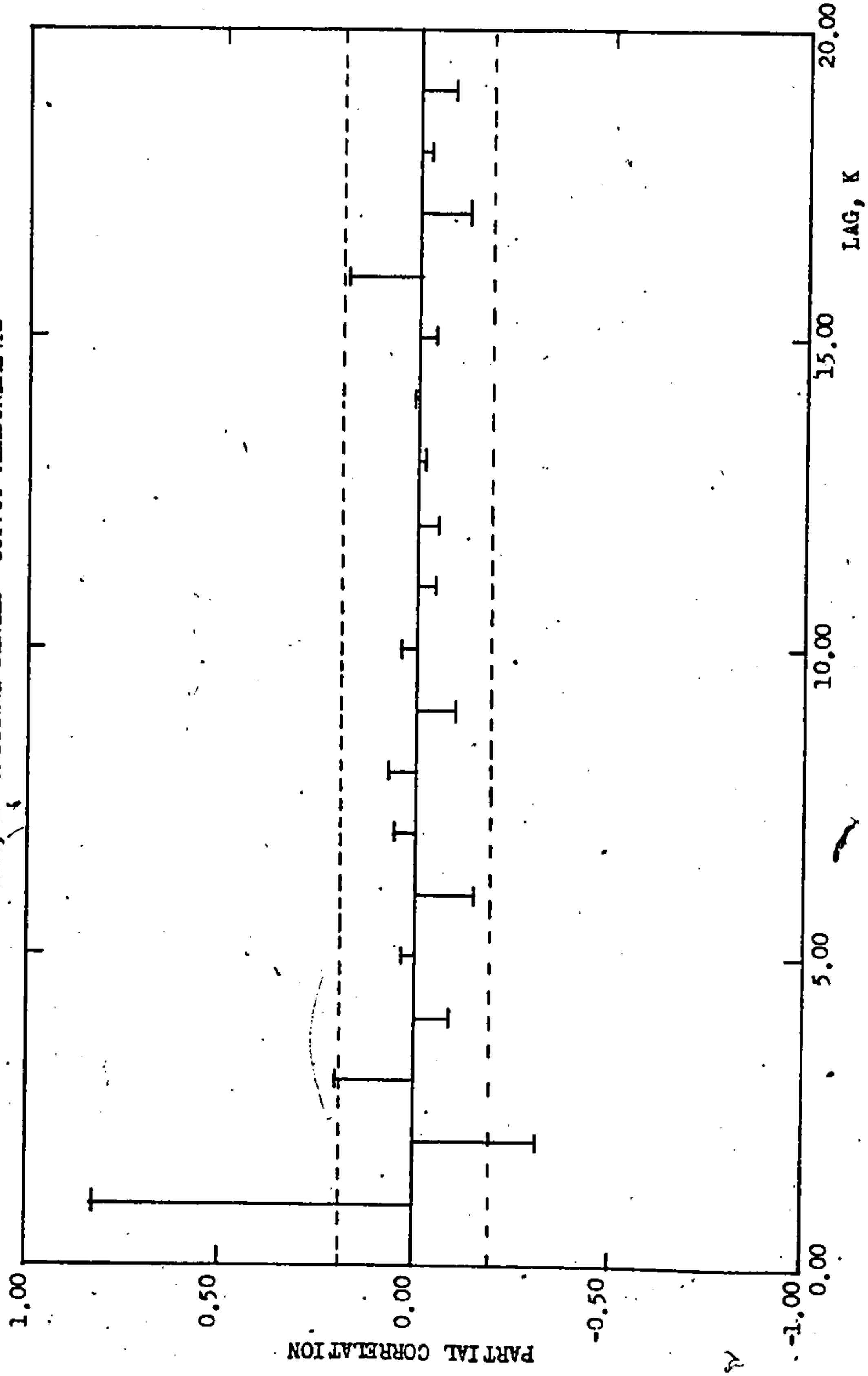


Figure 12:
PARTIAL CORRELATION VERSUS LAG, K ORIGINAL SERIES OUTPUT MEASUREMEYTS

AUTO AND PARTIAL CORRELATIONS OF FIRST DIFFERENCES OF SERIES

I	AUTO	PARTIAL
1	0.197	0.197
2	-0.271	-0.322
3	-0.151	-0.020
4	-0.032	-0.088
5	0.070	0.051
6	-0.031	-0.110
7	-0.198	-0.169
8	-0.026	0.032
9	0.052	-0.075
10	-0.013	-0.055
11	0.008	0.000
12	-0.014	-0.040
13	-0.029	-0.048
14	0.010	-0.035
15	-0.148	-0.203
16	-0.003	0.066
17	0.153	0.015
18	0.091	0.044
19	-0.153	-0.206
20	0.004	0.165

APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = 0.192

STANDARD DEVIATION OF SERIES = 0.6423900E+01

CHI-SQUARED STATISTIC = 0.3058E+02 BASED ON 20 DEGREES OF FREEDOM

Table 8 Identification of Output Measurement Series

Figure 13:
AUTO CORRELATION VERSUS LAG, K FIRST DIF SERIES OUTPUT MEASUREMENTS

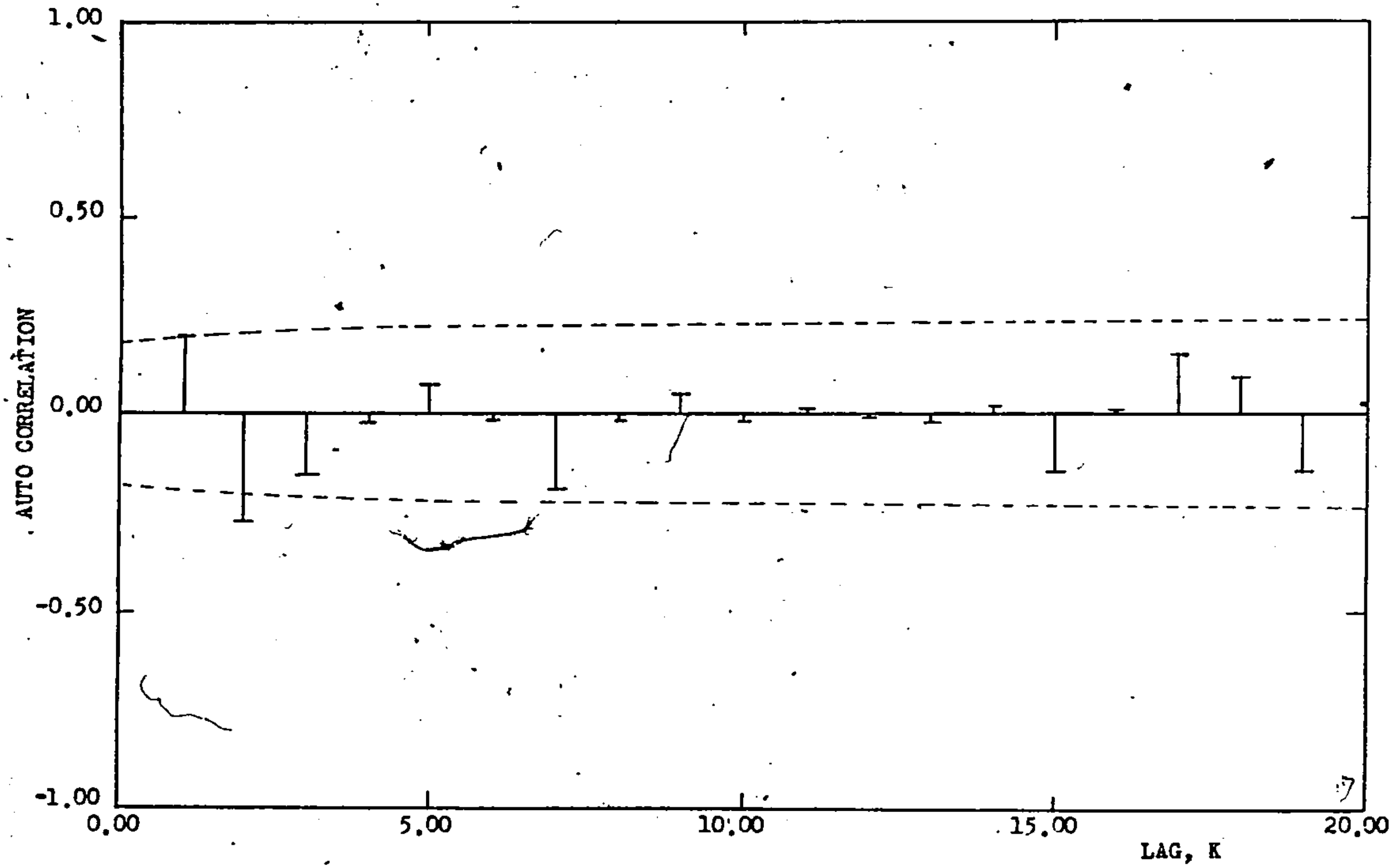
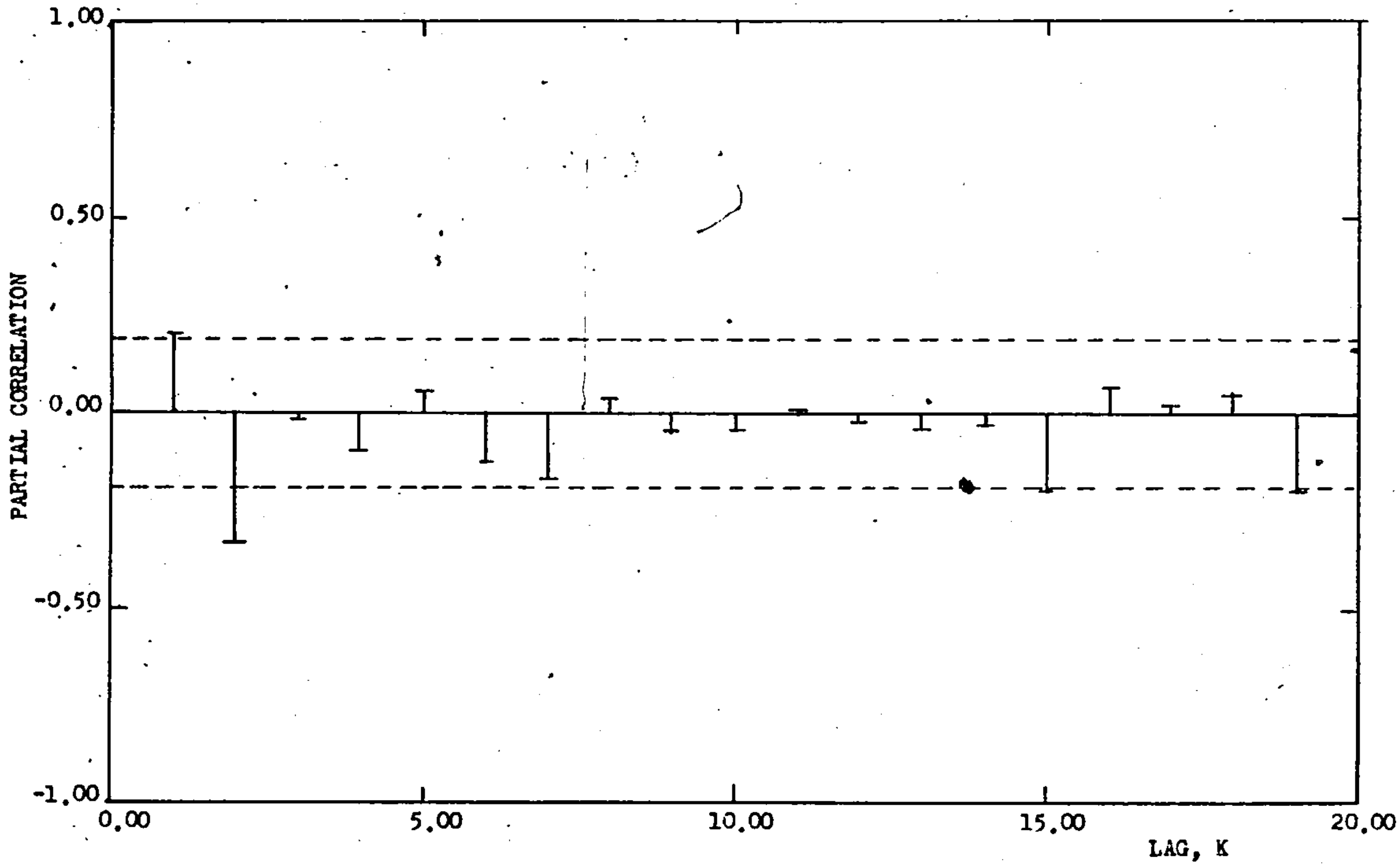


Figure 14:
PARTIAL CORRELATION VERSUS LAG, K FIRST DIF SERIES OUTPUT MEASUREMENTS



AUTO AND PARTIAL CORRELATIONS OF SECOND DIFFERENCES OF SERIES

I	AUTO	PARTIAL
1	-0.202	-0.202
2	-0.370	-0.429
3	0.018	-0.225
4	0.006	-0.289
5	0.122	-0.068
6	0.035	-0.042
7	-0.216	-0.220
8	0.077	-0.057
9	0.074	-0.100
10	-0.055	-0.108
11	0.033	-0.057
12	-0.006	-0.037
13	-0.035	-0.051
14	0.121	0.090
15	-0.208	-0.211
16	0.012	-0.086
17	0.158	-0.079
18	0.100	0.168
19	-0.252	-0.204
20	0.036	0.060

APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = 0.193

STANDARD DEVIATION OF SERIES = 0.8133812E+01

CHI-SQUARED STATISTIC = 0.4862E+02 BASED ON 20 DEGREES OF FREEDOM

Table 9 Identification of Output Measurement Series

Figure 15:
AUTO CORRELATION VERSUS LAG, K SECOND DIF SERIES OUTPUT MEASUREMENTS

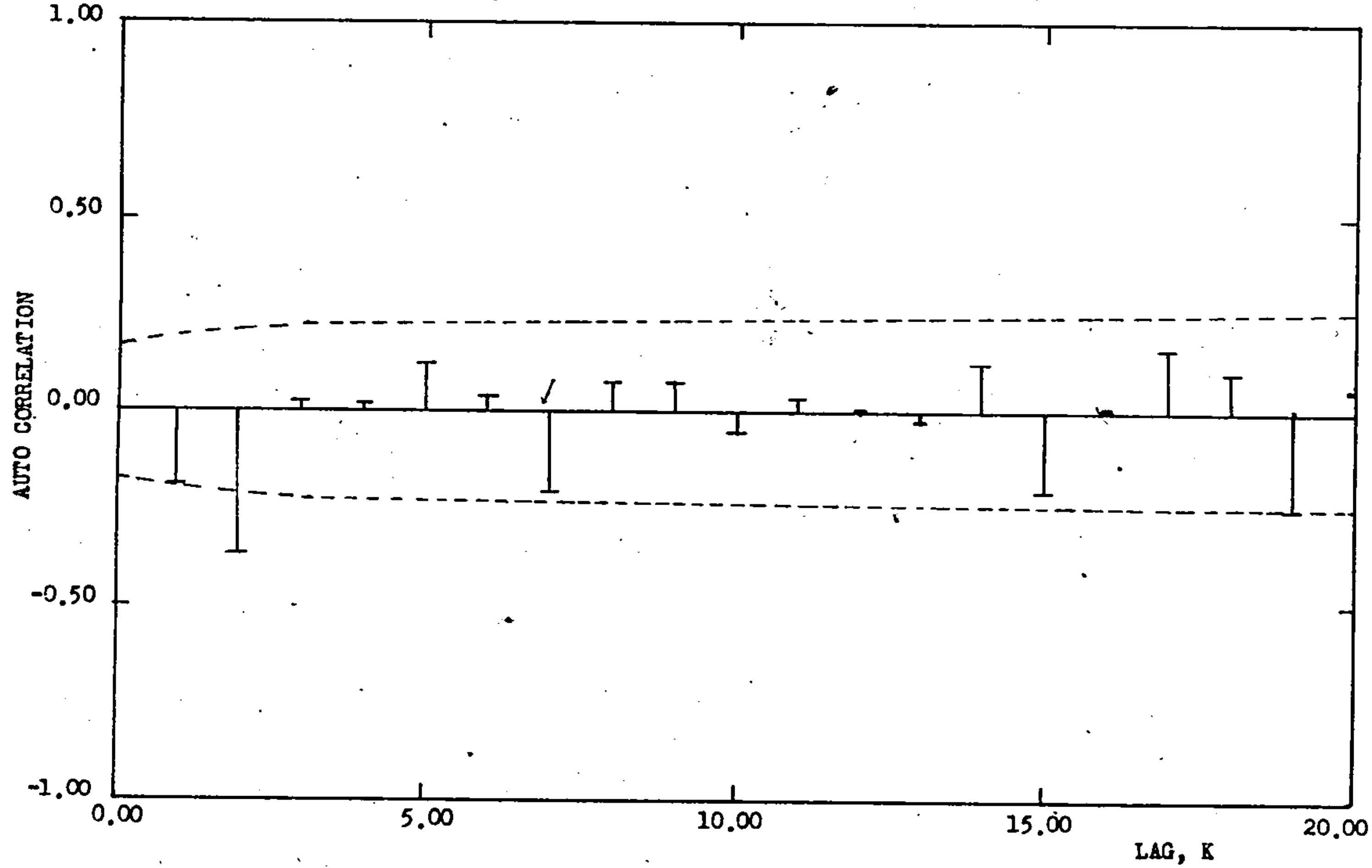
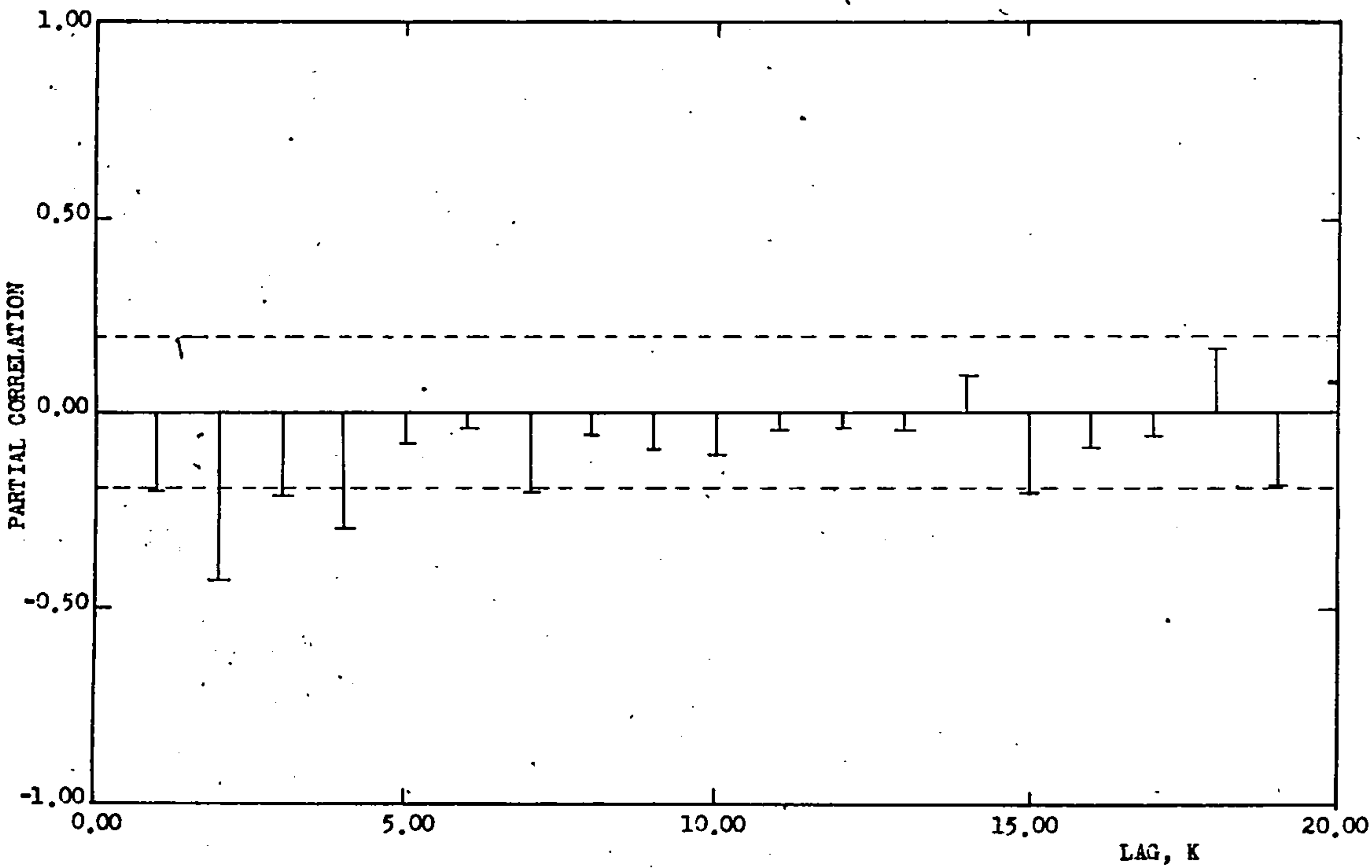


Figure 16:
PARTIAL CORRELATION VERSUS LAG, K SECOND DIF SERIES OUTPUT MEASUREMENTS



LAG,K	IMPULSE	STEP
0	0.2999E-01	0.2999E-01
1	-0.1754E-01	0.1245E-01
2	-0.5068E+00	-0.4944E+00
3	-0.6230E-01	-0.5567E+00
4	-0.5969E-01	-0.6164E+00
5	-0.6602E-01	-0.6824E+00
6	-0.4832E-01	-0.7307E+00
7	-0.6014E-01	-0.7908E+00
8	-0.2661E-01	-0.8174E+00
9	-0.3567E-01	-0.8531E+00
10	-0.2785E-01	-0.8810E+00
11	-0.3719E-01	-0.9182E+00
12	-0.3462E-01	-0.9528E+00
13	-0.5539E-01	-0.1008E+01
14	-0.3460E-01	-0.1043E+01
15	-0.4689E-02	-0.1047E+01
16	-0.4030E-01	-0.1088E+01
17	0.5592E-02	-0.1082E+01
18	-0.2242E-01	-0.1105E+01
19	-0.1861E-01	-0.1123E+01
20	-0.5014E-01	-0.1173E+01

Note that both of the original series were differenced from their means for this calculation.

Table 10 Impulse and Step Response of the Original U and Original Y series.

Figure 17:
IMPULSE WEIGHTS VERSUS LAG, K ORIGINAL OUTPUT SERIES

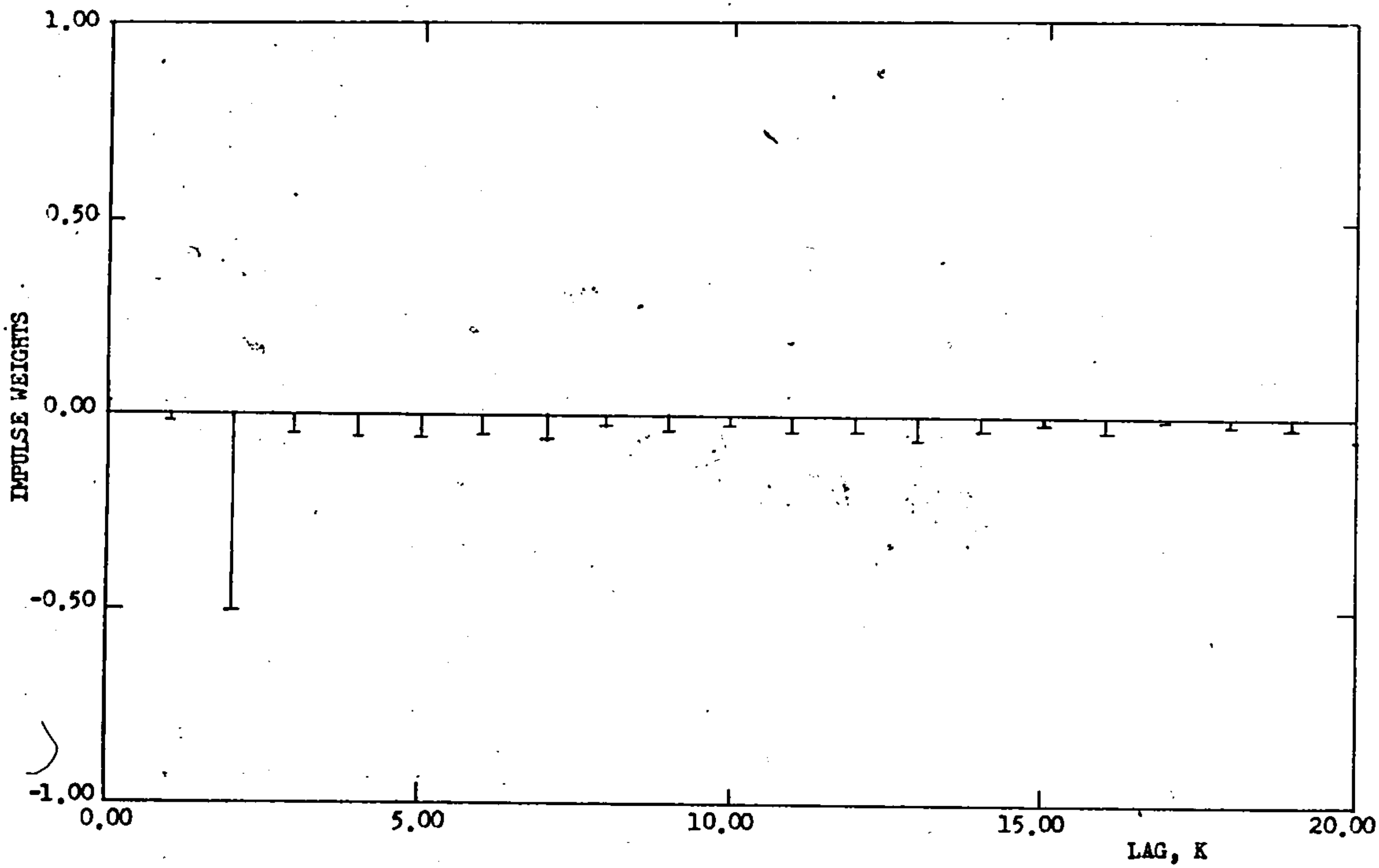
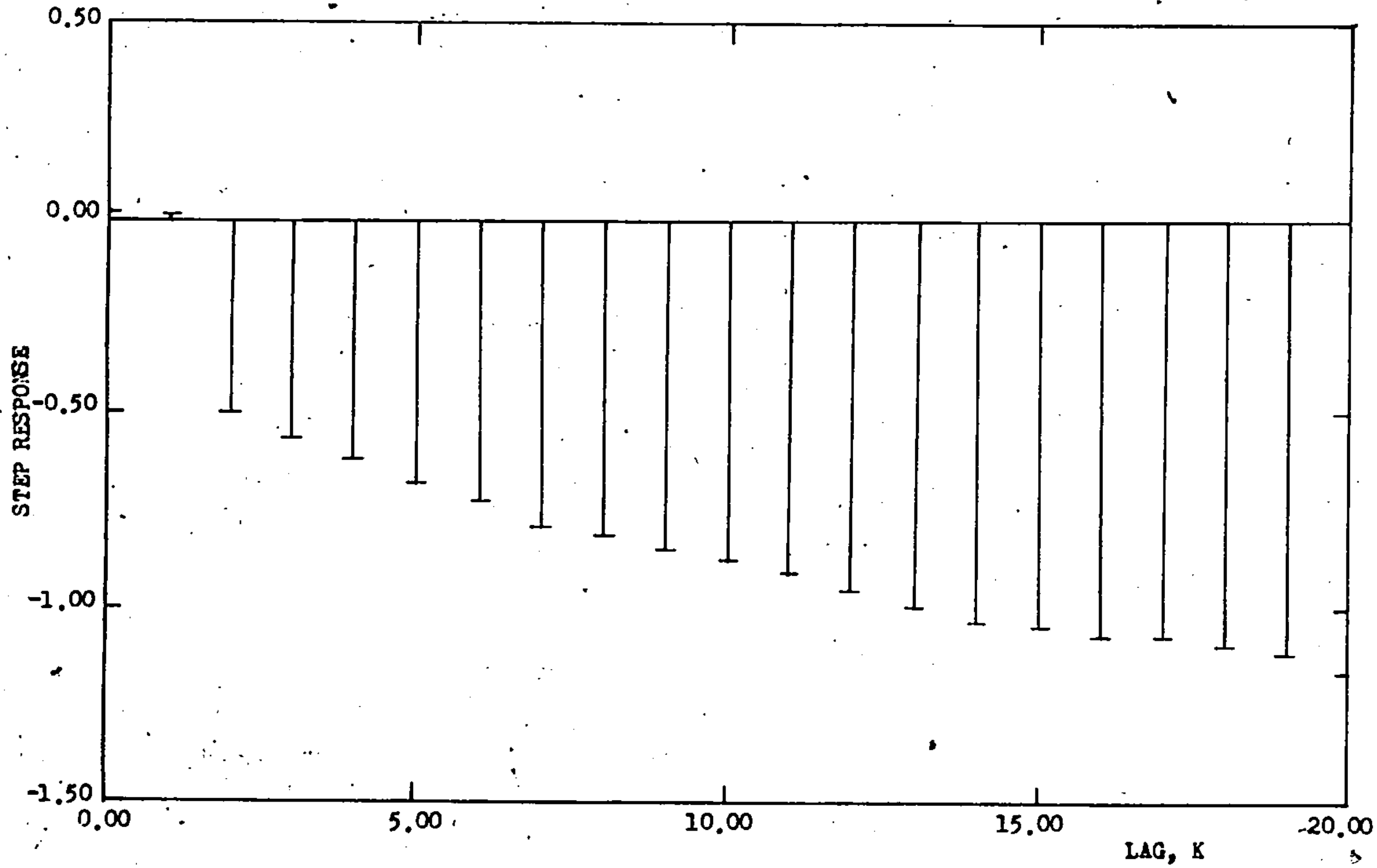


Figure 18:
STEP RESPONSE VERSUS LAG, K ORIGINAL OUTPUT SERIES



LAG,K	IMPULSE	STEP
0	0.2089E-01	0.2089E-01
1	-0.5396E-02	0.1550E-01
2	-0.5092E+00	-0.4937E+00
3	-0.6445E-01	-0.5582E+00
4	-0.4351E-01	-0.6017E+00
5	-0.5591E-01	-0.6576E+00
6	-0.5510E-01	-0.7127E+00
7	-0.4533E-01	-0.7580E+00
8	-0.2828E-01	-0.7863E+00
9	-0.2662E-01	-0.8129E+00
10	-0.1976E-01	-0.8327E+00
11	-0.2811E-01	-0.8608E+00
12	-0.4215E-01	-0.9029E+00
13	-0.2088E-01	-0.9238E+00
14	-0.3091E-01	-0.9547E+00
15	-0.8718E-02	-0.9634E+00
16	-0.2420E-01	-0.9876E+00
17	0.1466E-01	-0.9730E+00
18	-0.1511E-01	-0.9881E+00
19	-0.2670E-01	-0.1015E+01
20	-0.1505E-01	-0.1030E+01

Table 11 Impulse and Step Response of the First Differenced U Series and the First Differenced Y Series

Figure 19:
IMPULSE WEIGHTS VERSUS LAG, K FIRST DIF OUTPUT SERIES

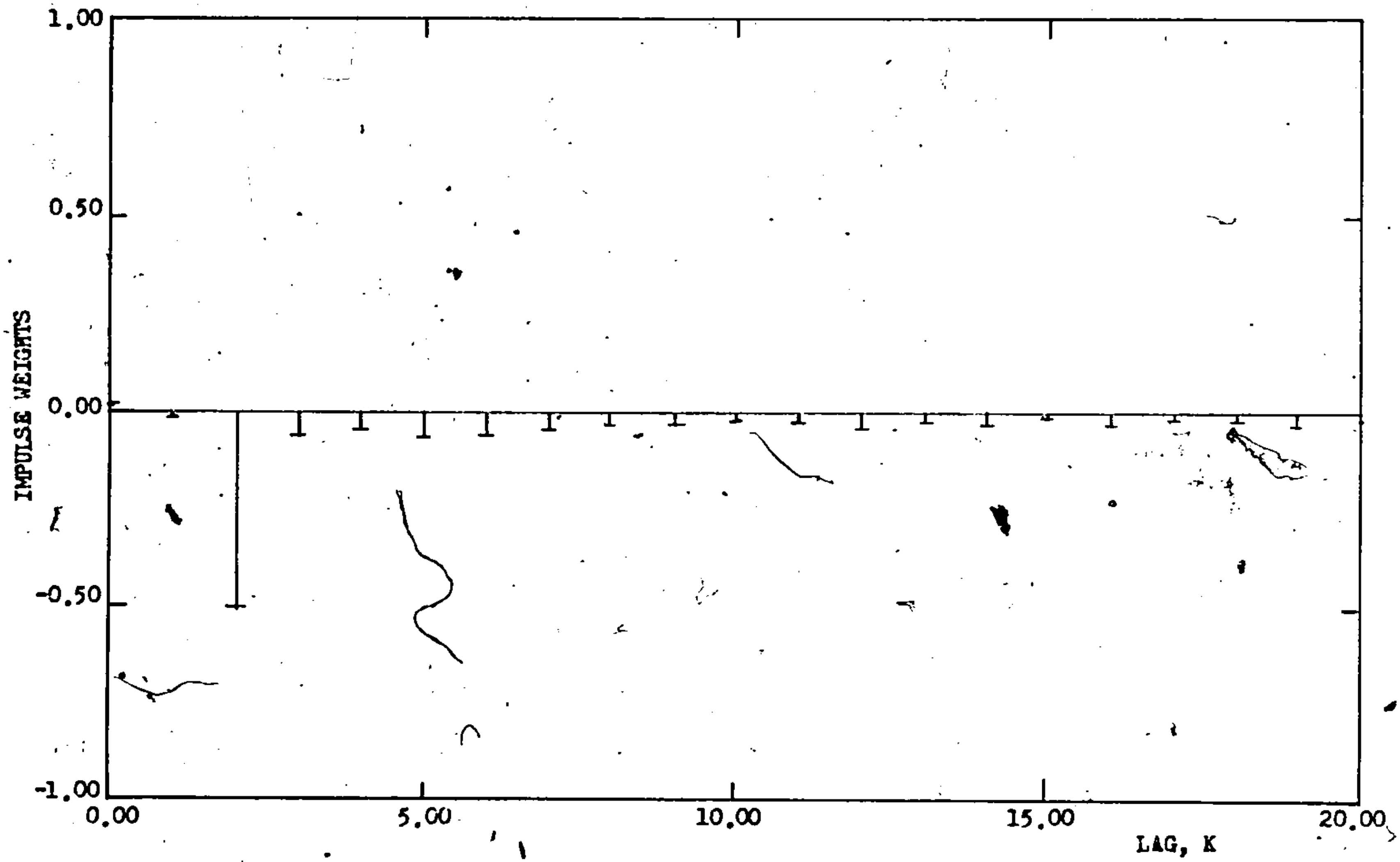
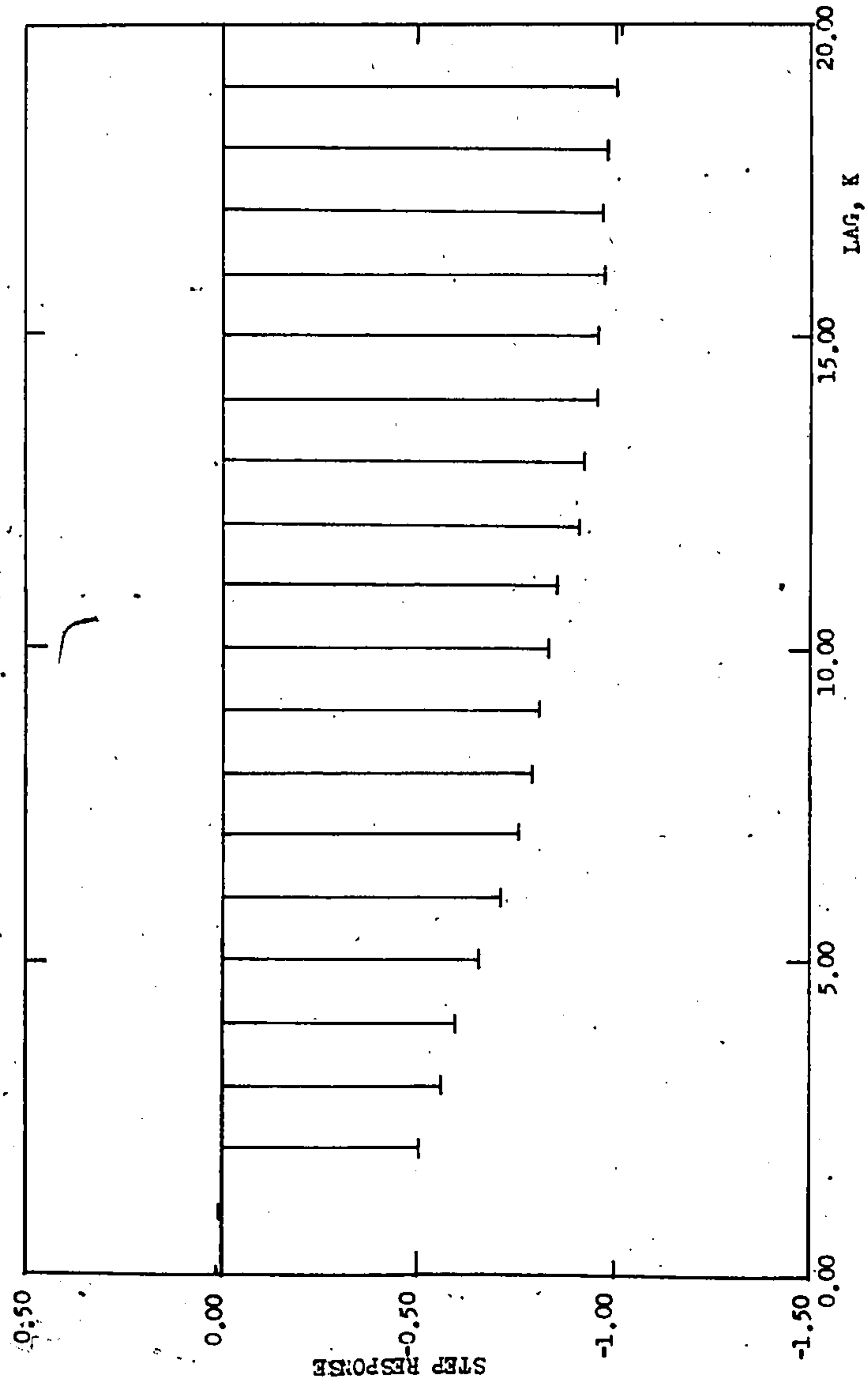


Figure 20:
STEP RESPONSE VERSUS LAG, K FIRST DIF OUTPUT SERIES



CROSS-CORRELATIONS BETWEEN OBSERVATIONS AND MANIPULATED VARIABLES   Y(T)*U(T+K)

0	-0.278	0	-0.278
-1	-0.593	1	-0.088
-2	-0.854	2	0.025
-3	-0.723	3	0.094
-4	-0.493	4	0.174
-5	-0.371	5	0.236
-6	-0.309	6	0.224
-7	-0.257	7	0.214
-8	-0.158	8	0.210
-9	-0.059	9	0.196
-10	-0.060	10	0.195
-11	-0.067	11	0.183
-12	-0.054	12	0.151
-13	-0.047	13	0.133
-14	-0.028	14	0.076
-15	0.001	15	0.037
-16	0.028	16	0.049
-17	0.074	17	0.079
-18	0.050	18	0.052
-19	-0.011	19	0.024
-20	-0.022	20	0.032

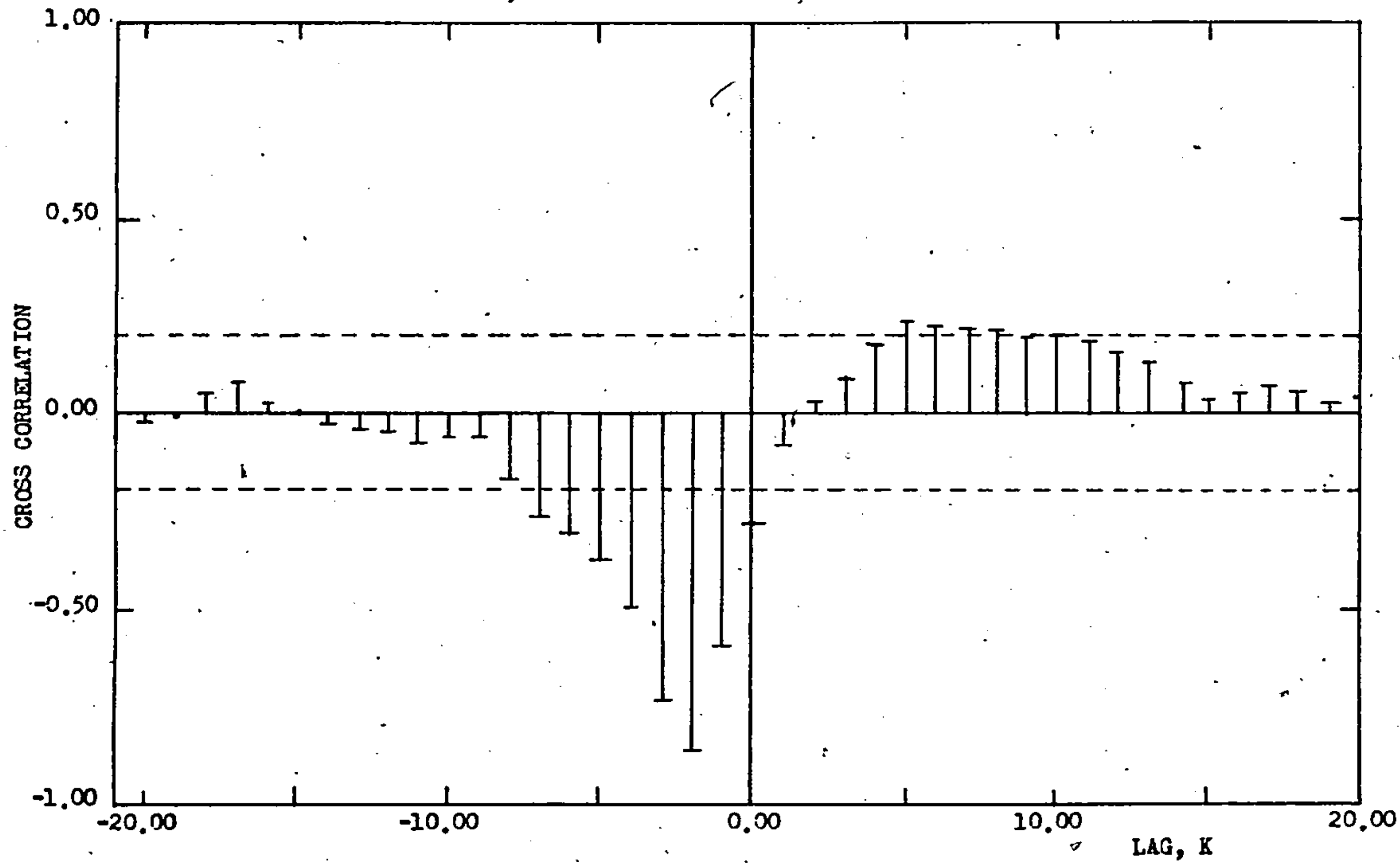
APPROX. 95 PERCENT CONF. LIMIT ON CROSS-CORRELATIONS = 0.192

STANDARD DEVIATIONS S(Y) = 0.1127E+02   S(U) = 0.1673E+02

CHI-SQUARED STATISTIC = 57.52
BASED ON (21 - NO. OF DYNAMIC PARAMETERS) DEGREES OF FREEDOM

Table 12 Cross Correlation Function of the Original U and Original Y Series

Figure 21:
CROSS CORRELATION VERSUS LAG, K ORIGINAL OUTPUT SERIES



Certain information is readily available from these preliminary evaluation techniques. The plots of the U and Y series are not deterministic in the manner of of a step function, but they both appear to be stochastic in nature. The magnitudes of the Auto and Partial Auto Correlations are greatly reduced by the first differencing of both series. The Partial Auto Correlation of the second differencing of both series are large and negative, which signifies too much differencing. Therefore, both series require first differencing in the model (d=1).

The Impulse response function in both zero and first differences of the series shows no response until the second lag. Therefore, the system model requires a lag of two (b=2). Note that the Step function is the sum of the Impulse function up to each lag. It is provided because many engineers think in terms of the step response of a system, rather than the impulse response.

It is possible for the user to derive values for the impulse function from the cross correlation function if the U series is white noise. Box and Jenkins[1] show that an ARIMA prewhitening transform for the U series may also be applied to the Y series. Then the cross correlation function of the transformed U and Y series will meet the impulse relation requirements. Note that both the computer and the user generated impulse values are approximations of the true system response.

The cross correlation function also indicates that the past values of the control action had an effect on the present observation value. This confirms the selection of the control variable as reasonable. It does not indicate that the best control variable available is in use in the system. This aspect of control engineering should be addressed before the model building stage of the design.

In diagnostic terms, the techniques of time series analysis indicate that the form and initial parameter estimates of an ARIMA model are set by the Auto and Partial Auto Correlation functions. Similarly, the impulse response function provides the form and initial parameter estimates of a transfer function model. The development of a transfer function model is always addressed before the fitting of a stochastic model to the residuals (i.e. the Noise Model) of the selected transfer function model for a given set of data.

2.4.2 The Transfer Function Model

The impulse and step response functions for various transfer function models have been tabulated by Box and Jenkins[1]. Relations are provided which allow the calculation of preliminary estimates of model parameters from the values of the impulse weights for any data set. By a comparison of the observed impulse weights to those tabulated, it is possible to entertain various model forms of the generalized model.

The Marquardt Nonlinear Least Squares procedure may then be used interactively to obtain the best parameter estimates for each model in turn. The user may observe the cross correlation function between the model residuals and the control action. When these cross correlations are within the 95% confidence limits, the transfer function model may be termed adequate. For these confidence limits, statistically one observation in twenty should lie outside the limits. In the case of several adequate models, the user may consult the variance of the residuals to determine the transfer function with the better overall fit to the data. The idea of a parsimonious model (i.e. the model with fewest parameters for best fit) should be foremost in the mind of the user during this stage of the model building process.

A special warning to the user is required at this point. The numerical package will converge for any model only if the initial parameter estimates are reasonable. The roots of any polynomial in the backwards operator (B) must lie outside the unit circle or the numerical equations will rapidly become unstable. Furthermore, the Box and Jenkins parameter estimation equations are based on polynomials in (B) of the form:

$$\omega(B) = \omega_0 - \omega_1 B - \omega_2 B^2 - \ldots$$

$$\delta(B) = 1 - \delta_1 B - \delta_2 B^2 - \ldots$$

The computer program is based on polynomials in (B) of the form:

$$\omega(B) = \omega_0 + \omega_1 B + \omega_2 B^2 + \ldots$$

$$\delta(B) = 1 + \delta_1 B + \delta_2 B^2 + \ldots$$

Therefore, any parameter estimated using the equations of Box and Jenkins should be multiplied by -1 for input to this time series package. The only exception is ω_0 which is equal in the two cases.

Five initial model forms were selected as reasonable for this data set, and the initial parameter estimates were determined for each of the models. The results are given in Table 13. Each of these models was tested by the computer.

No.	Model Form	δ_1	δ_2	ω_0	ω_1	ω_2
		Parameter Estimate				
1	$Y_t = (\omega_0 + \omega_1 B)\ U_{t-2}$	-	-	-0.51	0.07	-
2	$(1+\delta_1 B) Y_t = \omega_0\ U_{t-2}$	-0.13	-	-0.51	-	-
3	$(1+\delta_1 B) Y_t = (\omega_0+\omega_1 B)\ U_{t-2}$	-0.68	-	-0.51	0.28	-
4	$(1+\delta_1 B) Y_t = (\omega_0+\omega_1 B+\omega_2 B^2) U_{t-2}$	-.99	-	-0.51	0.44	0.008
5	$(1+\delta_1 B+\delta_2 B^2) Y_t = (\omega_0+\omega_1 B) U_{t-2}$	-1.01	0.02	-0.51	0.45	-

Table 13 Initial Models and Parameter Estimates Calculated from Impulse Weights

Note that both the U and Y series have been differenced once for all the models of Table 13.

The initial estimates of the second order terms in both models 4 and 5 are very close to zero. The numerical package predictably estimated the confidence limits on these parameters to include zero. Thus, the addition of the extra term did not improve the model behavior over Model 3, and the calculated value of that extra term was not significant. Note that all of the initial parameter values of model 3 are significantly non-zero.

It is important to note that for models 1, 2, and 3, the cross correlation curves were reasonably within the 95%

confidence limits. This test is observed with the best parameter estimates obtained for each model by the numerical package. Therefore, the better model of the three cases was determined by the best Chi-Squared test parameter which reflects the better overall fit of a model. Model 3 was observed to have the best overall behavior, and all of the best parameters observed had reasonable confidence limits.

The model parameters required between 15 and 20 iterations to converge on the best parameter estimates from the initial parameter values of table 13. The full set of diagnostics for model 3 have been recorded from the improved starting parameters ($\omega_0=-0.52$, $\omega_1=0.32$, $\delta_1=-0.72$), which served to lower the number of iterations and still fully demonstrate the numerical package. In this run of the numerical package ω_0 is parameter 1, ω_1 is parameter 2, and δ_1 is parameter ~~3.~~ The parameters are always ordered in the sequence input by the user to the interactive prompts for the model, when they are transferred to the numerical routines. This sequence will vary depending on the orders of the polynomials selected.

The user should keep track of the differencing and the number of lags selected for the model being tested. In this case, first order differencing and two lags were used.

2.4.3 Numerical Package Analysis of the Transfer Function

$$Y_t = \frac{(\omega_0 + \omega_1 B)}{(1 + \delta_1 B)} U_{t-2} + \frac{N_t}{\nabla} \qquad (2)$$ Test Model Form.

NON-LINEAR ESTIMATION, PROBLEM NUMBER 0

108 OBSERVATIONS, 3 PARAMETERS 456 SCRATCH REQUIRED

INITIAL PARAMETER VALUES

1	2	3
-0.5200E+00	0.3200E+00	-0.7200E+00
ω_0	ω_1	δ_1

INITIAL SUM OF SQUARES = 0.2136E+03

ITERATION NO. 1

DETERMINANT = 0.4506E-01 ANGLE IN SCALED COORD =63.87DEGREES

TEST POINT PARAMETER VALUES

1	2	3
-0.5183E+00	0.3249E+00	-0.7243E+00

TEST POINT SUM OF SQUARES = 0.2134E+03

PARAMETER VALUES VIA REGRESSION

1	2	3
-0.5183E+00	0.3249E+00	-0.7243E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.2133531E+03

ITERATION NO. 2

DETERMINANT = 0.4376E-01 ANGLE IN SCALED COORD =51.29DEGREES

TEST POINT PARAMETER VALUES

```
      1            2            3
-0.5182E+00  0.3218E+00 -0.7193E+00
```

TEST POINT SUM OF SQUARES = 0.2133E+03

PARAMETER VALUES VIA REGRESSION

```
      1            2            3
-0.5182E+00  0.3218E+00 -0.7193E+00
```

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.2132906E+03

ITERATION NO. 3

DETERMINANT = 0.4298E-01 ANGLE IN SCALED COORD =75.44DEGREES

TEST POINT PARAMETER VALUES

```
      1            2            3
-0.5180E+00  0.3212E+00 -0.7182E+00
```

TEST POINT SUM OF SQUARES = 0.2133E+03

PARAMETER VALUES VIA REGRESSION

```
      1            2            3
-0.5180E+00  0.3212E+00 -0.7182E+00
```

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.2132787E+03

ITERATION NO. 4

DETERMINANT = 0.4267E-01 ANGLE IN SCALED COORD =74.56DEGREES

TEST POINT PARAMETER VALUES

1	2	3
-0.5180E+00	0.3209E+00	-0.7175E+00

TEST POINT SUM OF SQUARES = 0.2133E+03

PARAMETER VALUES VIA REGRESSION

1	2	3
-0.5180E+00	0.3209E+00	-0.7175E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.2132723E+03

ITERATION NO. 5

DETERMINANT = 0.4253E-01 ANGLE IN SCALED COORD =76.65DEGREES

TEST POINT PARAMETER VALUES

1	2	3
-0.5179E+00	0.3207E+00	-0.7172E+00

TEST POINT SUM OF SQUARES = 0.2133E+03

PARAMETER VALUES VIA REGRESSION

1	2	3
-0.5179E+00	0.3207E+00	-0.7172E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.2132698E+03

ITERATION STOPS - RELATIVE CHANGE IN EACH PARAMETER LESS THAN 0.001

FINAL RESIDUAL VALUES

0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.3888E+01	0.2282E+01	0.1891E+01	0.1888E+01	0.1152E+01	-0.1929E+00
0.1776E+00	-0.3251E+00	-0.5333E+00	0.4344E+00	-0.9622E+00	0.1364E+01	-0.1322E+00	-0.9716E+00	-0.1034E+01	-0.6099E-01
-0.3973E-01	-0.1407E+01	0.4077E+00	-0.4296E+00	-0.4447E+00	0.1207E+01	-0.2968E+00	-0.2555E+00	0.8776E-01	0.1008E+00
0.5723E+00	0.7185E+00	0.4408E+01	0.2138E+01	-0.2467E+01	0.6622E+00	0.3173E+01	-0.5240E+00	-0.5792E+00	0.2156E-01
0.7155E-01	0.6572E+00	-0.5730E+00	-0.2868E-01	0.6049E+00	0.2508E+01	-0.6460E+00	-0.4129E+01	-0.8390E+00	-0.1642E+01
-0.3006E+01	-0.1087E+00	-0.2251E+01	0.3256E+01	-0.1789E+01	-0.9867E+00	0.1007E+01	-0.9925E+00	-0.6995E+00	-0.4438E+00
-0.8364E-01	-0.4065E+00	0.4397E+00	0.7363E+00	0.1344E+01	-0.2061E+01	-0.7197E+00	0.3130E+00	-0.1979E+00	0.1335E+01
0.5850E+01	0.7066E+00	-0.2605E+00	0.1474E+01	0.9891E+00	0.1128E+01	0.5761E+00	0.5937E+00	0.3135E+00	-0.2430E+00
0.5923E+00	-0.8338E+00	0.1929E-01	0.2653E+00	-0.3022E+00	-0.1320E-01	-0.1882E+00	-0.2192E+00	0.9030E+00	0.2914E+01
-0.1232E+01	-0.12[illegible]	-0.2116E+01	0.9342E-01	-0.1009E+01	0.9390E+00	0.4562E+00	-0.4345E-01	0.6955E+00	0.3493E-02
0.5282E+00	[illegible]	[illegible]1453E-01	0.1372E+01	-0.2537E+00	0.3899E+00	-0.9307E-01	0.7383E-01		

CORRELATION MATRIX

	1	2	3
1	1.0000		
2	-0.3827	1.0000	
3	0.2004	[illegible].9524	1.0000

NORMALIZING ELEMENTS

1	2	3
0.8403E-02	0.2619E-01	0.4156E-01

VARIANCE OF RESIDUALS = 0.2031E+01, 105 DEGREES OF FREEDOM

INDIVIDUAL CONFIDENCE LIMITS FOR EACH PARAMETER (ON LINEAR HYPOTHESIS)

1	2	3
-0.4940E+00	0.3954E+00	-0.5987E+00
-0.5419E+00	0.2461E+00	-0.8356E+00

FINAL PARAMETER VALUES:

1	2	3
-0.5179E+00	0.3207E+00	-0.7172E+00

AUTO AND PARTIAL CORRELATIONS OF THE RESIDUALS

I	AUTO	PARTIAL
1	0.170	0.170
2	0.000	-0.030
3	0.149	0.159
4	0.049	-0.006
5	-0.009	-0.009
6	-0.051	-0.073
7	0.065	0.084
8	0.071	0.045
9	-0.018	-0.015
10	0.000	-0.011
11	-0.077	-0.105
12	-0.082	-0.053
13	-0.009	0.042
14	-0.128	-0.119
15	-0.188	-0.144
16	-0.110	-0.082
17	-0.024	0.023
18	-0.092	-0.048
19	-0.093	-0.021
20	-0.094	-0.113
21	-0.080	-0.059
22	-0.225	-0.202
23	-0.164	-0.078
24	-0.071	-0.059
25	0.115	0.182
26	0.090	0.043
27	0.021	0.009
28	0.081	0.021
29	0.057	-0.002
30	-0.018	-0.059

APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = 0.182

CHI-SQUARED STATISTIC = 37.23
BASED ON (30 - NO. OF STOCHASTIC PARAMETERS) DEGREES OF FREEDOM

Table 14 Identification of Residuals Series of Transfer Function Model

Figure 22:
AUTO CORRELATION VERSUS LAG, K RESIDUALS SERIES

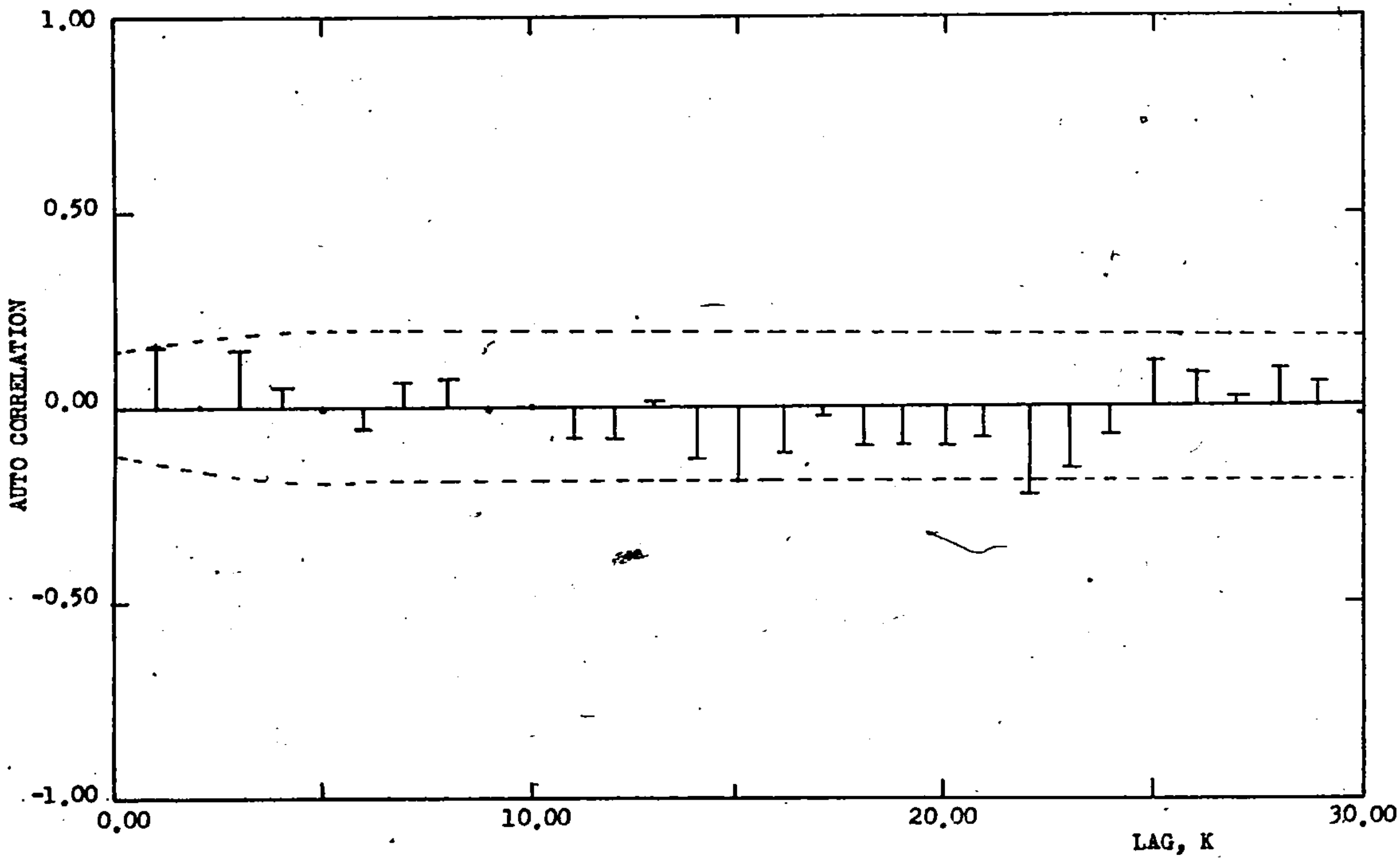
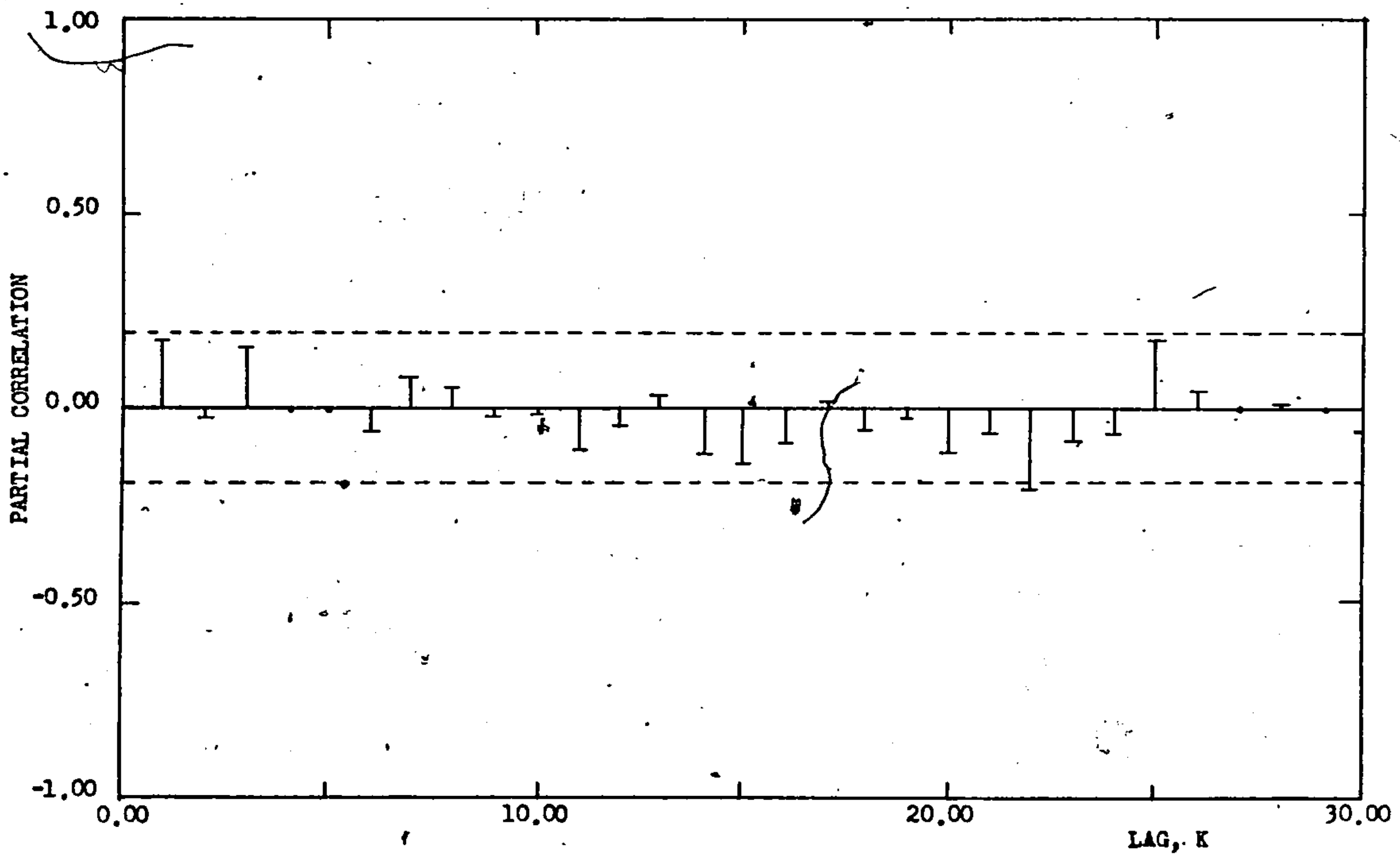


Figure 23:
PARTIAL CORRELATION VERSUS LAG, K RESIDUALS SERIES



CROSS-CORRELATIONS BETWEEN MANIPULATED VARIABLES AND RESIDUALS U(T)*A(T+K)

0	0.219	0	0.219
-1	0.008	1	0.087
-2	0.035	2	0.044
-3	0.082	3	0.058
-4	-0.039	4	0.089
-5	0.004	5	-0.049
-6	-0.046	6	-0.210
-7	-0.067	7	-0.155
-8	0.081	8	-0.018
-9	0.020	9	0.096
-10	0.077	10	0.053
-11	0.173	11	-0.085
-12	-0.073	12	-0.126
-13	0.051	13	-0.038
-14	-0.068	14	-0.071
-15	-0.232	15	-0.051
-16	0.178	16	0.025
-17	0.171	17	0.218
-18	-0.085	18	0.032
-19	-0.170	19	-0.211
-20	-0.117	20	-0.080

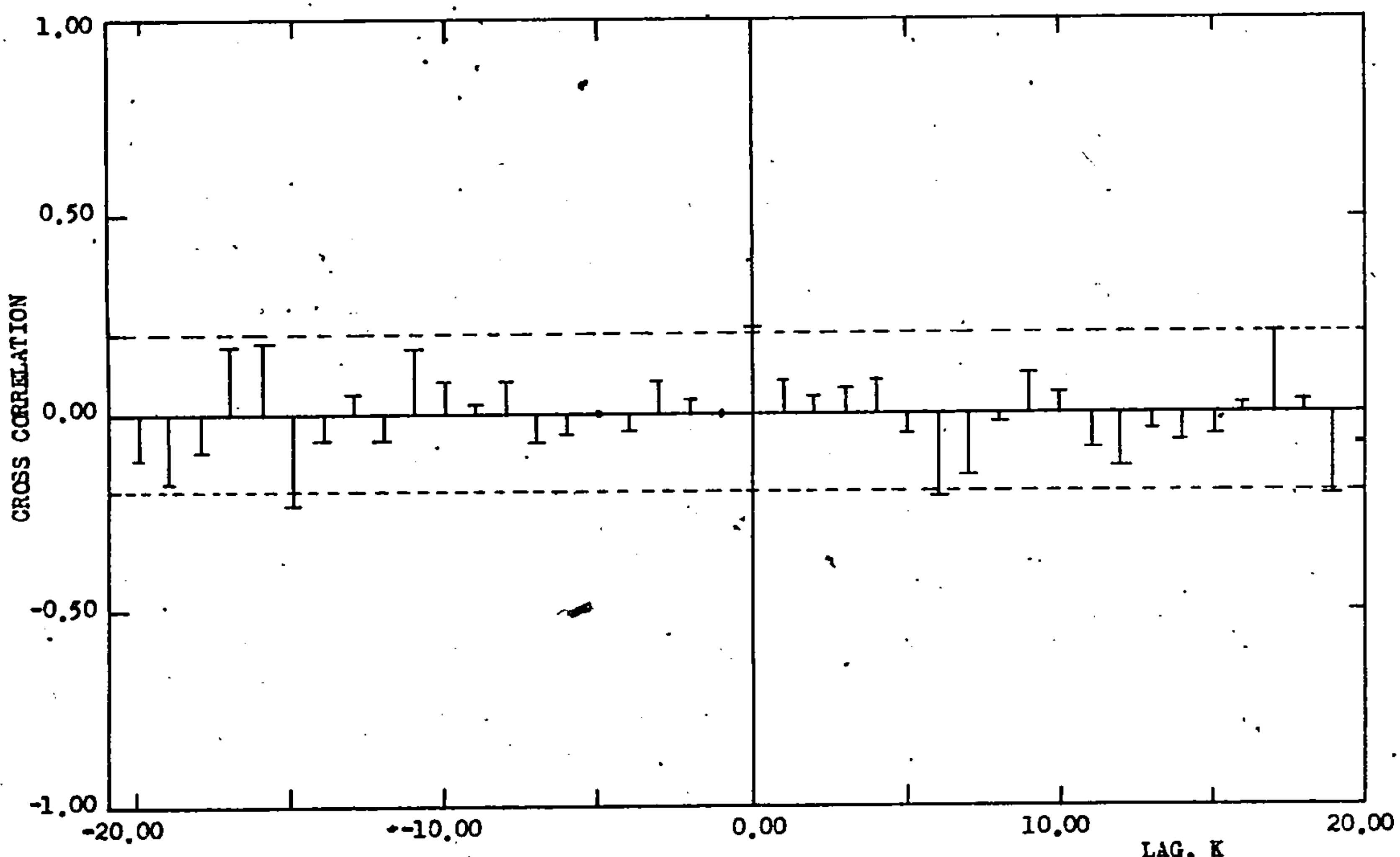
APPROX. 95 PERCENT CONF. LIMIT ON CROSS-CORRELATIONS = 0.192

STANDARD DEVIATIONS S(U) = 0.1202E+02 S(A) = 0.1393E+01

CHI-SQUARED STATISTIC = 27.50
BASED ON (21 - NO. OF DYNAMIC PARAMETERS) DEGREES OF FREEDOM

Table 15 Cross Correlation Function of Control Variable with Residuals of Transfer Function Model

Figure 24:
CROSS CORRELATION VERSUS LAG, K RESIDUALS SERIES



The residuals of the transfer function model were calculated using the best estimates of the parameters. These residuals were used to calculate Auto and Partial Auto Correlation functions, as well as the Cross Correlations with the control variable. The cross correlation function indicates that the transfer function is a reasonable model. The residuals of this model can be seen to behave as white noise in the Auto and Partial Auto Correlation functions. Thus, the model residuals N_t are in fact white noise A_t which makes a stochastic noise model unnecessary.

2.4.4 The Noise Model

In effect, the identification of the noise model parallels the earlier preliminary identification of the U and Y series. However, the model of 2.4.3 forces the noise to be stationary, so that no further differencing of the residuals is necessary. The Auto and Partial Auto Correlations of the transfer function model residuals indicate that those residuals are white noise already.

To demonstrate the use of a noise model in the computer package, and to show some of the symptoms of an unnecessary noise model, a simple noise model will be presented.

The form of the transfer function and noise model was:

$$Y_t = \frac{(\omega_0+\omega_1 B)}{(1+\delta_1 B)} U_{t-2} + (1+\theta_1 B)\frac{A_t}{\nabla} \qquad (3)$$

The model parameter order is (θ_1, ω_0, ω_1, δ_1) for the numerical routine. The tabular data from the computer analysis is given on the next few pages. There are several significant points to note.

1. The sum of squares of the residuals has decreased significantly over the transfer function model.
2. The θ_1 parameter has the largest confidence limits of the parameters. The confidence limit includes zero.
3. Both the Auto and Partial Auto correlation functions of the residuals are now significantly non-zero at the second lag.
4. The cross correlation function of the control variable with the residuals is unchanged.
5. The Chi-Squared statistic indicates a marginal, if any, improvement in the overall fit of the model.

2.4.5 Numerical Package Analysis of the Noise Model

The Test Model Form is given in Equation (3).

NON-LINEAR ESTIMATION, PROBLEM NUMBER 0

108 OBSERVATIONS, 4 PARAMETERS 576 SCRATCH REQUIRED

INITIAL PARAMETER VALUES

1	2	3	4
-0.1200E+00	-0.5400E+00	0.4400E+00	-0.9400E+00
θ_1	ω_0	ω_1	δ_1

INITIAL SUM OF SQUARES = 0.1526E+03

ITERATION NO. 1

DETERMINANT = 0.9341E-01 ANGLE IN SCALED COORD =67.39DEGREES

TEST POINT PARAMETER VALUES

1	2	3	4
-0.1201E+00	-0.5368E+00	0.4364E+00	-0.9355E+00

TEST POINT SUM OF SQUARES = 0.1525E+03

PARAMETER VALUES VIA REGRESSION

1	2	3	4
-0.1201E+00	-0.5369E+00	0.4364E+00	-0.9355E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.1524894E+03

ITERATION NO. 2

DETERMINANT = 0.8629E-01 ANGLE IN SCALED COORD =53.30DEGREES

TEST POINT PARAMETER VALUES

1	2	3	4
-0.1190E+00	-0.5368E+00	0.4370E+00	-0.9365E+00

TEST POINT SUM OF SQUARES = 0.1525E+03

PARAMETER VALUES VIA REGRESSION

1	2	3	4
-0.1190E+00	-0.5368E+00	0.4370E+00	-0.9365E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.1524693E+03

ITERATION NO. 3

DETERMINANT = 0.8752E-01 ANGLE IN SCALED COORD =53.43DEGREES

TEST POINT PARAMETER VALUES

1	2	3	4
-0.1185E+00	-0.5368E+00	0.4368E+00	-0.9361E+00

TEST POINT SUM OF SQUARES = 0.1525E+03

DETERMINANT = 0.1010E+00 ANGLE IN SCALED COORD =52.37DEGREES

PARAMETER VALUES VIA REGRESSION

1	2	3	4
-0.1185E+00	-0.5369E+00	0.4368E+00	-0.9361E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.1524754E+03

ITERATION NO. 4

DETERMINANT = 0.8713E-01 ANGLE IN SCALED COORD =52.91DEGREES

TEST POINT PARAMETER VALUES

1	2	3	4
-0.1184E+00	-0.5368E+00	0.4369E+00	-0.9363E+00

TEST POINT SUM OF SQUARES = 0.1525E+03

PARAMETER VALUES VIA REGRESSION

1	2	3	4
-0.1184E+00	-0.5368E+00	0.4369E+00	-0.9363E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.1524730E+03

ITERATION NO. 5

DETERMINANT = 0.8729E-01 ANGLE IN SCALED COORD =55.41DEGREES

TEST POINT PARAMETER VALUES

1	2	3	4
-0.1183E+00	-0.5368E+00	0.4369E+00	-0.9362E+00

TEST POINT SUM OF SQUARES = 0.1525E+03

DETERMINANT = 0.1007E+00 ANGLE IN SCALED COORD =54.25DEGREES

PARAMETER VALUES VIA REGRESSION

1	2	3	4
-0.1183E+00	-0.5366E+00	0.4359E+00	-0.9362E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.1524739E+03

ITERATION NO. 6

DETERMINANT = 0.8724E-01 ANGLE IN SCALED COORD =50.09DEGREES

TEST POINT PARAMETER VALUES

1	2	3	4
-0.1184E+00	-0.5368E+00	0.4369E+00	-0.9362E+00

TEST POINT SUM OF SQUARES = 0.1525E+03

PARAMETER VALUES VIA REGRESSION

1	2	3	4
-0.1184E+00	-0.5368E+00	0.4369E+00	-0.9362E+00

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.152473SE+03

ITERATION STOPS - RELATIVE CHANGE IN EACH PARAMETER LESS THAN 0.1000E-02

*FINAL RESIDUAL VALUES

0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	-0.4736E+00	-0.1045E+00	0.5822E+00
0.4677E-01	-0.7084E-01	-0.2025E+00	0.7563E+00	-0.4849E+00	0.1282E+01	0.1012E+00	-0.6094E+00
0.3358E+00	-0.1083E+01	0.6068E+00	-0.8983E-01	-0.2803E+00	0.1049E+01	-0.5881E-01	-0.3452E+00
0.5645E+00	0.5419E+00	0.4008E+01	0.2025E+01	-0.2770E+01	-0.4530E+00	0.2211E+01	-0.9141E+00
0.1784E+00	0.7913E+00	-0.1904E+[illegible]	0.3797E+00	0.1443E+01	0.3822E+01	0.1141E+01	-0.2824E+01
-0.1782E+01	0.8702E+00	-0.1652E+01	0.3223E+01	-0.1423E+01	-0.1048E+01	0.8688E+00	-0.1045E+01
0.1074E+00	-0.3133E+00	0.5383E+00	0.1156E+01	0.1977E+01	-0.1804E+01	-0.7144E+00	0.2182E+00
0.5003E+01	0.4006E+00	-0.1281E+01	0.1110E+00	0.6425E-02	0.1517E+00	-0.5109E-01	0.2899E+00
0.6859E+00	-0.6110E+00	0.1696E+00	0.4179E+00	-0.3490E+00	-0.3579E-01	-0.2822E+00	-0.3985E+00
-0.2061E+00	-0.5490E+00	-0.1782E+01	0.1809E+00	-0.8270E+00	0.4103E+00	-0.2948E-01	-0.4013E+00
0.9520E-01	0.1868E+00	-0.2817E+00	0.1214E+01	-0.2680E+00	0.4279E+00	0.6845E-01	0.2024E+00

CORRELATION MATRIX

	1	2	3	4
1	1.0000			
2	0.0231	1.0000		
3	-0.0205	-0.6314	1.0000	
4	0.0101	0.3154	-0.8546	1.0000

NORMALIZING ELEMENTS

1	2	3	4
0.8044E-01	0.8470E-02	0.1471E-01	0.1819E+01

VARIANCE OF RESIDUALS = 0.1466E+01, 104 DEGREES OF FREEDOM

INDIVIDUAL CONFIDENCE LIMITS FOR EACH PARAMETER (ON LINEAR HYPOTHESIS)

1	2	3	4
0.7645E-01	-0.5163E+00	0.4725E+00	-0.8922E+00
-0.3132E+00	-0.5573E+00	0.4013E+00	-0.9803E+00

FINAL PARAMETER VALUES:

1	2	3	4
-0.1184E+00	-0.5368E+00	0.4369E+00	-0.9352E+00

*Note that a truncated display of the final 108 residual values is provided in this example.

AUTO AND PARTIAL CORRELATIONS OF THE RESIDUALS

I	AUTO	PARTIAL
1	0.011	0.011
2	-0.224	-0.224
3	-0.024	-0.019
4	-0.067	-0.124
5	-0.069	-0.082
6	-0.100	-0.156
7	0.081	0.041
8	0.085	0.009
9	-0.050	-0.046
10	0.004	0.001
11	-0.083	-0.117
12	-0.022	-0.021
13	0.134	0.106
14	-0.069	-0.092
15	-0.130	-0.125
16	-0.032	-0.090
17	0.064	0.000
18	0.042	0.007
19	0.076	0.104
20	0.026	-0.035
21	-0.001	0.006
22	-0.164	-0.156
23	-0.122	-0.103
24	-0.019	-0.084
25	0.165	0.125
26	0.055	-0.061
27	-0.077	-0.082
28	-0.049	-0.107
29	-0.002	-0.039
30	-0.041	-0.069

APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = 0.192

CHI-SQUARED STATISTIC = 28.65
BASED ON (30 - NO. OF STOCHASTIC PARAMETERS) DEGREES OF FREEDOM

Table 16 Identification of Residuals Series of Transfer Function and Noise Model

CROSS-CORRELATIONS BETWEEN MANIPULATED VARIABLES AND RESIDUALS   U(T)*A(T+K)

0	0.108	0	0.106
-1	-0.094	1	-0.024
-2	-0.039	2	0.013
-3	0.100	3	-0.011
-4	-0.050	4	0.091
-5	-0.043	5	0.012
-6	-0.057	6	-0.161
-7	-0.042	7	-0.101
-8	0.121	8	0.054
-9	0.057	9	0.173
-10	0.056	10	0.142
-11	0.133	11	-0.042
-12	-0.056	12	-0.134
-13	0.069	13	-0.046
-14	-0.073	14	-0.082
-15	-0.215	15	-0.066
-16	0.233	16	0.024
-17	0.160	17	0.239
-18	-0.129	18	0.054
-19	-0.208	19	-0.247
-20	-0.144	20	-0.115

APPROX. 95 PERCENT CONF. LIMIT ON CROSS-CORRELATIONS = 0.192

STANDARD DEVIATIONS S(U) = 0.1202E+02   S(A) = 0.1180E+01

CHI-SQUARED STATISTIC = 24.76
BASED ON (21 - NO. OF DYNAMIC PARAMETERS) DEGREES OF FREEDOM

Table 17 Cross Correlations of Control Variable with the Residuals of the full Transfer Function and Noise Model.

The information in this numerical presentation indicates that the addition of the θ_1 parameter did not significantly improve on the fit of the transfer function model previously obtained for this system. Therefore, the initial diagnosis on the residuals of the transfer function model which indicated that there was no benefit to the addition of a noise model is correct. However, the addition of the extra parameter did not harm the fit of the model. It is the principle of the most parsimonious model which recognizes that more than one model is adequate, but only one suitable model should be obtained with a minimum number of parameters.

2.4.6 Final Model Summary

In terms of the six parameters of the general model under user control, which set the form of the model, the transfer function was shown to be (r, s, b) = (1, 1, 2) and the noise model was shown to be (p, d, q) = (0, 1, 0). The numerical package estimated the parameters for this model, which gives the final result:

$$Y_t = \frac{(-0.52 + 0.32B)}{(1 - 0.72B)} \cdot U_{t-2} + \frac{A_t}{\nabla}$$

Note that only two decimal accuracy has been retained in the presentation of the final model. This model is adequate for the particular realization of the data which has been processed. As more information becomes available, and the number of observations increases, it is probable that the third or fourth decimal accuracy may change, but the confidence limits on the parameters are small enough to justify the use of two decimal accuracy with reasonable certainty.

The user should recognize the fact that plant equipment ages, and various disturbances may enter the plant. These events will cause parameter drift of either a slow or rapid nature depending on the nature of the disturbance. Once a model form is determined, strategies exist for the on-line or periodic update of model parameters to compensate for the dynamic nature of the plant which is being modelled.[6,7] The ultimate use of a model dictates the attention which the user should direct to each of these concerns, however the fallacy of eight decimal accuracy in model parameters from a computer output should be noted regardless of the intended model use.

CHAPTER 3

THE COMPUTER PROGRAM FORMAT

3.1 The Interactive Routines

The Mainline of the Interactive Time Series (MITS) program, the interactive model subroutine (INTMDL), the data input subroutine (DATA), and the display interruption routine for the continue prompt (CNTNU) are the only routines requiring user interaction. All other subroutines receive integer flags to activate the options selected by the user in the interactive routines. For example, the value of the plot flag (IPLT) selected in the mainline activates the plotting package only if it is a positive integer. This structuring of the subroutine package allows the separation of specific functions for general use. This is illustrated by the generalized plotting routine provided in the appendix as an independent file.

3.2 The Interactive Mainline

The computer package has been structured so that each objective of the interactive program mainline corresponds to one subroutine call statement. This is illustrated in figure 25, which shows the relationship between the stages of the general model building scheme of chapter 1 and the corresponding objectives available in the program mainline. In many cases, these subroutines fulfill specific objectives by delegating numerical calculation, plotting or printing requirements to other subroutines.

Model Building Step	Diagnostic or Objective	Subroutine Called
Preliminary Diagnostics indicate Tentative Model Form	Plot (U vs T) (Y vs T)	PLOTXY
	Identification of U Series.	IDENT
	Identification of Y Series.	IDENT
	Impulse/Step Response.	IMPLSE
	Cross Correlation Y_t*U_{t+k}	CROSS
Set the Tentative Model	Input the desired form of the generalized model and initial parameter estimates	INTMDL
Improve Parameter Estimates	Marquardt Compromise Nonlinear Least Squares for Parameter Estimation.	TSHAUS
Evaluate Model	Identification of Residuals Series.	TSHAUS
	Cross Correlation U_t*A_{t+k}	CROSS
The Iterative Process loops back to INTMDL.		

Figure 25 A Comparison of The Model Building Scheme* To The Mainline Program Call Sequence

*Reference Figure 2, page 5.

3.3 Program File Manipulation

The input/output manipulation of data files is performed in the DATA subroutine and the interactive model subroutine INTMDL. The DATA subroutine opens and closes a single data file to obtain the original user experimental data. The INTMDL subroutine will create two files which are binary versions of the original data (TSDATA.DAT), and the current tentative model form with parameter estimates (BINARY.DAT). Special programs are given in the appendix which allow a printout of the contents of these files.

The TSDATA file allows the mainline program to iterate through various tentative model forms for the data set, without reopening the original fortran coded data file. Note that the mainline contains the close command for this file before program execution terminates normally.

The BINARY file allows the INTMDL subroutine to access and update the current model being considered. A new file is created each time an iteration through the model form loop is made by the mainline program. Note that the use of a remote data file for the automatic update of the model parameters carefully avoids the problems associated with complex subscript arguments for the parameter arrays.

CHAPTER 4

THE SUBROUTINES

4.1 Subroutine Classification

Historically, computer subroutines have been classified as Input/Output (I/O) routines or numerical routines. In the time series computer package there are three subdivisions of each of these categories. The following six tables indicate a general description of each subroutine called by the package, where the table classifications are:

1. The User Interactive I/O routines.
2. General I/O routines.
3. The Time Series Plot Package.
4. General Mathematical routines.
5. General Statistical routines.
6. The Time Series Numerical routines.

Note that category 1 was also discussed in chapter 3. The further problem of subroutines which are especially designed to work on the VAX machine has been partially simplified by placing each function of this type in its own subroutine. The user should create a suitable replacement subroutine which is specific for the desired machine where the package is to reside. The random number generator and some of the plotting software routines are in this category because of a dependence on the machine word size or the machine communication characteristics.

Routine Name	General Description
Program MITS	This is the Mainline of the Interactive Time Series computer package.
INTMDL	The Interactive Model subroutine. This routine uses the form of the general time series model.
CNTNU	Ask the user if program should continue. Call Exit on negative reply.
DATA	The user provides the name of the experimental data file, and DATA retreives the original (T, U, Y) data.

Table 18 The User Interactive I/O Routines

Note that file manipulation by these routines is discussed in chapter 3.

Routine Name	General Description
MATPRT	Matrix print routine for vectors, triangular matrices and square matrices.
RDBF	Read Binary File routine. For use with files TSDATA and BINARY.
TAPMOV	Diagnostic routine for movement of a tape a desired number of records for binary or coded files.
WTBF	Write Binary File routine.

Table 19 General I/O routines

Routine Name	General Description
CONLIM	Places confidence limits on Auto and Partial Auto Correlation plots.
CURVE FRAME PLOTXY SCALE VMAX VMIN TITLE	The entire plot package is used with a call to PLOTXY. VMAX and VMIN find the max and min of the data vectors, then SCALE the data. Use PLOT10 set-up software to set terminal and screen parameters. A FRAME and TITLEs are placed on screen, then a line or bar chart CURVE is plotted.

Table 20 The Time Series Plot Package

Note that the plot package requires the plotting software of the PLOT10 package to provide specific subroutines. The subroutines required are: INITT, TWINDO, DWINDO, MOVEA, DRAWA, DASHA, ANMODE, and FINITT. These subroutines are well documented in the PLOT10 user manual.

Routine Name	General Description
RANDN	This routine creates a normally distributed random number and is machine specific.
DIFF	Return a differenced series to the desired degree by operating on a series with (1-B) once for each degree of differencing desired.
INCR	Increment a vector of NV items, so that V(1)=V(2),...,V(9)=V(10),V(10)=VNEW for NV=10. Useful for past control values.
MATIN	Calculate the inverse and determinant of a matrix.
POLY	Multiply a polynomial in B by (1-B).
RLS	Recursive Least Squares routine. Reference 9: Goodwin and Payne (1977)

Table 21 General Mathematical routines

Routine Name	General Description
ACORR	Calculate Auto Correlation function AC(NLAG) for series Z(NOBS).
CRCORR	Calculate cross correlations for positive lags between series X and Y.
CROSS	Called by the mainline to calculate the full cross correlation function for two series, and display the results.
PARTAL	Calculate Partial Auto Correlation function PAUTO(NLAG) from AC(NLAG).
STATS	Calculate the mean XBAR, and the standard deviation SDX, for series X(NOBS).

Table 22 General Statistical routines

Note that the page references in the comment sections of these subroutines refer to Box and Jenkins[1] where the algorithms for the subroutines may be located. In some cases the computer algorithm is the combination of several algorithms so that some manipulation of the equations is necessary. In that case, references to both algorithms combined are provided in the subroutine comment section.

Routine Name	General Description
HAUSTS	This is the Marquardt Compromise Nonlinear Least Squares subroutine. This version is a FORTRAN77 rewrite of the original program from University of Wisconsin. (4) The original documentation of the I/O nomenclature and conventions has been retained.
IDENT	This routine is called by the mainline to perform an identification on a series.
IMPLSE	This routine is called by the mainline to perform the impulse/step test between an input and output series.
MODEL	This routine performs all calculations relating to the generalized model equation as selected by an option flag.
TSHAUS	This routine is called by the mainline to set up the matrices for a call to HAUSTS. The convergence criteria, maximum number of iterations, and special parameters have been selected for reasonable interactive package behavior.

Table 23 The Time Series Numerical routines

4.2 Benchmarking of Numerical Routines

The general mathematical routines and the general statistical routines were benchmarked by a hand calculation in most cases. The matrix inversion, recursive least squares and stats routines were checked for accuracy by running short fortran programs with the subroutines attached, for problems with known solutions.

The time series numerical routines IDENT, and IMPLSE use combinations of the other routines. Therefore, once the other routines are benchmarked, they may be regarded as display routines for the calculated results.

The Marquardt Compromise subroutine HAUSTS, was rewritten from a program from the University of Wisconsin computer library.(4) The documentation of this library routine included several test cases of data. One of these cases was run with the rewritten version to benchmark the new version of the code. The results of the test run agree within reasonable numerical round-off accuracy with those of the library test case at every stage of the output display. Note that all input/output variables and tests are the same as those presented in the original library documentation. The sample benchmark run is included in the next report section.

4.3 TSHAUS Benchmark Case

This case was run with the MASTER1 command file. The problem data was stored in format (3F10.4) as experimental data. The data function and initial parameter estimates were placed in the Model subroutine. The problem selected was example 1 section 6.1.1 of the library documentation.

Observed data was the energy radiated from a carbon filament lamp per cm^2 per second. The controlled variable was the filament temperature. Six data points were collected. By plotting the data on log-log paper, it is seen that the model should be of the form:

$Y_t = a * U_t^b$ where subscript t denotes case no.

Initial parameter estimates are approximately, a = 0.725 and b = 4.0 . The data file and model subroutine are listed below.

```
      SUBROUTINE MODEL(NPROB,TH,R,NOB,NPR,IFLAG)
      DIMENSION R(NOB),TH(NPR)
      COMMON/DATA/T(1000),U(1000),Y(1000)
      IF(IFLAG.LT.0)THEN
        TH(1)=0.725
        TH(2)=4.0
        NPR=2
      ELSE
        DO 100 K=1,NOB
          R(K)=Y(K)-TH(1)*(U(K)**TH(2))
100     CONTINUE
      END IF
      RETURN
      END
```

DATA FILE

T	U	Y
1.	1.309	2.138
2.	1.471	3.421
3.	1.490	3.597
4.	1.565	4.340
5.	1.611	4.882
6.	1.680	5.660

The TSHAUS output from the test case run is listed below. Note that convergence occurs in three iterations.

```
NON-LINEAR ESTIMATION, PROBLEM NUMBER  1

 6 OBSERVATIONS,     2 PARAMETERS                 32 SCRATCH REQUIRED

INITIAL PARAMETER VALUES

      1            2
 0.7250E+00   0.4000E+01

INITIAL SUM OF SQUARES =  0.1472E-01



                                           ITERATION NO.    1
DETERMINANT =  0.2021E-01       ANGLE IN SCALED COORD  =81.30DEGREES

TEST POINT PARAMETER VALUES

      1            2
 0.7633E+00   0.3875E+01

TEST POINT SUM OF SQUARES =  0.4470E-02

PARAMETER VALUES VIA REGRESSION

      1            2
 0.7633E+00   0.3875E+01

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION =  0.4470259E-02
```

ITERATION NO. 2

DETERMINANT = 0.2059E-01 ANGLE IN SCALED COORD =81.64DEGREES

TEST POINT PARAMETER VALUES

1	2
0.7684E+00	0.3862E+01

TEST POINT SUM OF SQUARES = 0.4318E-02

PARAMETER VALUES VIA REGRESSION

1	2
0.7684E+00	0.3862E+01

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.4318143E-02

ITERATION NO. 3

DETERMINANT = 0.2063E-01 ANGLE IN SCALED COORD =76.58DEGREES

TEST POINT PARAMETER VALUES

1	2
0.7689E+00	0.3860E+01

TEST POINT SUM OF SQUARES = 0.4317E-02

PARAMETER VALUES VIA REGRESSION

1	2
0.7689E+00	0.3860E+01

LAMBDA = 0.100E-01

SUM OF SQUARES AFTER REGRESSION = 0.4317263E-02

ITERATION STOPS - RELATIVE CHANGE IN EACH PARAMETER LESS THAN 0.1000E-02

FINAL RESIDUAL VALUES

-0.3612E-01 0.9831E-02 0.1257E-01 0.7336E-02 0.3667E-01 -0.3687E-01

CORRELATION MATRIX

	1	2
1	1.0000	
2	-0.9906	1.0000

NORMALIZING ELEMENTS

1	2
0.5522E+00	0.1549E+01

VARIANCE OF RESIDUALS = 0.1079E-02, 4 DEGREES OF FREEDOM

INDIVIDUAL CONFIDENCE LIMITS FOR EACH PARAMETER (ON LINEAR HYPOTHESIS)

1	2
0.8051E+00	0.3962E+01
0.7326E+00	0.3759E+01

FINAL PARAMETER VALUES:

1	2
0.7689E+00	0.3860E+01

The final parameter values of the library case were identical to these values. The printout of the library case displayed one more figure than this display, however, all displayed values were the same within reasonable limits throughout the entire output.

4.4 Subroutine Variables Description

The detailed subroutine call statements are listed in table 24. The order of the subroutines is the same as the order of appearance of the subroutines in the program listing in the appendix. The interactive routines are followed by the numerical routines in alphabetical order. The plot package is in alphabetical order at the end of the numerical package. In most cases, the meaning of the variables in the call statement are self-evident. In case of difficulty consult the detailed listing.

The subroutines are not linked to one another by common blocks of variables. The only common block in general use is entitled /DATA/ and contains the (T, U, Y) data. Therefore, required variable values are communicated from one routine to another via the subroutine call statement. The structure of the variable routing through the subroutines is evident in table 25. This table provides the list of subroutines called by each subroutine. Those subroutines which do not call any other routines are listed in table 26.

The user should note the reassignment of integer variables to new values at the beginning of each subroutine. This creates a new transfer address for each variable to prevent loader address problems during nested subroutine calls.

Table 24 Detailed Subroutine Call Statements

```
 INTMDL(#Tape, ParVec, #Par, #Obs, IFLAG)
 CNTNU
 RANDN(IntegerSeed, StdDev, ArithMean, RanNumber)
 ACORR('z', AutoCorr, StdDevZ, #Obs, #Lag)
 CONLIM(AutoCorr, #Lag, 95%ConLim, #Obs, IPLOT,IFLAG)
 CRCORR('x', 'y', CrossCorr, StdDevX, StdDevY, #Obs, #Lag)
 CROSS('x', 'a', #Obs, #Lag, IFLAG, IPLOT)
 DATA(#Obs, #Tape)
 DIFF('x', #Obs, #Differences, "x"Differenced, MeanX)
*HAUSTS(#Prob, #Obs, #Par, ParVec, DeltaParConverg, ...,IPLOT)
 IDENT('z', #Obs, #Lag, #DifferencesToTry, IPLOT)
 INCR(NewV, 'v', #V)
 IMPLSE(#Obs, #Lag, #Differences, IPLOT)
 MATIN(Square"A", DimensionA, Vector"B", IFLAG, Determinant)
 MATPRT(IFLAG, DimensionA, Vector"A", Vector"B", Square"C")
 MODEL(#Prob, ParVec, ResidualsVec, #Obs, #Par, IFLAG)
 PARTAL(AutoCorr, PartialAutoCorr, #Lag)
 POLY(CoefficientsOfPoly, #Differences, #Coefficients)
 RLS(NextObs, CovMatrix, ObsVec, ParVec, RLSPar, #Obs)
 STATS('x', #Obs, StdDevX, MeanX)
*TSHAUS(#Prob, #Obs, #Par, ParVec, ..., IPLOT)

 CURVE('x', 'y', #Obs, IOption, ISymbol)
 FRAME(MainHeading, XHeading, YHeading, "x"scaled, "y", #Obs)
 VMAX(Vector'x', #X, MaxX)
 VMIN(Vector'x', #X, MinX)
 PLOTXY('x', 'y', #Obs, Integer#Title, #XTitle, #YTitle,Opt,IPLOT)
 RDBF('x', #X, #Tape)
 SCALE('z', #Z, Heading, #Heading, "z"scaled)
 TAPMOV(#Tape, #Records, File type flag)
 TITLE(Integer#title, #XTitle, #YTitle, Title, XTitle, YTitle)
 WTBF('x', #X, #Tape)
```

* For detailed descriptions of these call statements see reference 4.

ParVec = Parameter Vector in Table 24.

In table 24 the symbol 'x' means a vector of values which are called X in this subroutine. Comment statements in the subroutines provide information on the values for special integer flags (IFLAG, and IPLOT).

<table>
<tr><th rowspan="2">Subroutine Name</th><th colspan="5">Subroutines Called By This Routine</th></tr>
<tr><th colspan="3">General Subroutines</th><th colspan="2">Plotting Routines</th></tr>
<tr><td>ACORR</td><td>STATS</td><td></td><td></td><td></td><td></td></tr>
<tr><td>CONLIM</td><td></td><td></td><td></td><td>DASHA</td><td>MOVEA</td></tr>
<tr><td>CRCORR</td><td>STATS</td><td></td><td></td><td></td><td></td></tr>
<tr><td>CROSS</td><td>CRCORR</td><td>CNTNU</td><td></td><td>PLOTXY
ANMODE</td><td>CONLIM
FINITT</td></tr>
<tr><td>DATA</td><td>CNTNU</td><td></td><td></td><td></td><td></td></tr>
<tr><td>HAUSTS</td><td>ACORR
MATPRT</td><td>CNTNU
MODEL</td><td>MATIN
PARTAL</td><td>PLOTXY
ANMODE</td><td>CONLIM
FINITT</td></tr>
<tr><td>IDENT</td><td>DIFF
CNTNU</td><td>STATS
PARTAL</td><td>ACORR</td><td>PLOTXY
ANMODE</td><td>CONLIM
FINITT</td></tr>
<tr><td>IMPLSE</td><td>CNTNU
RLS</td><td>DIFF</td><td>STATS</td><td>PLOTXY
ANMODE
FINITT</td><td>DASHA
MOVEA</td></tr>
<tr><td>MODEL</td><td>CONTROL
POLY</td><td>INCR
RDBF</td><td>INTMDL
WTBF</td><td></td><td></td></tr>
<tr><td>TSHAUS</td><td>HAUSTS</td><td></td><td></td><td>ANMODE</td><td></td></tr>
<tr><td>CURVE</td><td>MOVEA</td><td>DRAWA</td><td>ANMODE</td><td></td><td></td></tr>
<tr><td>FRAME</td><td>ANMODE
TWINDO</td><td>DRAWA
VMAX</td><td>DWINDO
VMIN</td><td>ENCODE</td><td>INITT</td></tr>
<tr><td>PLOTXY</td><td>ANMODE
TITLE</td><td>CURVE</td><td>DIFF</td><td>FRAME</td><td>SCALE</td></tr>
<tr><td>SCALE</td><td>VMAX</td><td>VMIN</td><td></td><td></td><td></td></tr>
</table>

Table 25 Cross Index of Subroutines

Note that the plotting subroutines are together in the bottom half of the table 25. The subroutines called by the mainline are described in chapter 3.

Package Subroutines			PLOT10 Subroutines		
DIFF	INCR	MATIN	ANMODE	DASHA	DRAWA
MATPRT	PARTAL	POLY	DWINDO	FINITT	INITT
RLS	STATS	VMAX	MOVEA	TWINDO	
VMIN	RDBF	TAPMOV			
TITLE	WTBF				

Table 26 Subroutines Which do NOT Call Other Routines

The user should pay particular attention to the dimension and type of variable which is used in each call sequence for a given subroutine. The use of the wrong type of variable may result in erroneous transmission of results from the subroutine in the event that the error does not cause a fatal fortran error. Therefore, the independent use of the subroutine package is possible for the user with some degree of familiarity with FORTRAN77 and the VAX computer system.

CHAPTER 5

APPLICATIONS AND CONCLUSIONS

5.1 Possible Package Applications

The application of the ideas incorporated into the entire package will allow the interactive design of a stochastic and transfer function model for a single input/ single output system. The final model is generally intended for use in forecasting or control applications. This version of the computer package is accessed by executing the MASTER command file. Note that the facility for educational user experience with known plant models is also provided.

A subset of this computer package is available by executing the MASTER1 command file. This subset package allows the user to fit any form of model to the experimental data. This aspect of the subset package was demonstrated by the general benchmark case for the numerical routines illustrated in chapter 4.

Some minor alterations to the subset package allow the user to perform a multivariable analysis on the data. Box and Jenkins[1] should be consulted during this application.

The modular nature of the fortran subroutines allows the separate use of the subroutines as library or reference routines. Any application of this type should ensure the availability of the PLOT10 software with the plot package and the availability of all package subroutines which are called by the subroutine being used. Furthermore, caution should be exercised if the package is to be used on another machine because of the machine dependent nature of some of the subroutines.

5.2 The Multivariable Analysis Problem

The file MASTER1 should be executed in this application. The multivariable data should be separated into the various permutations of the variables, with one permutation per data file. For example, data with 2 observations and two control variables (T, U_1, U_2, Y_1, Y_2) should be placed into 4 data files: (T, U_1, Y_1) (T, U_1, Y_2) (T, U_2, Y_1) and (T, U_2, Y_2). Then the time series package may be executed for each of these data files, as far as the fitting of a model. The user may then assemble all the data to establish a model which relates all of the variables: Y_1 = Function of (A_t, U_1, U_2) for example.

The only problem to be considered is the input of the variables Y_1, Y_2, U_1, and U_2 so that they are all available for use in the model subroutine at the same time.

Once all of the variables are available, then the user must fit a transfer function to the data for the calculated set of residuals of the model. An adequate transfer function model will allow the creation of a suitable noise model. All of this may be accomplished using the MASTER1 files.

The data is read into the time series package in the user supplied model subroutine when the IFLAG parameter is less than zero. Two common blocks have been added to the subset package to allow the storage of this data, so that it is available for calculation of model residuals when the IFLAG parameter is greater than or equal to zero. The common blocks are:

```
COMMON/DATAU/U1(1000),U2(1000),...,U5(1000)
COMMON/DATAY/Y1(1000),Y2(1000),...,Y5(1000)
```

These two lines should also be placed in the user supplied model subroutine. The data should then be read in as the variables U1 to U5, and Y1 to Y5, as the user requires. Note that the usual COMMON/DATA/T(1000),U(1000),Y(1000) statement should be included also. The data that is read in should be differenced to the desired degree by the user, since the data is not entering the package by the usual method. All other functions of the model subroutine must also be met by the modified version used in this application.

5.3 Conclusions and Future Work

The ideal situation in the control of a chemical process would be the availability of a real-time, interactive, computer control package. This would permit the on-line real-time collection and display of process data. This data could be used to update or establish suitable control model structures and parameter estimates.

The present computer package will process data from a single file. Future work which will lead towards the ideal situation has several facets:

1. Suitable data updating techniques are required to allow a continuous increase in the number of data points available in real-time.
2. The existing plot package should be altered to permit the real-time display of data. The use of vanishing vectors allows the update of a curve within an existing display frame on screen.
3. The real-time solution of numerical problems introduces a delay in the information available from the computer. Suitable algorithms for control must account for these effects.
4. Analog/digital and communication/instrumentation facilities must be present in the VAX to process interface.

CITATION INDEX

1 Box, G.E.P., and Jenkins, G.M., 1976: Time Series Analysis: Forecasting and Control. Holden-Day.

2 Woods, D.R., et.al., 1975: Teaching Problem Solving Skills, Annals of Engineering Education 1, 238.

3 Friedman, F.L., and Koffman, E.B., 1977: Problem Solving and Structured Programming in Fortran. Addison-Wesley.

4-1 Meeter, D.A., 1965: Nonlinear Least Squares (GAUSHAUS) Vol IV Rev B Sec 3.22, University of Wisconsin Computing Center.

4-2 Marquardt, D.L., 1963: An Algorithm for Least Squares Estimation of Non-linear Parameters, J.Soc.Ind.Appl. Math. 2, 431.

5 MacGregor, J., 1981: Private Communication.

6 Harmon Ray, W., 1981: Advanced Process Control. McGraw-Hill

7 Astrom, K.J., 1970: Introduction to Stochastic Control Theory. Academic Press.

8 Franklin, G.F., and Powell, J.D., 1980: Digital Control of Dynamic Systems. Addison-Wesley.

9 Goodwin, G.C., and Payne, R.L., 1977: Dynamic System Identification: Experimental Design and Data Analysis. Academic Press.

APPENDIX A

TIME SERIES PROGRAM LISTING

```
      PROGRAM MITS
      DIMENSION TH(10), R(1000) ,SCRAT(10000)
      CHARACTER*17    ANS ,FNAM
      CHARACTER*1     NUM(9)
      COMMON/DATA/T(1000),U(1000),Y(1000)
      DATA T/1000*0.0/,U/1000*0.0/,Y/1000*0.0/
      DATA NUM/'1','2','3','4','5','6','7','8','9'/
      NPROB=0
      IPLT=0
      WRITE(6,*)'TECTRONICS TERMINAL IN USE? (Y/N):'
      ACCEPT 1000,ANS
      IF(ANS(1:1).EQ.'N') IPLT=-100
      IF(IPLT.GE.0) THEN
        CALL ANMODE
        WRITE(6,*)' ~ERA M  ~ERA W  ~WOR 0 '
      END IF
*
      WRITE(6,*)'        '
      WRITE(6,*)'INTERACTIVE TIME SERIES PACKAGE (VERSION 1.0/81)'
      WRITE(6,*)' DATA FROM EXPERIMENT OR ASSIGNMENT (E/A):'
      ACCEPT 1000,ANS
      IF(ANS(1:1).EQ.'E') THEN
*
************************************************ READ DATA
*
      NOB=0
      CALL DATA(NOB,NPROB)
    WRITE(6,*)'NUMBER OF OBSERVATIONS IN FILE = ',NOB
*                 THIS CALL TO DATA OPENS UNIT = NPROB
*                 WITH FILE NAME PROVIDED BY USER, TO GET DATA
*                 THEN IT CLOSES THAT FILE.
****************************************************GET ASSIGNMENT
      ELSE
       DO WHILE (NPROB.LE.0 .OR. NPROB.GT.10)
       WRITE(6,*)'INPUT PROBLEM NUMBER ( 1 TO 9 ):'
       ACCEPT*,NPROB
       END DO
       WRITE(6,*)'    '
       WRITE(6,*)'INPUT STUDENT NUMBER:'
       ACCEPT*,NSTUD
       IF(MOD(NSTUD,2).EQ.0)NSTUD=NSTUD+1
*************************************************STUDENT 11 ONLY*******
      IF(NSTUD.EQ.11) THEN
      WRITE(6,*)'CREATE DATA FILE? (Y/N):'
      ACCEPT 1000, ANS
      IF(ANS(1:1) .EQ. 'Y') THEN
*                      SET-UP MODEL
        NPR=10
*           THIS CALL TO MODEL OPENS UNIT = NPROB
*           WITH THE FILENAME = PLNT.DAT;NPROB , AS A NEW FILE.
        NOB=10
```

```
        CALL MODEL(NPROB,TH,R,NOB,NPR,-2)
        WRITE(6,*)'DONE:  EXIT FROM EXECUTION'
        CLOSE (UNIT=NPROB)
        CALL EXIT
      END IF
      END IF
*******************************************END OF STUDENT 11 PACKAGE*****
      FNAM='[MOORE]PLNT.DAT; '
      FNAM(17:17)=NUM(NPROB)
     OPEN(UNIT=NPROB,FILE=FNAM,STATUS='OLD',FORM='UNFORMATTED',
    *READONLY)
      REWIND(NPROB)
      VAR=1.0
      IF(NPROB.EQ.3)VAR=0.25
*                           GET RANDOM NORMAL VARIABLE (MEAN=0.0,STDEV=1.0)
      DO 100 I=1,1000
      CALL RANDN(NSTUD,VAR,0.0,TEMP)
      R(I)=TEMP
100   CONTINUE
      ISAMPLE=0
      DO WHILE (ISAMPLE.EQ.0.OR.ISAMPLE.GT.100)
      WRITE(6,*)'INPUT SAMPLE TIME (1 TO 100):'
      ACCEPT*,ISAMPLE
      ISAMPLE=ABS(ISAMPLE)
      END DO
      NOB=10000
      DO WHILE (NOB.EQ.0.OR.NOB.GT.1000)
       WRITE(6,*)'INPUT NUMBER OF OBSERVATIONS DESIRED (50 TO 1000):'
       ACCEPT*,NOB
       NOB=ABS(NOB)
      END DO
      CALL MODEL(NPROB,TH,R,NOB,ISAMPLE,2)
      CLOSE(UNIT=NPROB)
****************************************************** READY FOR DATA
      END IF
      IF(IPLT.GE.0) THEN
      WRITE(6,*)'PLOT U VERSUS TIME? (Y/N):'
      ACCERT 1000,ANS
      IF(ANS(1:1).EQ.'Y') THEN
        WRITE(6,*)'~ERA W ~ERA M ~WOR 30'
        CALL DIFF(U,NOB,0,R,0.0)
        CALL PLOTXY(T,R,NOB,0,1,2,1,3)
      END IF
      WRITE(6,*)'PLOT Y VERSUS TIME? (Y/N):'
      ACCEPT 1000,ANS
      IF(ANS(1:1).EQ.'Y') THEN
        WRITE(6,*)'~ERA W ~ERA M ~WOR 30'
        CALL DIFF(Y,NOB,0,R,0.0)
        CALL PLOTXY(T,R,NOB,0,1,1,1,3)
      END IF
      CALL CNTNU
```

```
         END IF
*
        DO 300 I=1,2
        IF(IPLT.GE.0)WRITE(6,*)'~ERA W  ~WOR 0'
        IF(I.EQ.1) THEN
         WRITE(6,*)'PERFORM IDENTIFICATION ON U SERIES? (Y/N):'
        ELSE
         WRITE(6,*)'PERFORM IDENTIFICATION ON Y SERIES? (Y/N):'
        END IF
         ACCEPT 1000,ANS
         IF(ANS(1:1).EQ.'Y')THEN
          NL=101
          ND=4
          DO WHILE(NL.EQ.0 .OR. NL.GT.100 .OR. ND.GT.2)
           WRITE(6,*)'INPUT NUMBER OF LAGS FOR CORRELATIONS (1 TO 100):'
           ACCEPT*,NL
           NL=ABS(NL)
           WRITE(6,*)'INPUT NUMBER OF DIFFERENCES TO BE ATTEMPTED (0,1,2):'
           ACCEPT*,ND
           ND=ABS(ND)
           IF(IPLT.GE.0) THEN
           WRITE(6,*)'ARE PLOTS DESIRED? (Y/N):'
           ACCEPT 1000,ANS
           IPLT=0
           IF(ANS(1:1).EQ.'Y'.AND.I.EQ.2)IPLT=1
           IF(ANS(1:1).EQ.'Y'.AND.I.EQ.1)IPLT=2
           END IF
          END DO
        IF(I.EQ.1) CALL IDENT(U,NOB,NL,ND,IPLT)
        IF(I.EQ.2) CALL IDENT(Y,NOB,NL,ND,IPLT)
        END IF
300     CONTINUE
*
******                                   PERFORM IMPULSE TEST
*
        IF(IPLT.GE.0)WRITE(6,*)'~ERA W ~WOR 0'
        WRITE(6,*)'PERFORM IMPULSE TEST? (Y/N):'
        ACCEPT 1000,ANS
        IF(ANS(1:1).EQ.'Y') THEN
         NL=101
         ND=10
         DO WHILE(NL.GT.100.OR.ND.GE.2)
          WRITE(6,*)'INPUT NO.OF IMPULSE WEIGHTS TO BE CALC. (MAX 20):'
          ACCEPT*,NL
          NL=ABS(NL)
          IF(NL.EQ.0)NL=101
          WRITE(6,*)'INPUT NO. OF DIFFERENCES TO BE ATTEMPTED (0,1):'
          ACCEPT*,ND
          ND=ABS(ND)
         END DO
         IF(IPLT.GE.0) THEN
```

```
      IPLT=0
      WRITE(6,*)'ARE PLOTS DESIRED? (Y/N):'
      ACCEPT 1000,ANS
      IF(ANS(1:1).EQ.'Y') IPLT=1
     END IF
     CALL IMPLSE(NOB,NL,ND,IPLT)
    END IF
     IF(IPLT.GE.0)WRITE(6,*)'~ERA W  ~WOR 0'
      WRITE(6,*)'CROSS CORRELATIONS U VERSUS Y? (Y/N):'
      ACCEPT 1000,ANS
      IF(ANS(1:1).EQ.'Y')THEN
        NL=0
        DO WHILE(NL.EQ.0 .OR. NL.GT.100)
        WRITE(6,*)'INPUT NUMBER OF LAGS FOR CROSS CORRELATION (MAX 100):'
        ACCEPT*,NL
        NL=ABS(NL)
        END DO
        IF(IPLT.GE.0) THEN
        WRITE(6,*)'ARE PLOTS DESIRED? (Y/N):'
        IPLT=0
        ACCEPT 1000,ANS
        IF(ANS(1:1).EQ.'Y')IPLT=1
        END IF
        CALL CROSS(U,Y,NOB,NL,0,IPLT)
      END IF
*
***************                              SET-UP DESIRED MODEL FORM
*
      IFLAG=1
      DO WHILE (IFLAG.EQ.1)
      NPR=10
      IF(IPLT.GE.0)WRITE(6,*)'~ERA M ~ERA W ~WOR 0'
      CALL MODEL(NPROB,TH,R,NOB,NPR,-1)
*                        THIS CALL TO MODEL OPENS UNIT = NPROB+1
*                        AS FILE NAME = TSDATA.DAT AS A NEW FILE
*                        CONTAINING STORED BINARY OF ORIGINAL
*                        DATA (T,U,Y).
*
*                        THIS CALL ALSO OPENS UNIT = NPROB
*                        WITH FILE NAME = BINARY.DAT AS A NEW FILE
*                        WHICH CONTAINS MODEL PARAMETERS.
*
*                             ESTIMATE PARAMETERS
      WRITE(6,*)'ESTIMATE PARAMETER VALUES WITH TSHAUS? (Y/N):'
      ACCEPT 1000,ANS
      IF(ANS(1:1).EQ.'Y')THEN
        IF(IPLT.GE.0) THEN
        WRITE(6,*)'PLOT CORRELATIONS FOR RESIDUALS? (Y/N):'
        ACCEPT 1000,ANS
        IPLT=0
        IF(ANS(1:1).EQ.'Y')IPLT=3
```

```
      END IF
      L=5*NPR + NPR*NPR + NOB + NPR*NOB
     CALL TSHAUS(NPROB,NOB,NPR,TH,1.E-6,1.E-3,20,0.01,10.,SCRAT,L,IPLT)
     END IF
*                              GET RESIDUALS
      CALL MODEL(NPROB,TH,R,NOB,NPR,1)
      IF(IPLT.GE.0) THEN
      WRITE(6,*)'PLOT RESIDUALS VERSUS TIME? (Y/N):'
      ACCEPT 1000,ANS
      IF(ANS(1:1).EQ.'Y') THEN
       CALL ANMODE
      WRITE(6,*)'~ERA W ~ERA M ~WOR 30'
       CALL PLOTXY(T,R,NOB,3,1,8,1,3)
       CALL CNTNU
      END IF
      END IF
     IF(IPLT.GE.0)WRITE(6,*)'~ERA W  ~WOR 0'
      WRITE(6,*)'CROSS CORRELATIONS U VERSUS RESIDUALS? (Y/N):'
      ACCEPT 1000,ANS
      IF(ANS(1:1).EQ.'Y')THEN
        NL=0
        DO WHILE(NL.EQ.0 .OR. NL.GT.100)
        WRITE(6,*)'INPUT NUMBER OF LAGS FOR CROSS CORRELATION (MAX 100):'
        ACCEPT*,NL
        NL=ABS(NL)
        END DO
        IF(IPLT.GE.0) THEN
        WRITE(6,*)'ARE PLOTS DESIRED? (Y/N):'
        IPLT=0
        ACCEPT 1000,ANS
        IF(ANS(1:1).EQ.'Y')IPLT=1
        END IF
        CALL CROSS(U,R,NOB,NL,1,IPLT)
     END IF
     CALL CNTNU
*
*
**********************  RETURN DATA TO ORIGINAL VALUES
*
     OPEN(UNIT=NSAVE,FILE='TSDATA.DAT',STATUS='OLD',FORM='UNFORMATTED')
      REWIND(NSAVE)
      READ(NSAVE)NOB
      CALL RDBF(T,NOB,NSAVE)
      CALL RDBF(U,NOB,NSAVE)
      CALL RDBF(Y,NOB,NSAVE)
      CLOSE(UNIT=NSAVE)
      CLOSE(UNIT=NT)
*
       WRITE(6,*)'TRY NEW MODEL? (Y/N):'
       ACCEPT 1000,ANS
       IF(ANS(1:1).EQ.'N')IFLAG=0
```

```
      END DO
      STOP 'DONE'
1000    FORMAT(A10)
1001    FORMAT(3F10.3)
      END
      SUBROUTINE INTMDL(NTP,PVEC,NPAR,NOBS,IFLAG)
      DIMENSION          PVEC(10),IPAR(6)
      CHARACTER*10       FNAM,ANS
      CHARACTER*5        NAME(4)
      CHARACTER*1        NUM(9)
      COMMON/DATA/T(1000),U(1000),Y(1000)
      DATA NAME/'THETA','SIGMA','OMEGA','DELTA'/
      DATA NUM/'1','2','3','4','5','6','7','8','9'/
        DO 10 I=1,10
10      PVEC(I)=0.0
        DO 20 I=1,6
20      IPAR(I)=100
        NT=NTP
        NPAR=1
        NOB=NOBS
        IFLG=IFLAG
      IF(ABS(IFLG).EQ.2) THEN
        FNAM='PLNT.DAT; '
        FNAM(10:10)=NUM(NT)
      ELSE
        FNAM='BINARY.DAT'
      END IF
      OPEN(UNIT=NT,FILE=FNAM,STATUS='NEW',FORM='UNFORMATTED',ERR=50)
50    REWIND(NT)
      WRITE(6,*)'MODEL FORM:'
      WRITE(6,*)'Y(T)=(OMEGA(B)/DELTA(B))*U(T-NB)',
     *'+(THETA(B)/SIGMA(B))*A(T)'
      WRITE(6,*)'FUNCTIONS IN B ARE = OMEGA(0) + OMEGA(1)*B ',
     *'+ OMEGA(2)*B**2 + ...'
*
******                        SAVE T,U,Y IN PRESENT DATA FORM SO THAT WE
******                        CAN RETURN TO ORIGINAL DATA AT ANY TIME
*
        IF(ABS(IFLG).LT.2) THEN
         NSAVE=NT+1
      OPEN(UNIT=NSAVE,FILE='TSDATA.DAT',STATUS='NEW',FORM='UNFORMATTED')
         REWIND(NSAVE)
         WRITE(NSAVE)NOB
        CALL WTBF(T,NOB,NSAVE)
        CALL WTBF(U,NOB,NSAVE)
        CALL WTBF(Y,NOB,NSAVE)
        CLOSE(UNIT=NSAVE)
        END IF
*
        ND=100
        DO WHILE(ND.GT.3)
```

```
        WRITE(6,*)'INPUT DEGREE OF DIFFERENCING FOR U AND Y (0,1,2):'
        ACCEPT*,ND
        ND=ABS(ND)
        END DO
        IPAR(5)=ND
      IF(ABS(IFLG).LT.2) THEN
        NU=NOB
        CALL STATS(U,NU,SDU,UBAR)
        CALL DIFF(U,NU,ND,U,UBAR)
        CALL STATS(Y,NOB,SDY,YBAR)
        CALL DIFF(Y,NOB,ND,Y,YBAR)
        NOBS=NOB
      END IF
*
        NB=100
        DO WHILE(NB.GT.10)
        WRITE(6,*)'INPUT LAG OF CONTROL ACTION (NB) (0 TO 10):'
        ACCEPT*,NB
        NB=ABS(NB)
        END DO
        IPAR(6)=NB
        WRITE(6,*)'     NOTE:  MAX OF 10 PARAMETERS ALLOWED IN MODEL.'
        DO 100 I=1,4
          DO WHILE(IPAR(I).GT.10)
          WRITE(6,1000)NAME(I)
1000    FORMAT(1H ,'INPUT DEGREE OF POLYNOMIAL ',A5,'(B) (0,1,2,3):')
          ACCEPT*,IPAR(I)
          IPAR(I)=ABS(IPAR(I))
          IF(I.EQ.3)IPAR(I)=IPAR(I)+1
          END DO
100     CONTINUE
        IF((IPAR(1)+IPAR(2)+IPAR(3)+IPAR(4)).EQ.1)THEN
          WRITE(6,*)'ALL POLYNOMIALS ARE DEGREE ZERO'
          WRITE(6,*)'EXIT FROM EXECUTION'
          CALL EXIT
        END IF
        IF(IPAR(3).EQ.1) THEN
        WRITE(6,*)'ARE ALL CONTROLLER VALUES ZERO: (Y/N)'
        ACCEPT 1005,ANS
1005    FORMAT(A10)
        IF(ANS(1:1).EQ.'Y')IPAR(3)=0
        END IF
        CALL WTBF(IPAR,6,NT)
*
        WRITE(6,*)'   '
        WRITE(6,*)'   '
        WRITE(6,*)'INPUT PRELIMINARY NUMERICAL ESTIMATES OF PARAMETERS.'
        DO 200 I=1,4
        IF(NPAR.GT.10.OR.(NPAR+IPAR(I)).GT.10) THEN
          WRITE(6,*)'TOO MANY PARAMETERS, CALL EXIT'
          CALL EXIT
```

```
      END IF
        IF(I.EQ.3.AND.IPAR(3).NE.0)THEN
          WRITE(6,*)'INPUT OMEGA(0):'
          ACCEPT*,PVEC(NPAR)
          NPAR=NPAR+1
        ELSE
          WRITE(6,1001)NAME(I)
1001      FORMAT(1H ,'NOTE: ',A5,'(0) = 1.0'/)
        END IF
        NLOOP=IPAR(I)
        IF(I.EQ.3)NLOOP=NLOOP-1
        IF(NLOOP.GT.0) THEN
          DO 150 J=1,NLOOP
            WRITE(6,1002)NAME(I),J
1002      FORMAT(1H ,'INPUT ',A5,'(',I2,') :')
            ACCEPT*,PVEC(NPAR)
            NPAR=NPAR+1
150       CONTINUE
        END IF
        IF(IPAR(I).GT.0) THEN
        IST=NPAR-IPAR(I)
        CALL WTBF(PVEC(IST),IPAR(I),NT)
        END IF
200     CONTINUE
        NPAR=NPAR-1
        RETURN
      END
      SUBROUTINE CNTNU
      CHARACTER*10 ANS
      WRITE(6,*)'CONTINUE? (Y/N):'
      ACCEPT 1000,ANS
1000  FORMAT(A10)
      IF(ANS(1:1).EQ.'N')CALL EXIT
      RETURN
      END
      SUBROUTINE RANDN(IX,S,AM,V)
      A=0.0
      DO 50 I=1,12
      Y=RAN(IX)
50    A=A+Y
      V=(A-6.0)*S+AM
      RETURN
      END
```

```
      SUBROUTINE ACORR(Z,AC,SDZ,NOBS,NLAG)
      DIMENSION Z(NOBS),AC(101)
*
*       ROUTINE TO CALCULATE NLAG AUTO CORRELATIONS (AC) FOR VECTOR Z
*       NOTE  SDZ = STANDARD DEVIATION OF Z
*             MAX VALUE OF NLAG IS 100
*                                             REFERENCE PAGE 32 BOX + JENKINS
*
      NL=NLAG
      NOB=NOBS
      CALL STATS(Z,NOB,SDEVZ,ZBAR)
      I=0
      DO WHILE (I.LE.NL)
        J=1
        SZZ=0.0
            DO WHILE (J.LE.(NOB-I))
            SZZ = SZZ+(Z(J)-ZBAR)*(Z(J+I)-ZBAR)
            J=J+1
            END DO
        I=I+1
*       NOTE THAT AC(1) IS FOR ZERO LAG
        AC(I)=SZZ/(FLOAT(NOB))
      END DO
      SDZ = SDEVZ
*       NOTE VARIANCE OF Z = AC FOR ZERO LAG(STORED IN AC(1))
      VZ = AC(1)
      DO 20 J=1,NL
   20 AC(J) = AC(J+1)/VZ
*       NOTE NOW AC(1) IS AC FOR LAG ONE
      RETURN
      END


      SUBROUTINE CONLIM(ARRAC,NLAG,CL,NOBS,IPLOT,IFLAG)
      DIMENSION ARRAC(101),ARRCL(101),AXIS(101)
*
*       SUBROUTINE TO CALCULATE AND THEN PLOT (IF IPLT=1)
*       THE CONFIDENCE INTERVAL FOR VECTOR AC
*
      NL=NLAG
      NOB=NOBS
      IPLT=IPLOT
      IFLG=IFLAG
*
*       MAKE CL = 2 SIGMA OR 95 % CONFIDENCE INTERVAL
*
      TEMP=1./(SQRT(FLOAT(NOB)))
      CL=2.*TEMP
      IF(IPLT.LE.0)RETURN
*
```

```
*       PLOT AND CALCULATE CONFIDENCE LIMITS(IPLT=1)
*
      TONE=1.0
      IF(IFLG.LT.0)TONE=-1.0
*       POSITIVE AXIS IF IFLG.GE.0 ; NEGATIVE AXIS IF IFLG.LT.0
      DO 10 I=1,NL
   10 AXIS(I)=(FLOAT(I))*TONE
*
*       PLOT STRAIGHT LINE C.I. FOR PARTIAL CORR(IFLG.EQ.0)
*
      IF(IFLG.LT.0)IFLG=-IFLG
      IF(IFLG.NE.1)THEN
        CALL MOVEA(0.0,CL)
        CALL DASHA(AXIS(NL),CL,3)
        CALL MOVEA(0.0,-CL)
        CALL DASHA(AXIS(NL),-CL,3)
      ELSE
*
*       PLOT CURVED ACTUAL C.I. FOR AUTO CORR(ABS(IFLG).EQ.1)
*                                 REFERENCE PAGE 177, BOX + JENKINS
        SUM=1.0
        DO 30 I=1,NL
           ARRCL(I)=CL*(SQRT(SUM))
           SUM=SUM+2.*ARRAC(I)*ARRAC(I)
   30   CONTINUE
        CALL MOVEA(0.0,ARRCL(1))
        DO 35 I=1,NL
   35   CALL DASHA(AXIS(I),ARRCL(I),3)
        DO 40 I=1,NL
           ARRCL(I)=-ARRCL(I)
   40   CONTINUE
        CALL MOVEA(0.0,ARRCL(1))
        DO 45 I=1,NL
   45   CALL DASHA(AXIS(I),ARRCL(I),3)
      END IF
      RETURN
      END


      SUBROUTINE CRCORR(X,Y,CC,SDX,SDY,NOBS,NLAG)
      DIMENSION X(NOBS),Y(NOBS),CC(101)
*
*       SUBROUTINE TO CALCULATE CROSS CORRELATION OF X WITH Y
*       ASSUME BOTH SERIES DIFFERENCED FOR STATIONARITY
*                     REFERENCE PAGE 374 BOX + JENKINS
*
      NL=NLAG
      NOB=NOBS
*                                 CALCULATE STATS FOR X AND Y
      CALL STATS(X,NOB,SDEVX,XBAR)
```

```
      CALL STATS(Y,NOB,SDEVY,YBAR)
      SDX=SDEVX
      SDY=SDEVY
*                                 CALCULATE CROSS PRODUCTS
        I=0
        DO WHILE(I.LE.NL)
        SXY=0.0
        J=1
                DO WHILE(J.LE.(NOB-I))
                SXY=SXY+(X(J)-XBAR)*(Y(J+I)-YBAR)
                J=J+1
                END DO
        I=I+1
*               NOTE THAT CC(1) IS FOR LAG ZERO
        CC(I)=(SXY/(FLOAT(NOB)))/(SDX*SDY)
        END DO
      RETURN
      END


      SUBROUTINE CROSS(X,A,NOBS,NLAG,IFLAG,IPLOT)
      DIMENSION X(NOBS),A(NOBS)
      DIMENSION CC1(101),CC2(101),YN(205),AXIS(205)
      DATA YN/205*0.0/
      DATA AXIS/205*0.0/
*
*               SUBROUTINE TO COMBINE THE RESULTS OF X CROSS
*               CORRELATED WITH A , WITH THE RESULTS OF A CROSS
*               CORRELATED WITH X.  RESULTS ARE THEN PLOTTED WITH
*               CONFIDENCE LIMITS IF DESIRED(IPLT=1).
*
      NOB=NOBS
      NL=NLAG
      IPLT=IPLOT
      IF(IPLT.GT.0)THEN
      CALL ANMODE
      WRITE(6,*)' ~ERA W   ~WOR 30 '
      END IF
      CALL CRCORR(X,A,CC2,SDX,SDA,NOB,NL)
      CALL CRCORR(A,X,CC1,SDA,SDX,NOB,NL)
      CL = 2.0/SQRT(FLOAT(NOB))
*
*               SETUP VALUES FOR CROSS CORRELATION PLOTS, COMBINING CC1/CC2
*
       IF(IPLT.GT.0)THEN
         DO 10 I=1,NL
          IP1NL=NL+1+I
          IM1NL=NL+1-I
          AXIS(IP1NL)=FLOAT(I)
          AXIS(IM1NL)=-FLOAT(I)
```

```
          YN(IP1NL)=CC2(I+1)
          YN(IM1NL)=CC1(I+1)
10       CONTINUE
        NLP1=NL+1
        NTOT=2*NL+1
        YN(NLP1)=CC2(1)
        CALL PLOTXY(AXIS,YN,NTOT,0,2,5,11,IPLT)
        CALL CONLIM(CC1,NL,CL,NOB,IPLT,-1)
        CALL CONLIM(CC2,NL,CL,NOB,IPLT,1)
        CALL FINITT(10,10)
      END IF
      IF(IFLAG.EQ.1) THEN
      PRINT 100
  100 FORMAT ('1   CROSS-CORRELATIONS BETWEEN MANIPULATED VARIABLES AND
     1RESIDUALS    U(T)*A(T+K)'//)
      ELSE
      PRINT 110
  110 FORMAT('1   CROSS-CORRELATIONS BETWEEN OBSERVATIONS AND MANIPULAT
     1ED VARIABLES    U(T)*Y(T+K)'//)
      END IF
*
*       PRINT TABLE OF CROSS CORRELATIONS
*
      K=1
        DO WHILE (K.LE.(NL+1))
         KK = K-1
         K1 = -KK
*               NOTE THAT   CC1(1)=CC2(1)  BOTH ARE FOR LAG ZERO
         PRINT 101 , K1,CC1(K),KK,CC2(K)
        K=K+1
        END DO
  101 FORMAT (5X,I3,5X,F6.3,10X,I3,5X,F6.3)
      PRINT 102 , CL
  102 FORMAT(//' APPROX. 95 PERCENT CONF. LIMIT ON CROSS-CORRELATIONS =
     1',F6.3)
*
      IF(IFLAG.EQ.1) THEN
      PRINT 103, SDX,SDA
  103 FORMAT ( /' STANDARD DEVIATIONS  S(U) =',E12.4,5X,'S(A) =',E12.4 )
      ELSE
      PRINT 104, SDX,SDA
  104 FORMAT ( /' STANDARD DEVIATIONS S(Y) =',E12.4,5X,'S(U) =',E12.4 )
      END IF
      Q = 0.0
      NL1=NL-1
      DO 20 J=1,NL1
20    Q = Q+ CC2(J)*CC2(J)/FLOAT(NOB-J)
      Q=Q*FLOAT(NOB*NOB)
      NDF=NL+1
      PRINT 105, Q,NDF
  105 FORMAT (/' CHI-SQUARED STATISTIC = ',F6.2/' BASED ON (',I3,' - NO.
```

```
     1 OF DYNAMIC PARAMETERS) DEGREES OF FREEDOM'/)
        CALL CNTNU
       RETURN
       END


      SUBROUTINE DATA(NOBS,NTP)
      COMMON /DATA/T(1000),U(1000),Y(1000)
      CHARACTER           NAME*10
      INTEGER             TAPE_STATUS*4
      PARAMETER (END_OF_FILE =-1)
      DATA T,U,Y/1000*0.0,1000*0.0,1000*0.0/
*
*            NOTE 1000 POINTS FOR DATA MAXIMUM ALLOWED
*
      NOB=NOBS
      NT=NTP
      I=1
      IF(NT.LE.0)NT=5
*                       INPUT DATA TYPE AND FILENAME
      ITYPE=0
      DO WHILE(ITYPE.LT.1 .OR. ITYPE.GT.2)
        WRITE(6,*)'INPUT DATA TYPE'
        WRITE(6,*)'      TYPE = 1 DATA FORMAT (T,U) 2F10.3'
        WRITE(6,*)'      TYPE = 2 DATA FORMAT (T,U,Y) 3F10.3'
        ACCEPT*,ITYPE
      END DO
        WRITE(6,*)'INPUT DATA FILENAME (IE. TIME.DAT)'
        ACCEPT 1000,NAME
        OPEN(UNIT=NT,FILE=NAME,STATUS='OLD',READONLY,ERR=99)
      IF(ITYPE.EQ.1)READ(NT,100,IOSTAT=TAPE_STATUS)T(I),U(I)
      IF(ITYPE.EQ.2)READ(NT,200,IOSTAT=TAPE_STATUS)T(I),U(I),Y(I)
      I=2
          DO WHILE (TAPE_STATUS.NE.END_OF_FILE .AND. I.LE.1000)
        IF(ITYPE.EQ.1)READ(NT, 100, IOSTAT=TAPE_STATUS)T(I),U(I)
        IF(ITYPE.EQ.2)READ(NT, 200, IOSTAT=TAPE_STATUS)T(I),U(I),Y(I)
          I=I+1
          END DO
        NOBS=I-2
        CLOSE(UNIT=NT)
      CALL CNTNU
      RETURN
99    WRITE(6,*)'ERROR ON OPEN WITH FILENAME:',NAME
      CALL EXIT
100   FORMAT(2F10.3)
200   FORMAT(3F10.3)
1000  FORMAT(A10)
      RETURN
      END
```

```
      SUBROUTINE DIFF(X,NOBS,NDIFFD,XD,XB)
      DIMENSION X(NOBS),XD(NOBS)
*
*       FOR ZERO DIFFERENCING SUBTRACT THE MEAN FROM ALL X
*       NOTE X VECTOR IS UNCHANGED, XD IS RETURNED.
*       XB IS THE MEAN OF THE VECTOR X, WHICH IS NOBS IN LENGTH.
*       ALSO, NOBS IS RETURNED AS LENGTH OF XD NOT X.
*
      NOB=NOBS
      NDIF=NDIFFD
      XBAR=XB
      IF(NDIF.LE.0)THEN
      DO 10 I=1,NOB
   10 XD(I)=X(I)-XBAR
      ELSE
*
*       REGULAR DIFFERENCING
*
        DO 20 I=1,NOB
   20   XD(I)=X(I)
        ND=1
        DO WHILE (ND.LE.NDIF)
                I=1
                DO WHILE (I.LE.(NOB-ND))
                XD(I)=XD(I+1)-XD(I)
                I=I+1
                END DO
        ND=ND+1
        END DO
        NOBS=NOB-NDIF
       END IF
       RETURN
       END


       SUBROUTINE HAUSTS(NPRBD,NOBS,NQ,TH,EP1S,EP2S,MIT,FLAM,FNU,
      1   Q,P,E,PHI,TB,  R,A,D,DELZ,IPLOT)
*
*                FORTRAN II VERSION
*      ADAPTED FOR THE VAX 11/750  (R. F. MOORE  8/81)
*
       COMMON /DATA/T(1000),U(1000),Y(1000)
       DIMENSION TH(NQ), R(NOBS)
       DIMENSION Q(NQ), P(NQ), E(NQ), PHI(NQ), TB(NQ)
       DIMENSION A(NQ,NQ), D(NQ,NQ), DELZ(NQ,NOBS)
       DIMENSION AC(101),PP(101),AXIS(101),TRANS(1000)
*
*               SUBROUTINE TO DETERMINE BEST FIT OF PARAMETERS
```

```
*                 FOR A GIVEN MODEL FORM.
*
      ACOS(X) = ATAN(SQRT(1.0/X**2 - 1.0))
*             SET THE NUMBER OF LAGS FOR RESIDUAL AC
      NL=30
      DO 50 I=1,NL
50    AXIS(I)=FLOAT(I)
*
*
*                           TO AVOID LOADER TRANSFER ADDRESS PROBLEMS
*                           ALL CONSTANTS TRANSFERRED BY CALL STATEMENT
*                           ARE ASSIGNED NEW NAMES FOR USE BY THIS ROUTINE.
      IPLT=IPLOT
      NP = NQ
      NPROB = NPRBO
      NOB = NOBS
      EPS1 = EP1S
      EPS2 = EP2S
      NPSQ = NP * NP
       NSCRAC = 5*NP+NPSQ +  NOB+NP*NOB
*
*
*                                          PRINT HEADINGS + INITIAL VALUES
      PRINT 1000, NPROB, NOB, NP, NSCRAC
      PRINT 1001
      CALL MATPRT(1, NP, TH, TEMP, TEMP)
*
*
*                                          CHECK PARAMETERS
      IF(MINO(NP-1,50-NP,NOB-NP,MIT-1,999-MIT).LT.0  .OR.
     *   (FNU-1.0).LE.0.0 )THEN
*****   PARAMETER ERROR
        PRINT 1002
        CALL EXIT
        END IF
      I=1
      DO WHILE (I.LE.NP)
        IF(TH(I).EQ.0.0)THEN
*****     PARAMETER ERROR
          PRINT 1002
          CALL EXIT
        END IF
      I=I+1
      END DO
*
      GA = FLAM
      NIT = 1
      IF(EPS1.LT.0.0)EPS1=0.0
      SSQ = 0.0
      CALL MODEL(NPROB, TH, R, NOB, NP,0)
*
```

```
*              CALCULATE AND PRINT INITIAL SUM OF SQUARES OF RESIDUALS
*
      DO 100 I = 1, NOB
100   SSQ=SSQ+R(I)*R(I)
      PRINT 1003, SSQ
*
*************************************************************** BEGIN ITERATION
*
      DO WHILE (NIT.LE.MIT)
      GA = GA / FNU
      INTCNT = 0
      PRINT 1004, NIT
      DO 120 I=1,NP
          TEMP = TH(I)
*                    TEMPORARILY CHANGE EACH TH(J) IN TURN BY 1 PER CENT
          P(I) = 0.01*TH(I)
          TH(I)= TH(I)+P(I)
          Q(I)=0.0
            CALL MODEL(NPROB, TH, TRANS, NOB, NP,0)
              DO 110 J = 1, NOB
              DELZ(I,J) = R(J) - TRANS(J)
110           Q(I) = Q(I) + DELZ(I,J) * R(J)
*         NOTE THAT Q(I) IS THE SUM OF DEVIATIONS OVER OBS. FOR EACH PARAMETER
*         ALSO, NORMALIZE Q(I) BY MAGNITUDE OF CHANGE WHICH CAUSED DEVIATION.
          Q(I)= Q(I)/P(I)
120       TH(I)=TEMP
*                    RETURN TH(J) BACK TO ORIGINAL VALUE
*
*********************************************     Q=XT*R (STEEPEST DESCENT)
*
      DO 150 I = 1, NP
        DO 140 J=1,I
*                    SUM THE CROSS PRODUCTS FOR EACH 1 PER CENT PARAMETER
*                    CHANGE WITH THE OTHER PARAMETER CHANGES
        SUM = 0.0
          DO 130 K = 1, NOB
130       SUM = SUM + DELZ(I,K) * DELZ(J,K)
*                    NORMALIZE SUM BY PRODUCT OF 1%  CHANGES IN PARAMETERS
        SUM= SUM/(P(I)*P(J))
        D(I,J) = SUM
140     D(J,I) = SUM
150   E(I) = SQRT(D(I,I))
*
************************************** APPLY PRESENT LAMDA VALUE ( IN GA )
*
      NEWGA=1
      NEWSTP=1
      DO WHILE (INTCNT.LT.36.AND.NEWGA.EQ.1)
      NEWGA=0
*
      DO 170  I = 1, NP
```

```
      DO 160 J=1,I
      DO 170  I = 1, NP


      DO 160 J=1,I
      A(I,J) = D(I,J) / (E(I)*E(J))
160   A(J,I) = A(I,J)
170   CONTINUE
*                                         A= SCALED MOMENT MATRIX
      DO 180 I=1,NP
      P(I)=Q(I)/E(I)
      PHI(I)=P(I)
180   A(I,I) = A(I,I) + GA
*
       CALL MATIN(A, NP, P, 1, DET)
*                                         P/E = CORRECTION VECTOR
      STEP=1.0
        SUM1=0.0
        SUM2=0.0
        SUM3=0.0
      DO 190 I=1,NP
           SUM1=SUM1 + P(I)*PHI(I)
           SUM2=SUM2 + P(I)*P(I)
           SUM3=SUM3 + PHI(I) * PHI(I)
190   PHI(I) = P(I)
*                                         DETERMINE ANGLE IN SCALED CO-ORDINATES
      ANGLE = SUM1/SQRT(SUM2*SUM3)
      ANGLE = AMIN1(ANGLE, 1.0)
      ANGLE = 57.295*ACOS(ANGLE)
      PRINT 1005, DET, ANGLE
*
*****                                     APPLY PRESENT STEP SIZE AND TEST FOR
*****                                     SUM OF SQUARES TO BE REDUCED TO LAST
*****                                     VALUE OBTAINED.
*
      DO WHILE (INTCNT.LT.36 .AND. NEWSTP.EQ.1)
      NEWSTP=0
*
        DO 200 I = 1, NP
          P(I) = PHI(I) *STEP / E(I)
          TB(I) = TH(I) + P(I)
200     CONTINUE
*                PRINT TEST POINT PARAMETER VALUES
        PRINT 1006
        CALL MATPRT(1,NP,TB,TEMP,TEMP)
        SUMB=0.0
        CALL MODEL(NPROB, TB, R, NOB, NP,0)
        DO 210 I=1,NOB
210     SUMB=SUMB+R(I)*R(I)
*       PRINT AND TEST SUM OF SQUARES OF RESIDUALS FOR EXIT THIS ITERATION
        PRINT 1007, SUMB
        IF(SUMB .LE. (1.0+EPS1)*SSQ) INTCNT=100
*       NOTE THAT EXIT FROM DO LOOP ON INTCNT .GT. 36
        IF(AMIN1(ANGLE-30.0, GA) .LE. 0.0 ) THEN
          STEP=STEP/2.0
```

```
          NEWSTP=1
        ELSE
          GA=GA*FNU
          NEWGA=1
        END IF
*
        INTCNT = INTCNT + 1
      END DO
*
*
      END DO
*
      IF(INTCNT.GE.36.AND.INTCNT.LT.100) THEN
******          END ITERATIONS
        PRINT 1008
        NIT=MIT
      ELSE
******       CHECK STOPPING CRITERIA
*
        PRINT 1009
        DO 220 I=1,NP
220     TH(I)=TB(I)
         CALL MATPRT(1, NP, TH, TEMP, TEMP)
        PRINT 1010, GA, SUMB
*
          IFLAG=0
          IF(EPS2.GT.0.0) THEN
*               TEST EPS2 FOR STOPPING CRITERIA
            I=1
            IFLAG=1
            DO WHILE(IFLAG.EQ.1 .AND. I.LE.NP)
              IF(ABS(P(I))/(1.E-20+ABS(TH(I))) .GT. EPS2) IFLAG=0
*               IF IFLAG = 0 TEST EPS1 FOR STOPPING CRITERIA
            I=I+1
            END DO
*               IF IFLAG = 1 STOP ON EPS2 CRITERIA
            IF(IFLAG.EQ.1) THEN
******  END ITERATIONS
              PRINT 1011,EPS2
              NIT=MIT
            END IF
          END IF
*
        IF(EPS2.LE.0.0 .OR. IFLAG.EQ.0) THEN
*               NOW CHECK TO STOP ON EPS1 CRITERIA
          IF(EPS1.GT.0.0 .AND. ABS(SUMB-SSQ) .LE. EPS1*SSQ) THEN
******  END ITERATIONS
            PRINT 1012, EPS1
            NIT=MIT
          END IF
        END IF
```

```
      END IF
      SSQ=SUMB
      NIT=NIT+1
      CALL CNTNU
*
      END DO
*
*****************************************************   END ITERATION
*
*       PRINT FINAL RESIDUAL VALUES
      PRINT 1013
      PRINT 1014, (R(I), I = 1, NOB)
      CALL CNTNU
      SSQ=SUMB
      IDF=NOB-NP
*       CALCULATE + PRINT CORRELATION MATRIX
      PRINT 1015
       CALL MATIN(D, NP, P, 0, DET)
      DO 230 I=1,NP
230   E(I) = SQRT(D(I,I))
      DO 250 I=1,NP
        DO 240 J = I, NP
        A(I,J) = D(I,J) / (E(I)*E(J))
240     A(J,I) = A(I,J)
250   CONTINUE
       CALL MATPRT(3, NP, TEMP, TEMP, A)
      PRINT 1018
       CALL MATPRT(1, NP, E, TEMP, TEMP)
      IF(IDF.EQ.0) RETURN
*       CALCULATE + PRINT VARIANCE OF RESIDUALS
      VAR = SSQ / IDF
      PRINT 1017, VAR, IDF
       SDEV = SQRT(VAR)
*       CALCULATE + PRINT CONFIDENCE LIMITS ON PARAMETERS
      DO 260 I=1,NP
      P(I)=TH(I)+2.0*E(I)*SDEV
260   TB(I)=TH(I)-2.0*E(I)*SDEV
      PRINT 1018
       CALL MATPRT(2, NP, TB, P, TEMP)
      PRINT 1023
        CALL MATPRT(1,NP,TH,TEMP,TEMP)
      CALL CNTNU
*       CALCULATE + PRINT AUTO + PARTIAL CORRELATIONS FOR RESIDUALS
      CALL ACORR(R,AC,SDF,NOB,NL)
      CALL PARTAL(AC,PP,NL)
       IF(IPLT.GT.0) THEN
        CALL ANMODE
        TYPE*,' ~ERA W  ~WOR 30 '
        CALL PLOTXY(AXIS,AC,NL,3,2,3,11,IPLT)
        CALL CONLIM(AC,NL,CL,NOB,IPLT,1)
        CALL FINITT(10,10)
```

```
        CALL CNTNU
        CALL PLOTXY(AXIS,PP,NL,3,2,4,11,IPLT)
        CALL CONLIM(AC,NL,CL,NOB,IPLT,0)
        CALL FINITT(10,10)
      END IF
      PRINT 1019
      DO 270 I=1,NL
270   PRINT 1020 , I,AC(I),PP(I)
*               PRINT CONFIDENCE LIMITS ON AUTO CORRELATION OF RESIDUALS
      CALL CONLIM(AC,NL,CL,NOB,0,0)
      PRINT 1021,CL
*               CALCULATE AND PRINT CHI-SQUARED STATISTIC ALSO.
      CHI = 0.
      DO 280 I=1,NL
280   CHI = CHI+AC(I)*AC(I)/FLOAT(NOB-I)
      CHI = CHI*FLOAT(NOB*NOB)
      PRINT 1022, CHI
      CALL CNTNU
      RETURN
*
************************************************************ FORMAT BLOCK
*
1000  FORMAT(38H1NON-LINEAR ESTIMATION, PROBLEM NUMBER   I3,// I5,
     *  14H OBSERVATIONS, I5, 11H PARAMETERS  I14, 17H SCRATCH REQUIRED)
1001  FORMAT(/25H0INITIAL PARAMETER VALUES  )
1002  FORMAT(/16H0PARAMETER ERROR    )
1003  FORMAT(/25H0INITIAL SUM OF SQUARES =   E12.4)
1004  FORMAT(/////45X,13HITERATION NO.  I4)
1005  FORMAT(14H DETERMINANT = E12.4, 6X, 25H ANGLE IN SCALED COORD  =
     *  F5.2, 8HDEGREES    )
1006  FORMAT(30H0TEST POINT PARAMETER VALUES        )
1007  FORMAT(28H0TEST POINT SUM OF SQUARES =    E12.4)
1008  FORMAT(//53H0**** THE SUM OF SQUARES CANNOT BE REDUCED TO THE SUM,
     *62H OF SQUARES AT THE END OF THE LAST ITERATION - ITERATING STOPS/
     *)
1009  FORMAT(/32H0PARAMETER VALUES VIA REGRESSION  )
1010  FORMAT(/9H0LAMBDA =E10.3,//,34H SUM OF SQUARES AFTER REGRESSION =
     *E15.7)
1011  FORMAT(/54H0ITERATION STOPS - RELATIVE CHANGE IN EACH PARAMETER L,
     *11HESS THAN   E12.4)
1012  FORMAT(/54H0ITERATION STOPS - RELATIVE CHANGE IN SUM OF SQUARES L,
     *11HESS THAN  E12.4)
1013  FORMAT(22H1FINAL RESIDUAL VALUES  )
1014  FORMAT(/10E12.4)
1015  FORMAT(/19H0CORRELATION MATRIX    )
1016  FORMAT(21H0NORMALIZING ELEMENTS    )
1017  FORMAT(24H0VARIANCE OF RESIDUALS =    ,E12.4,1H,I4,
     *20H DEGREES OF FREEDOM    )
1018  FORMAT(54H0INDIVIDUAL CONFIDENCE LIMITS FOR EACH PARAMETER (ON L,
     *20HINEAR HYPOTHESIS)    )
1019  FORMAT('1   AUTO AND PARTIAL CORRELATIONS OF THE RESIDUALS '//
```

```
     * 9X,'I',14X,'AUTO',14X,'PARTIAL'//)
1020  FORMAT(8X,I2,13X,F6.3,13X,F6.3)
1021  FORMAT(//' APPROX.  95 PERCENT CONF.  LIMIT ON CORRELATIONS = ',
     * F5.3)
1022  FORMAT(/' CHI-SQUARED STATISTIC = ',F6.2/' BASED ON (30 - NO. OF S
     *TOCHASTIC PARAMETERS) DEGREES OF FREEDOM'/)
1023  FORMAT(/24H FINAL PARAMETER VALUES:)
      END



      SUBROUTINE IDENT(Z,NOBS,NLAG,NDIFFD,IPLOT)
      DIMENSION Z(NOBS),AC(101),PP(101)
      DIMENSION ZD(1000),AXIS(101)
*
*                     SUBROUTINE TO IDENTIFY THE CHARACTERISTICS
*                     OF A TIME SERIES Z, BY DIFFERENCING UP TO NDIFFD
*                     TIMES FOR STATIONARITY, AND DISPLAYING UP TO NLAG
*                     AUTO AND PARTIAL CORRELATIONS, TO PERMIT IDENTIFICATION
*                     OF APPROPRIATE MODEL FORM FOR THE SERIES.
*
      IF(IPLOT.GT.0) THEN
        CALL ANMODE
        WRITE(6,*)' ~ERA W  ~WOR 30 '
      END IF
      NOB=NOBS
      NL=NLAG
      NDIFF=NDIFFD
      IPLT=IPLOT
      DO 100 I=1,NL
  100 AXIS(I)=FLOAT(I)
      CALL STATS(Z,NOB,SDZ,ZBAR)
*
*                             BEGIN DIFFERENCE LOOP
*
        ND=0
        DO WHILE (ND.LE.NDIFF)
          NDATA=NOB
          CALL DIFF(Z,NDATA,ND,ZD,ZBAR)
*                                     ZD = DIFFERENCED VECTOR OF Z VALUES
*
      CALL ACORR(ZD,AC,SDZ,NDATA,NL)
      CALL PARTAL(AC,PP,NL)
*
       IF(IPLT.GT.0)THEN
       CALL PLOTXY(AXIS,AC,NL,ND,2,3,11,IPLT)
       CALL CONLIM(AC,NL,CL,NDATA,IPLT,1)
       CALL FINITT(10,10)
       CALL CNTNU
       CALL PLOTXY(AXIS,PP,NL,ND,2,4,11,IPLT)
       CALL CONLIM(AC,NL,CL,NDATA,IPLT,0)
```

```
      CALL FINITT(10,10)
      END IF
         IF(ND.EQ.0)PRINT 1000
         IF(ND.EQ.1)PRINT 1001
         IF(ND.EQ.2)PRINT 1002
1000  FORMAT('1    AUTO AND PARTIAL CORRELATIONS OF ORIGINAL SERIES '//)
1001  FORMAT('1 AUTO AND PARTIAL CORRELATIONS OF FIRST DIFFERENCES OF SE
     *RIES '//)
1002  FORMAT('1 AUTO AND PARTIAL CORRELATIONS OF SECOND DIFFERENCES OF S
     *ERIES '//)
         PRINT 1003
1003    FORMAT(9X,'I',14X,'AUTO',14X,'PARTIAL' //)
      DO 120 I=1,NL
120   PRINT 1004, I,AC(I),PP(I)
1004    FORMAT (8X,I2,13X,F6.3,13X,F6.3)
*
*                     CALCULATE AND PRINT STATISTICS FOR CURVE
*
      CALL CONLIM(AC,NL,CL,NDATA,0,0)
      PRINT 1005,CL
1005  FORMAT(//' APPROX. 95 PERCENT CONF. LIMIT ON CORRELATIONS = ',F5.3
     1)
      PRINT 1006, SDZ
1006   FORMAT(/' STANDARD DEVIATION OF SERIES =',E14.7)
      CHI = 0.
      DO 130 I=1,NL
130   CHI = CHI + AC(I)*AC(I)/FLOAT(NDATA-I)
      CHI = CHI*FLOAT(NDATA*NDATA)
      PRINT 1007, CHI,NL
1007  FORMAT(/' CHI-SQUARED STATISTIC = ',E12.4,' BASED ON ',I2,' DEGREE
     1S OF FREEDOM'/)
      ND = ND+1
      CALL CNTNU
      END DO
      RETURN
      END


      SUBROUTINE INCR(VNEW,V,NV)
      DIMENSION V(NV)
      NV1=NV-1
      DO 100 I=1,NV1
100   V(I)=V(I+1)
      V(NV)=VNEW
      RETURN
      END


      SUBROUTINE IMPLSE(NOBS,NLAG,NDIF,IPLOT)
```

```
      COMMON /DATA/T(1000),U(1000),Y(1000)
      REAL*8    DEV,P(441),TH(21),TST(21),FLAM
      DIMENSION RESP(101),STP(101),AXIS(101)
      DIMENSION YD(1000),UD(1000),R(1000),PCL(101),BCL(101)
*
*             INITIALIZE THE COV. MATRIX AND PARAMETERS
*
      IF(IPLOT.GT.0) THEN
        CALL ANMODE
        WRITE(6,*)' ~ERA W  ~WOR 30 '
      END IF
      IPLT=IPLOT
      NOB=NOBS
      NL=NLAG
      NP=NL+1
      ND=NDIF
      FLAM=1.0
      DO 5 I=1,NP
    5 AXIS(I)=FLOAT(I-1)
*
**************** NOTE THAT SERIES ARE NOT YET DIFFERENCED (STATIONARY)
*
      CALL STATS(U,NOB,SDU,UBAR)
      CALL STATS(Y,NOB,SDY,YBAR)
      NDLOOP=0
      DO WHILE (NDLOOP.LE.ND)
      NPSQ=NP*NP
      DO 90 I=1,NPSQ
90    P(I)=0.0
      NY=NOB
      CALL DIFF(U,NOB,NDLOOP,UD,UBAR)
      CALL DIFF(Y,NY,NDLOOP,YD,YBAR)
*
*                               SET DIAGONAL OF COV. MATRIX (P) TO ESTIMATE
*                               ALSO INITIALIZE TH(I) ESTIMATES TO ZERO
*
      EST=1000.*(SDU*SDU)
      DO 100 I=1,NP
        TH(I)=0.0
        I1=I+(I-1)*NP
        P(I1)=EST
100   CONTINUE
*                               BEGIN ESTIMATION
      LOC=NP
      DO WHILE (LOC.LE.NOB)
        DO 110 I=1,NP
110     TST(I)=UD(LOC-I+1)/SDU
        DEV=YD(LOC)/SDY
        LOC=LOC+1
        CALL RLS(DEV,P,TST,TH,FLAM,NP)
      END DO
```

```
*
*       NOTE THAT THE V WEIGHTS ARE SCALED BY (SDU/SDY) SO THAT U,Y SCALED.
*       THE TH VALUES ARE THE V WEIGHTS, REFERENCE PAGE 379 BOX + JENKINS
*       THIS METHOD SHOULD BE BETTER CONDITIONED AS A SOLUTION THAN THE
*       MATRIX INVERSION METHOD.
*
      DO 115 I=1,NP
115   RESP(I)=TH(I)*(SDY/SDU)
      DO 130 I=NP,NOB
        IMNP=I-NP+1
        SUMA=0.0
          DO 120 J=1,NP
120       SUMA=SUMA+RESP(J)*UD(I-J+1)
*       NOTE THAT YD IS NOW THE OBSERVED YD - THE BEST ESTIMATE OF YD(IE. SUMA)
*       THIS IS THEN THE NOISE ESTIMATE ON YD.
        R(IMNP)=YD(I)-SUMA
130   CONTINUE
      NR=NOB-NP
      CALL STATS(R,NR,SDR,RBAR)
*
*       SET THE CONFIDENCE LIMITS (CL) FOR THE IMPULSE WEIGHTS
*       PCL = POSITIVE CL , BCL = BOTTOM CL
*
        DO 140 I=1,NP
        II=I+NP*(I-1)
        PCL(I)=2.0*SQRT(P(II))*SDR/SDU
        BCL(I)=-PCL(I)
140     CONTINUE
*
*             CALCULATE THE STEP RESPONSE
*
      STP(1)=RESP(1)
      DO 150 I=2,NP
      STP(I)=RESP(I)+STP(I-1)
150   CONTINUE
*
*             PLOT AND PRINT IMPULSE AND STEP RESPONSES
*
      IF(IPLT.GT.0) THEN
       CALL ANMODE
       WRITE(6,*)'"ERA W  "WOR 30 '
        CALL PLOTXY(AXIS,RESP,NP,NDLOOP,2,S,11,3)
*                   PLACE CONFIDENCE LIMITS ONTO IMPULSE RESPONSE CURVE
        CALL MOVEA(AXIS(1),PCL(1))
         DO 154 I=2,NP
154      CALL DASHA(AXIS(I),PCL(I),3)
        CALL MOVEA(AXIS(1),BCL(1))
         DO 156 I=2,NP
156      CALL DASHA(AXIS(I),BCL(I),3)
        CALL FINITT(10,10)
        CALL CNTNU
```

```
      CALL PLOTXY(AXIS,STP,NP,NDLOOP,2,7,11,3)
      CALL FINITT(10,10)
    END IF
    PRINT 1000
    DO 160 I=1,NP
    IM1=I-1
160 PRINT 1001,IM1,RESP(I),STP(I)
1000 FORMAT(//32H0       LAG,K      IMPULSE   STEP)
1001 FORMAT(/,5X,I5,5X,2E12.4)
    CALL CNTNU
    NDLOOP=NDLOOP+1
    END DO
    RETURN
    END


    SUBROUTINE MATIN(A, NV, B, NB1, DET)
    DIMENSION A(NV, NV), B(NV, 1)
*
*             SUBROUTINE TO SOLVE MATRIX EQ'N   AX = B
*             RETURNS INVERSE OF A IN MATRIX A
*             RETURNS SOLUTION X VECTOR IN B IF NB1=1
*             RETURNS DETERMINANT OF A AS DET.
*                    REFERENCE  PAGE 3 SHARP PC1211 MANUAL(SAME METHOD)
*
    NVAR=NV
    NB=NB1
    DET = 1.0
    ICOL=1
      DO WHILE(ICOL.LE.NVAR)
      PIVOT = 1.0/ (AMAX1(A(ICOL, ICOL) , 1.E-20))
      DET = A(ICOL, ICOL) * DET
*
*             DIVIDE PIVOT ROW BY PIVOT ELEMENT (DIAGONAL CHOSEN AS PIVOT)
*             CAUTION:  IF DIAGONAL ELEMENT IS ZERO, PROBLEMS ARISE
*                       ROUTINE WORKS BEST WITH DIAGONALLY DOMINANT MATRICES
*
      A(ICOL, ICOL) = 1.0
      DO 100 L=1,NVAR
100     A(ICOL, L) = A(ICOL, L)*PIVOT
      IF(NB .NE. 0) THEN
        DO 200 L=1,NB
200     B(ICOL, L) = B(ICOL, L)*PIVOT
      END IF
*
*   REDUCE NON-PIVOT ROWS
*
      L1=1
      DO WHILE (L1.LE.NVAR)
      IF(L1 .NE. ICOL) THEN
```

```
          T = A(L1, ICOL)
          A(L1, ICOL) = 0.
          DO 300 L=1,NVAR
 300        A(L1, L) = A(L1, L) - A(ICOL, L)*T
          IF(NB .NE. 0) THEN
            DO 400 L=1,NB
 400          B(L1, L) = B(L1, L)-B(ICOL,L)*T
          END IF
       END IF
       L1=L1+1
       END DO
       ICOL=ICOL+1
      END DO
      RETURN
      END


      SUBROUTINE MATPRT(ITYP, NQ, A, B, C)
      DIMENSION A(NQ),B(NQ),C(NQ,NQ)
*
*          SUBROUTINE TO PRINT MATRICES  A,B,C USING A FORMAT
*          SELECTED BY VALUE OF ITYPE.  MAX OF 10 VALUES PER LINE OUTPUT
*
       ITYPE=ITYP
       NP = NQ
       NR = NP/10
       LOW = 1
       LUP = 10
        DO WHILE (NR.GE.0)
        IF(NR.LE.0)THEN
          LUP=NP
*          THEREFORE  NP.LT.10, CHECK FOR RETURN
          IF(LOW .GT. LUP) RETURN
        END IF
*****                          PRINT HEADING
      PRINT 500, (J,J=LOW,LUP)
        IF(ITYPE.EQ.3) THEN
*                         PRINT LOWER TRIAGONAL MATRIX C
*                         NOTE LUP=NP IF NR.LE.0
          DO 90 I=LOW,LUP
 90       PRINT 720,I,(C(J,I),J=LOW,I)
        LOW2=LUP+1
*                         PRINT THE REST OF TRIAGONAL MATRIX C
        IF(LOW2 .LE. NP) THEN
          DO 95 I=LOW2,NP
 95       PRINT 720,I,(C(J,I),J=LOW,LUP)
        END IF
        END IF
*          ITYPE = 2 PRINT BOTH VECTORS B, THEN A.
*          ITYPE = 1 PRINT ONLY VECTOR A.
```

```
      IF(ITYPE.EQ.2)PRINT 600, (B(J),J=LOW,LUP)
      IF(ITYPE.EQ.2 .OR. ITYPE.EQ.1)PRINT 600, (A(J),J=LOW,LUP)
*             SET PARAMETERS FOR NEXT CYCLE IF NP.GT.10
     LOW = LOW + 10
     LUP = LUP + 10
     NR = NR - 1
       END DO
 500 FORMAT(/I8,9I12)
 600 FORMAT(10E12.4)
 720 FORMAT(1H ,I3,1X,F7.4,9F12.4)
     RETURN
     END


     SUBROUTINE MODEL(NPRBO,TH,A,NOBS,NPAR,IFLAG)
     COMMON /DATA/T(1000),U(1000),Y(1000)
     COMMON /DATA1/YNEW,Y10(10),U10(10),A10(10),P1(10),P2(10),P3(10),NPOLY
     DIMENSION A(NOBS),TH(NPAR)
     DIMENSION THETA(10),SIGMA(10),OMEGA(10),DELTA(10)
*
*             SUBROUTINE TO CREATE AN ARRAY OF RESIDUALS FOR THE MODEL;
*             Y(T) = (OMEGA(B)/DELTA(B))*U(T-NB) + (THETA(B)/SIGMA(B))*A(T)
*             WHERE Y(T) AND U(T) HAVE BEEN DIFFERENCED ND TIMES (STATIONARY)
*             THETA(B) = POLYNOMIAL IN B OF ORDER NP
*             SIGMA(B) = POLYNOMIAL IN B OF ORDER NQ
*             OMEGA(B) = POLYNOMIAL IN B OF ORDER NR
*             DELTA(B) = POLYNOMIAL IN B OF ORDER NS
*             THE MEASUREMENTS Y(T) OBSERVE THE CONTROL ACTION U(T-NB) LAG NB
*
*             IFLG = -2 SAME AS FOR -1 BUT FOR STUDENT 11 TO CREATE PLANT
*                    -1 STORE NEW MODEL INFO. ON TAPE NT1 INTERACTIVELY
*                     0 USE NEW MODEL INFO. RECEIVED IN VECTOR TH(NPAR)
*                     1 USE EXISTING MODEL INFO. ON TAPE NT1 FOR TH(NPAR)
*                     2 CREATE OUTPUT Y(T) INSTEAD OF A(T),OTHERWISE IFLG=1
*                     3 CREATE SINGLE OUTPUT YNEW FOR PRESENT MODEL  /DATA1/
*
     NPROB=NPRBO
     NT=NPROB
     NOB=NOBS
     NPR=NPAR
     IFLG=IFLAG
     IF(IFLG.NE.3) THEN
*                                      INITIALIZE PARAMETER VECTORS
       DO 100 I=1,10
       THETA(I)=0.0
       SIGMA(I)=0.0
       OMEGA(I)=0.0
       DELTA(I)=0.0
 100   CONTINUE
     IF(IFLG.LT.0)THEN
```

```
*       CALL INTERACTIVE ROUTINE TO SET UP TAPE NT1 WITH MODEL PARAMETERS
*       THIS ROUTINE MUST BE CALLED BEFORE MODEL IS USED BY ANY PROGRAM.
        CALL INTMDL(NT,TH,NPR,NOB,IFLG)
        NOBS=NOB
        NPAR=NPR
        RETURN
      END IF
*
      IF(IFLG.EQ.0) THEN
*       STORE NEW MODEL PARAMETER VALUES RECEIVED
        REWIND(NT)
        READ(NT)NP,NQ,NR,NS,ND,NB
        IF(NP.NE.0)CALL WTBF(TH(1),NP,NT)
        IF(NQ.NE.0)CALL WTBF(TH(NP+1),NQ,NT)
        IF(NR.NE.0)CALL WTBF(TH(NQ+NP+1),NR,NT)
        IF(NS.NE.0)CALL WTBF(TH(NR+NQ+NP+1),NS,NT)
      END IF
*       READ THE MODEL PARAMETERS FROM TAPE
        REWIND(NT)
        READ(NT)NP,NQ,NR,NS,ND,NB
        IF(NP.NE.0)CALL RDBF(THETA,NP,NT)
        IF(NQ.NE.0)CALL RDBF(SIGMA,NQ,NT)
        IF(NR.NE.0)CALL RDBF(OMEGA,NR,NT)
        IF(NS.NE.0)CALL RDBF(DELTA,NS,NT)
*
*               TRANSLATE THE MODEL INTO A NEW FORM
*               P1(B)*Y(T) = P2(B)*U(T-NB) + P3(B)*A(T)
*               THEN P1(B) = DELTA(B)*SIGMA(B)
*                    P2(B) = OMEGA(B)*SIGMA(B)
*                    P3(B) = THETA(B)*DELTA(B)
*
        P1(1)=1.0
        P2(1)=OMEGA(1)
        P3(1)=1.0
        DO 120 I=2,10
          P1(I)=DELTA(I-1) + SIGMA(I-1)
          P2(I)=OMEGA(I) + OMEGA(1)*SIGMA(I-1)
          P3(I)=DELTA(I-1) + THETA(I-1)
          IM1=I-1
          IM2=I-2
          IF(I.GT.2) THEN
            DO 110 J=1,IM2
              P1(I)=P1(I)+DELTA(IM1-J)*SIGMA(J)
              P2(I)=P2(I)+OMEGA(IM1-J+1)*SIGMA(J)
              P3(I)=P3(I)+DELTA(IM1-J)*THETA(J)
110         CONTINUE
          END IF
120     CONTINUE
*
*               ACCOUNT FOR ND DIFFERENCING IN MODEL FOR CREATING DATA(IFLG=2)
        IF(IFLG.EQ.2) THEN
```

```
          CALL POLY(P1,ND,10)
          CALL POLY(P2,ND,10)
        END IF
*
*             TEST FOR NUMBER OF NON ZERO POLYNOMIAL TERMS
*
        I=1
        DO WHILE (P1(I).NE.0.0 .OR. P2(I).NE.0.0 .OR. P3(I).NE.0.0)
        I=I+1
        END DO
        NPOLY=I-1
        IF(NPOLY.GT.9)NPOLY=9
*             NUM IS THE MAX NUMBER OF SUBSCRIPTS BACKWARDS, WHICH
*             ARE REFERENCED IN THE ITERATIVE EQUATION FOR A(I)
        NUM=I+NB
        IF(NUM.GT.10)THEN
         WRITE(6,*)'MORE THAN TEN BACKWARDS OPERATORS IN MODEL',NUM
         WRITE(6,*)'CALL EXIT'
         CALL EXIT
        END IF
      END IF
***********************************************************************
*
         IF(IFLG.GE.2)THEN
*                               HERE Y(I) IS VECTOR OF OBSERVATIONS RETURNED
*                               AND A(I) IS VECTOR OF RANDOM DISTURBANCES
*                               U(I) IS THE CONTROL ACTION
          IA=1
           IF(IFLG.EQ.2) THEN
           DO 130 I=1,10
            Y10(I)=0.0
            U10(I)=0.0
            A10(I)=0.0
130        CONTINUE
           END IF
*                              HERE NPR IS THE INTEGER SAMPLE TIME
*                              THE MODEL IS SOLVED FOR EACH TIME INTERVAL = 1
*                              OBSERVATIONS AND CONTROL ARE TAKEN EACH NPR INT.
          DO 150 K=1,NOB
           DO 145 I=1,NPR
            YNEW=0.0
            DO 140 J=1,NPOLY
        YNEW=YNEW+P2(J)*U10(11-J-NB)+P3(J)*A10(11-J)-P1(J+1)*Y10(11-J)
140         CONTINUE
            CALL INCR(YNEW,Y10,10)
            CALL INCR(A(IA),A10,10)
            IA=IA+1
            IF(IA.GE.1000) IA=1
            IF(I.NE.NPR) CALL INCR(U10(10),U10,10)
145        CONTINUE
           T(K)=FLOAT(K*NPR)
```

```
          Y(K)=YNEW
          CALL CONTROL(UNEW,U10,Y10,T(K))
          IF(UNEW.GT.10.)UNEW=10.
          IF(UNEW.LT.-10.)UNEW=-10.
          U(K)=UNEW
          CALL INCR(UNEW,U10,10)
150      CONTINUE
       ELSE
*****************************************************************************
*
*               TYPICAL TSHAUS CALL TO MODEL
******          CREATE THE OUTPUT VECTOR A(I) TO BE RETURNED
*
      DO 160 I=1,NUM
160   A(I)=0.0
*               NOTE THAT A(I) UP TO NUM-1 ARE DESIRED TO BE ZERO
*               THE VALUE FOR A(NUM) UP TO NOB MUST BE COMPUTED
      DO 180 I=NUM,NOB
        A(I)=0.0
*                               HERE A(I) IS VECTOR OF RESIDUAL MODEL NOISE
*                               AND Y(I) IS VECTOR OF OBSERVATIONS IN /DATA/
          DO 170 J=1,NPOLY
          A(I)=A(I)+P1(J)*Y(I-J+1)-P2(J)*U(I-J+1-NB)-P3(J+1)*A(I-J)
170       CONTINUE
180   CONTINUE
        END IF
      RETURN
      END


      SUBROUTINE PARTAL(AC,PAUTO,NLAG)
      DIMENSION AC(NLAG),PAUTO(NLAG),PHAT(101),PHATN(101)
*
*                    PARTIAL AUTO-CORRELATION ROUTINE
*                    AC = AUTO CORRELATION VECTOR RECEIVED
*                    PAUTO = PARTIAL AUTO CORRELATION RETURNED
*                    NLAG = NUMBER OF LAGS
*                    PHAT(J) = PHI(LAG-1,J), PAUTO(J) = PHI(LAG,LAG)
*                            SEE REFERENCE FOR PHI.
*    FORMULAE PAUTO(1),(2)   REFERENCE PAGE 64 BOX + JENKINS
*    FORMULAE PAUTO(NLAG)    REFERENCE PAGE 497 BOX + JENKINS
*
      NL=NLAG
      PAUTO(1) = AC(1)
      PHAT(1) = AC(1)*(1.-AC(2))/(1.-AC(1)**2)
      PHAT(2) = (AC(2)-AC(1)**2)/(1.-AC(1)**2)
      PAUTO(2) = PHAT(2)
      LAG=3
        DO WHILE (LAG.LE.NL)
        NLM1 = LAG-1
```

```
        FNUM = 0.0
        DENOM = 0.0
         DO 100 J=1,NLM1
         FNUM = PHAT(J)*AC(LAG-J)+FNUM
100      DENOM = DENOM+PHAT(J)*AC(J)
        PHATN(LAG) = (AC(LAG)-FNUM)/(1.-DENOM)
        PAUTO(LAG) = PHATN(LAG)
         DO 200 J=1,NLM1
200      PHATN(J) = PHAT(J)-PHATN(LAG)*PHAT(LAG-J)
         DO 300 J=1,LAG
300      PHAT(J) = PHATN(J)
        LAG=LAG+1
        END DO
      RETURN
      END


      SUBROUTINE POLY(P,NDIF,NP)
      DIMENSION P(NP),PNEW(10)
*               SUBROUTINE TO MULTIPLY A POLYNOMIAL IN THE BACKWARDS
*               DIFFERENCE OPERATOR "B" BY (1-B) (IE. "DEL").  THIS
*               WILL ACCOUNT FOR THE DEL**ND TERM IN THE GENERAL MODEL
*
        ND=NDIF
        IF(ND.LE.0.OR.ND.GT.2) RETURN
        DO 1000 K=1,ND
          PNEW(1)=P(1)
          DO 100 I=1,10
            IM1=I-1
            PNEW(I)=P(I)-P(IM1)
100       CONTINUE
1000    CONTINUE
        DO 1001 I=1,10
1001    P(I)=PNEW(I)
      RETURN
      END


      SUBROUTINE RLS(Y,COV,PHI,THETA,FL,NP1)
      REAL*8 Y,COV(441),PHI(NP1),THETA(NP1),S(21)
      REAL*8 ERRS,K(21),SUM,FL,DEN
*               RECURSIVE LEAST SQUARES ROUTINE
*               Y = MOST RECENT ERROR OBSERVED
*               COV = COVARIANCE MATRIX
*               PHI = LAST NP1 OBSERVATIONS
*               THETA = MOST RECENT PARAMETER ESTIMATES (TO BE UPDATED)
*               FL = 1.0 FOR NORMAL LEAST SQUARES
*                  = .LT. 1.0 FOR EXPONENTIALLY DISCOUNTED LEAST SQUARES
*               EFFECTIVE WINDOW LENGTH (1/1-FL) EFFECTIVE NO.PAST OBS.IN USE
```

```
*                        REFERENCE COURSE 704 JAN 5/81 COURSE NOTES.
*
      NP=NP1
      SUM=0.0
      DO 100 I=1,NP
      DO 100 J=1,NP
      IJ=J+(I-1)*NP
      SUM=SUM+PHI(I)*PHI(J)*COV(IJ)
100   CONTINUE
      DEN=SUM+FL
      ERRS=0.0
      DO 300 I=1,NP
        SUM=0.0
        DO 200 J=1,NP
          IJ=J+(I-1)*NP
          SUM=SUM+COV(IJ)*PHI(J)
200     CONTINUE
        S(I)=SUM
        K(I)=SUM/DEN
        ERRS=ERRS+PHI(I)*THETA(I)
300   CONTINUE
*
      ERRS=Y-ERRS
      DO 500 I=1,NP
        DO 400 J=1,I
          IJ=J+(I-1)*NP
          COV(IJ)=(COV(IJ)-S(I)*S(J)/DEN)/FL
          JI=I+(J-1)*NP
          COV(JI)=COV(IJ)
400     CONTINUE
          THETA(I)=THETA(I)+K(I)*ERRS
500   CONTINUE
*
      RETURN
      END


      SUBROUTINE STATS(X,NOBS,SDX,XBAR)
      DIMENSION X(NOBS)
*
*      FIND THE MEAN OF X
*
      NOB=NOBS
      SUMX=0.0
      SUMXSQ=0.0
      XBAR=0.0
      SDX=0.0
      RNOB=FLOAT(NOB)
      IF(NOB.LE.1) RETURN
*                          SUM OF X, X*X
```

```
      DO 100 I=1,NOB
        SUMX=SUMX+X(I)
        SUMXSQ=SUMXSQ+X(I)*X(I)
100   CONTINUE
      XBAR=SUMX/RNOB
*
*       FIND THE STANDARD DEVIATION OF X
*
      SDX=SQRT((SUMXSQ-SUMX*SUMX/RNOB)/RNOB)
      RETURN
      END


      SUBROUTINE TSHAUS(NPROB,NOBS,NP,TH,EPS1,EPS2,MIT,FLAM,FNU,
     1 SCRAT,LEN,IPLOT)
*
*       SUBROUTINE TO SET UP DIMENSIONS OF VARIABLES FOR HAUSTS ROUTINE
*       RESIDUALS ARE RETURNED IN SCRAT(IG) WHERE IG=5*NP+1
*       MINIMUM VALUE OF LEN = 5*NP+NP*NP+NOBS+NP*NOBS
      DIMENSION SCRAT(LEN)
        IF(IPLOT.GT.0) THEN
        CALL ANMODE
        WRITE(6,*)' ~ERA W  ~WOR O '
        END IF
      NOB=NOBS
      IPLT=IPLOT
      IA=1
      IB=IA+NP
      IC=IB+NP
      ID=IC+NP
      IE=ID+NP
      IG = IE+NP
      IH=IG+NOB
      II = IH + NP * NOB
      IJ = IH
*
          CALL  HAUSTS(NPROB,NOB,NP,TH,   EPS1,EPS2,MIT
     1,FLAM,FNU,  SCRAT(IA), SCRAT(IB), SCRAT(IC), SCRAT(ID),
     2  SCRAT(IE),    SCRAT(IG), SCRAT(IH), SCRAT(II),SCRAT(IJ),IPLT)
      RETURN
      END


      SUBROUTINE CURVE(X,Y,NOBS,IPTN,ISYMBL)
      DIMENSION X(NOBS),Y(NOBS)
      CHARACTER         IN*1
*
*         THIS SUBROUTINE JOINS THE LINES BETWEEN THE POINTS
*         (X(I),Y(I)) AND (X(I+1),Y(I+1)) TO FORM A SMOOTH CURVE.
```

```
*          NOB IS THE DIMENSION OF THE X AND Y ARRAYS.
*          IOPT DETERMINES WHETHER A LINE OR BAR CHART IS PRODUCED.
*                 IOPT = 0        A DOTTED LINE
*                 IOPT = 1        A SOLID LINE
*                 IOPT = 2        A BAR CHART
*
      NOB=NOBS
      ISYM=ISYMBL
      IOPT=IPTN
      IN=' '
      IF(ISYM .EQ. 1) IN='X'
      IF(ISYM .EQ. 2) IN='O'
      IF(ISYM .EQ. 3) IN='I'
      IF(ISYM .EQ. 4) IN='*'
      IF(ISYM .EQ. 5) IN='+'
      IF(IOPT.NE.2) THEN
*
*      PLOT A LINE
*
      IDOT=1
      IF(IOPT.EQ.0)IDOT=2
      CALL MOVEA(X(1),Y(1))
      DO 10 I=2,NOB,IDOT
        CALL MOVEA(X(I-1),Y(I-1))
        CALL DRAWA(X(I),Y(I))
        IF(MOD(I,NOB/10) .NE. 0 .OR. IN.EQ.' ') GO TO 10
        CALL ANMODE
        WRITE(6,1000) IN
10     CONTINUE
      ELSE
*
*       PLOT A BAR CHART
*
       DO 20 I=1,NOB
        CALL MOVEA(X(I),0.0)
        CALL DRAWA(X(I),Y(I))
        CALL ANMODE
        WRITE(6,1000) IN
20      CONTINUE
      END IF
1000    FORMAT(1H ,'~STR/',A1,'/')
      RETURN
      END


      SUBROUTINE FRAME(HDG,HDGX,HDGY,XSC,YSC,NOBS)
      CHARACTER*10   HDG(8),HDGX(8),HDGY(3),LISTY
      CHARACTER*6    IFORMX,IFORMY,ZZ
      DIMENSION XSC(NOBS),YSC(NOBS)
      IFORMX='(F5.1)'
```

```
      IFORMY='(F5.1)'
      NOB=NOBS
*
*          THIS SUBROUTINE PRODUCES A BASIC FRAME FOR PLOTTING.
*          XMIN   - MINIMUM VALUE FOR X AXIS
*          XMAX   - MAXIMUM VALUE FOR X AXIS
*          YMIN   - MINIMUM VALUE FOR Y AXIS
*          YMAX   - MAXIMUM VALUE FOR Y AXIS
*          IFORMX - FORMAT FOR PRINTING NUMBERS UNDER X AXIS TICKS
*          IFORMY - FORMAT FOR PRINTING NUMBERS BESIDE Y AXIS TICKS
*          HDG    - HEADING TO BE PRINTED AT THE TOP OF THE PLOT
*          HDGX   - TITLE UNDER THE X AXIS
*          HDGY   - TITLE BESIDE THE Y AXIS
*          XP     - DISTANCE BETWEEN TICKS AND NUMBERS ON THE X AXIS
*          YP     - DISTANCE BETWEEN TICKS AND NUMBERS ON THE Y AXIS
*
*
*       DETERMINE MAX AND MIN VALUES, ROUNDED OFF, FOR EACH AXIS
*
      CALL VMAX(XSC,NOB,XMAX)
      IF(XMAX.LT.0.)XMAX=XMAX+1.0
      CALL VMIN(XSC,NOB,XMIN)
      XMIN=FLOAT(IFIX(XMIN-1.0))
      IF(XMIN.GT.0.)XMIN=XMIN-1.0
      CALL VMAX(YSC,NOB,YMAX)
      YMAX=FLOAT(IFIX(YMAX+1.0))
      IF(YMAX.LT.0.)YMAX=YMAX+1.0
      CALL VMIN(YSC,NOB,YMIN)
      YMIN=FLOAT(IFIX(YMIN-1.0))
      IF(YMIN.GT.0.)YMIN=YMIN-1.0
*       ALLOW 10 PERCENT OF AXIS FOR TITLES
        XST=(XMAX-XMIN)*0.1
        YST=(YMAX-YMIN)*0.1
*
*       DRAW THE FRAME
*
      CALL INITT(60)
      CALL TWINDO(10,1010,10,600)
      CALL DWINDO(XMIN-XST,XMAX,YMIN-YST,YMAX+YST)
      CALL MOVEA(XMIN,YMIN)
      CALL DRAWA(XMIN,YMAX)
      CALL DRAWA(XMAX,YMAX)
      CALL DRAWA(XMAX,YMIN)
      CALL DRAWA(XMIN,YMIN)
        YST=YST*0.5
*                                  PRINT MAIN HEADING
        CALL MOVEA(XMIN-XST,YMAX+YST)
        XST=XST*0.85
        CALL ANMODE
        WRITE(6,1000)(HDG(I),I=1,8)
*
```

```
*           Y AXIS TICKS
*
       YP=0.2
       IF(ABS(YMAX-YMIN).GT.1.0)YP=0.5
       IF(ABS(YMAX-YMIN).GT.2.0)YP=1.0
       IF(ABS(YMAX-YMIN).GT.5.0) YP=2.0
       IF(ABS(YMAX-YMIN).GT.10.0)YP=5.0
       IF(ABS(YMAX-YMIN).GT.50.0)YP=10.0
       IN=IFIX((YMAX-YMIN)/YP+1.005)
       TLEN=(XMAX-XMIN)/100.
*                        FORCE TICKS ON AXIS TO LINE UP WITH 0.0
         YY=YMIN
       IF(YMAX.GT.0.0 .AND. YMIN.LT.0.0)THEN
         YY=0.0
         DO WHILE ((YY-YP).GE.YMIN)
           YY=YY-YP
         END DO
       END IF
       DO 50 K=1,IN
       IF(YY.LE.YMAX) THEN
         CALL MOVEA(XMIN,YY)
         CALL DRAWA(XMIN+TLEN,YY)
         CALL MOVEA(XMAX-TLEN,YY)
         CALL DRAWA(XMAX,YY)
         CALL MOVEA(XMIN-XST,YY)
         ENCODE(6,IFORMY,ZZ)YY
         CALL ANMODE
           WRITE(6,1001) ZZ
         YY=YY+YP
       END IF
50     CONTINUE
*
*          PLOT Y AXIS
*
       IF(XMAX.GT.0.0 .AND. XMIN.LT.0.0)THEN
       CALL MOVEA(0.0,YMIN)
       CALL DRAWA(0.0,YMAX)
       END IF
*
*          X AXIS TICKS
*
       XST=(XST/0.85)*0.6
       XP=0.2
       IF(ABS(XMAX-XMIN).GT.1.0)XP=0.5
       IF(ABS(XMAX-XMIN).GT.2.0)XP=1.0
       IF(ABS(XMAX-XMIN).GT.5.0)XP=2.0
       IF(ABS(XMAX-XMIN).GT.10.0)XP=5.0
       IF(ABS(XMAX-XMIN).GT.50.0)XP=10.0
       IN=IFIX((XMAX-XMIN)/XP+1.005)
       TLEN=(YMAX-YMIN)/100.
       XX=XMIN
```

```
*               FORCE TICKS ON AXIS TO LINE UP WITH 0.0
          IF(XMAX.GT.0.0 .AND. XMIN.LT.0.0)THEN
            XX=0.0
            DO WHILE((XX-XP).GE.XMIN)
              XX=(XX-XP)
            END DO
          END IF
        DO 100 K=1,IN
        IF(XX.LE.XMAX) THEN
        CALL MOVEA(XX,YMIN)
        CALL DRAWA(XX,YMIN+TLEN)
        CALL MOVEA(XX,YMAX-TLEN)
        CALL DRAWA(XX,YMAX)
        CALL MOVEA(XX-XST,YMIN-YST)
        ENCODE(6,IFORMX,ZZ)XX
        CALL ANMODE
          WRITE(6,1001) ZZ
        XX=XX+XP
        END IF
100     CONTINUE
*
*             PLOT X AXIS
*
        IF(YMAX.GT.0.0 .AND. YMIN.LT.0.0)THEN
        CALL MOVEA(XMIN,0.0)
        CALL DRAWA(XMAX,0.0)
        END IF
*
*           PUT HEADINGS ON AXES
*
        CALL MOVEA(XMIN,YMIN-2*YST)
        CALL ANMODE
          WRITE(6,1000)(HDGX(I),I=1,8)
        XST=(XMAX-XMIN)*0.1
        CALL MOVEA(XMIN-XST,YMAX)
        CALL ANMODE
        WRITE(6,*)'~RVE 0,-23'
        DO 150 I=1,3
         LISTY(1:10)=HDGY(I)
         DO 145 J=1,10
         IF(LISTY(J:J).EQ.' ') THEN
          JM1=J-1
          IF(JM1.GT.0) THEN
           IF(LISTY(JM1:JM1).NE.' ')WRITE(6,*)' ~RVE 0,-23'
          ELSE
           WRITE(6,*)'~RVE 0,-23'
          END IF
         ELSE
         WRITE(6,1002)LISTY(J:J)
         END IF
145      CONTINUE
```

```
150   CONTINUE
1000    FORMAT(1H ,'~STR/',8A10,'/')
1001    FORMAT(1H ,'~STR/',A6,'/')
1002    FORMAT(1H ,'~STR/',A1,'/ ~RVE 0,-23')
      RETURN
      END


      SUBROUTINE VMAX(X,NX,XMAX)
      DIMENSION X(NX)
*                IN VECTOR X FIND MAXIMUM VALUE
      XMAX=X(1)
      DO 10 K=1,NX
      IF(X(K).GT.XMAX)XMAX=X(K)
   10 CONTINUE
      RETURN
      END


      SUBROUTINE VMIN(X,NX,XMIN)
      DIMENSION X(NX)
*                IN VECTOR X FIND MINIMUM VALUE
      XMIN=X(1)
      DO 10 K=1,NX
      IF(X(K).LT.XMIN)XMIN=X(K)
   10 CONTINUE
      RETURN
      END


      SUBROUTINE PLOTXY(X,Y,NOBS,IHDG,IHX,IHY,IPTN,IPLOT)
      DIMENSION X(NOBS),Y(NOBS),XS(1000),YS(1000)
      CHARACTER*10   HG(8),HX(8),HY(3)
*
*                SUBROUTINE TO PLOT X VERSUS Y WITH TITLES IN ONE CALL
*
*           IOPT LT 10  PLOT LINE CURVE
*                GE 10  PLOT BAR CHART
*           IPLT = 1    OUTPUT SERIES IN USE FOR TITLES
*                = 2    CONTROL VARIABLE SERIES IN USE
*                = 3    NEITHER IN USE
*
       IF(IPLOT.LE.0) RETURN
       CALL ANMODE
       WRITE(6,*)' ~ERA W  ~GRA 1,35  ~SHRINK YES '
       NO3=NOBS
       IH=IHDG
       IX=IHX
```

```
      IY=IHY
      IOPT=IPTN
      IPLT=IPLOT
      IF(IH.GE.0.AND.IPLT.LT.3)IH=IH+(IPLT-1)*10
*            ADD 10 TO IH TO MARK CONTROL HEADING
      ISYM1=IOPT
      IF(IOPT.GE.10)ISYM1=IOPT-10
      IOPT1=1
      IF(IOPT.GE.10)IOPT1=2
      CALL TITLE(IH,IX,IY,HG,HX,HY)
*            SUPPRESS SCALING ON CROSS, AUTO AND PARTIAL CORRELATION CURVES
      IF(IY.EQ.1 .OR. IY.EQ.2 .OR. IY.EQ.8) THEN
        CALL SCALE(X,NOB,HX,7,XS)
        CALL SCALE(Y,NOB,HY,3,YS)
      ELSE
        CALL DIFF(X,NOB,0,XS,0.0)
        CALL DIFF(Y,NOB,0,YS,0.0)
      END IF
      CALL FRAME(HG,HX,HY,XS,YS,NOB)
      CALL CURVE(XS,YS,NOB,IOPT1,ISYM1)
*            RETURN IH TO ORIGINAL VALUE
      IF(IH.GE.0.AND.IPLT.LT.3)IH=IH-(IPLT-1)*10
      RETURN
      END


      SUBROUTINE RDBF(X,LENX,NT)
*     USE ONE BINARY READ TO READ VECTOR X VARIABLE SIZE
      DIMENSION X(LENX)
      READ(NT) X
      RETURN
      END


      SUBROUTINE SCALE(Z,NZ,HDGZ,NHDG,ZS)
      DIMENSION Z(NZ),ZS(NZ)
      CHARACTER*10   HDGZ(NHDG)
      CHARACTER*10   NUM(9)
      DATA NUM/' 10**-4   ',' 10**-3   ',' 10**-2   ',' 10**-1   ',
     +'          ',' 10**1    ',' 10**2    ',' 10**3    ',' 10**4    '/
*
*              SUBROUTINE TO SCALE VECTOR Z BY FACTOR 10**I
*              THIS FACTOR IS ADDED TO THE AXIS TITLE
*              NO SCALING OCCURS IF DIVISION IS BY 0.1,1.0,10.0
*              SCALED VECTOR ZS IS RETURNED, Z IS UNCHANGED.
*
      NOB=NZ
      NH=NHDG
      CALL VMAX(Z,NOB,ZMAX)
```

```
      ZMAX=ABS(ZMAX)
      CALL VMIN(Z,NOB,ZMIN)
      IF(ABS(ZMIN).GT.ZMAX)ZMAX=ABS(ZMIN)
      I=-3
      DO WHILE(ZMAX.GT.(10.**I))
      I=I+1
      END DO
      I=I-1
      IF(I.GE.-1.AND.I.LE.1)I=0
      HDGZ(NH)=NUM(I+5)
      DO 15 K=1,NOB
      ZS(K)=Z(K)/(10.**I)
   15 CONTINUE
      RETURN
      END


      SUBROUTINE TAPMOV(NT,NR,MODE)
*
*     THIS SUBROUTINE WILL MOVE A TAPE FORWARD A DESIRED NO. OF RECORDS
*
      IF(NR.LE.0)RETURN
      IF(NT.LE.0)RETURN
*
*     DECIDE IF TAPE IS BINARY (MODE=1), OR CODED (MODE=0)
*
      IF(MODE.NE.1)THEN
      DO 40 K=1,NR
      READ(NT,500) X
   40 CONTINUE
      ELSE
      DO 60 K=1,NR
      READ(NT) X
   60 CONTINUE
      END IF
  500 FORMAT(8F10.4)
      RETURN
      END


      SUBROUTINE TITLE(IHDG,IHX,IHY,HDG,HDGX,HDGY)
      CHARACTER*10   HDG(8),HDGX(8),HDGY(3)
*
*                         INITIALIZE CHARACTER ARRAYS
*
      DO 100 K=1,8
      HDG(K)='          '
      HDGX(K)='          '
      IF(K.LE.3) HDGY(K)='          '
```

```
100   CONTINUE
      IF (IHX.EQ.0.OR.IHY.EQ.0) THEN
*
*      READ THE TITLES FROM CARDS IF EITHER IHX OR IHY = 0
*
      READ(5,1000)(HDG(I),I=1,8)
      READ(5,1000)(HDGX(I),I=1,8)
      READ(5,1000)(HDGY(I),I=1,3)
1000  FORMAT(8A10)
      RETURN
      END IF
*
*      SET THE X AXIS TITLE
*
      IF(IHX.GT.0.AND.IHX.LE.3) THEN
      GO TO (1,2,3),IHX
    1 HDGX(6)=' TIME     '
      GO TO 1100
    2 HDGX(6)=' LAG, K   '
      GO TO 1100
    3 HDGX(6)='X VARIABLE'
 1100 CONTINUE
      END IF
*
*      SET Y AXIS TITLES
*
      IF(IHY.GT.0.AND.IHY.LE.8) THEN
      GO TO (11,12,13,14,15,16,17,18),IHY
   11 HDGY(1)='OUTPUT MEA'
      HDGY(2)='SUREMENTS'
      GO TO 1200
   12 HDGY(1)='CONTROL VA'
      HDGY(2)='RIABLE    '
      GO TO 1200
   13 HDGY(1)='AUTO CORRE'
      HDGY(2)='LATION    '
      GO TO 1200
   14 HDGY(1)='PARTIAL CO'
      HDGY(2)='RRELATION '
      GO TO 1200
   15 HDGY(1)='CROSS CORR'
      HDGY(2)='ELATION   '
      GO TO 1200
   16 HDGY(1)='IMPULSE WE'
      HDGY(2)='IGHTS     '
      GO TO 1200
   17 HDGY(1)='STEP RESPO'
      HDGY(2)='NSE       '
      GO TO 1200
   18 HDGY(1)='RESIDUALS '
      HDGY(2)='          '
```

```
 1200 CONTINUE
      END IF
*
*     SET MAIN GRAPH HEADING
*
      HDG(1)=HDGY(1)
      HDG(2(()=HDGY(2)
      HDG(3)='  VERSUS   '
      HDG(4)=HDGX(6)
      IF(IHDG.LT.0) RETURN
      IHDGN=IHDG
      IF(IHDG.GE.10)IHDGN=IHDG-10
*
*           SELECT SERIES TITLE
*
      IF(IHDGN.EQ.0)HDG(5)=' ORIGINAL  '
      IF(IHDGN.EQ.1)HDG(5)=' FIRST DIF'
      IF(IHDGN.EQ.2)HDG(5)='SECOND DIF'
      IF(IHDGN.EQ.3)HDG(5)=' RESIDUALS'
      IF(IHDGN.EQ.4)HDG(5)=' PREWHITEN'
      IF(IHDGN.GE.0)HDG(6)=' SERIES    '
      IF(IHY.EQ.1.OR.IHY.EQ.2.OR.IHDGN.EQ.3) RETURN
      HDG(6)=' OUTPUT SE'
      HDG(7)='RIES      '
      IF(IHDG.LT.10) RETURN
      HDG(6)=' CONTROL S'
      HDG(7)='ERIES     '
      RETURN
      END


      SUBROUTINE WTBF(X,LENX,NT)
*     USE ONE BINARY WRITE TO WRITE VECTOR X VARIABLE SIZE
      DIMENSION X(LENX)
      WRITE(NT) X
      RETURN
      END
```

APPENDIX B

SPECIAL FILE LISTINGS

line #	time	control	observation
300	232.	40.	75.8
400	532.	36.	84.3
500	832.	52.	86.4
600	1132.	67.	89.1
700	1432.	74.	76.9
800	1732.	70.	69.4
900	2032.	65.	64.7
1000	2332.	89.	65.9
1100	2632.	81.	68.6
1200	2932.	69.	56.3
1300	3232.	73.	58.3
1400	3532.	78.	63.7
1500	3832.	77.	61.2
1600	4132.	75.	57.8
1700	4432.	51.	58.3
1800	4732.	52.	58.1
1900	5032.	80.	71.8
2000	5332.	80.	72.3
2100	5632.	81.	57.6
2200	5932.	79.	55.7
2300	6232.	70.	54.5
2400	6532.	69.	55.0
2500	6832.	70.	58.0
2600	7132.	69.	59.2
2700	7432.	55.	58.5
2800	7732.	67.	58.7
2900	8032.	68.	67.3
3000	8332.	69.	61.6
3100	8632.	69.	60.8
3200	8932.	69.	60.3
3300	9232.	56.	60.3
3400	9532.	41.	60.8
3500	9832.	38.	68.2
3600	10132.	52.	81.0
3700	10432.	51.	85.9
3800	10732.	42.	77.2
3900	11032.	52.	78.4
4000	11332.	84.	86.3
4100	11632.	78.	81.1
4200	11932.	80.	63.8
4300	12232.	68.	65.2
4400	12532.	67.	63.3
4500	12832.	76.	59.4
4600	13132.	99.	69.4
4700	13432.	119.	64.8
4800	13732.	119.	53.1
4900	14032.	91.	43.8
5000	14332.	98.	41.1
5100	14632.	93.	50.0
5200	14932.	92.	45.9
5300	15232.	93.	45.2

Appendix B.1:

Data File Listing for Worked Example of Chapter 2.

File Format (3F10.4)

5400	15532.	63.	44.3
5500	15832.	44.	43.9
5600	16132.	51.	57.3
5700	16432.	83.	72.0
5800	16732.	95.	68.7
5900	17032.	80.	52.3
6000	17332.	81.	46.3
6100	17632.	78.	51.9
6200	17932.	83.	50.6
6300	18232.	77.	51.6
6400	18532.	77.	49.0
6500	18832.	91.	51.5
6600	19132.	103.	52.1
6700	19432.	74.	45.7
6800	19732.	68.	40.2
6900	20032.	64.	52.1
7000	20332.	69.	55.2
7100	20632.	66.	58.4
7200	20932.	34.	56.4
7300	21232.	34.	59.6
7400	21532.	45.	82.4
7500	21832.	50.	85.0
7600	22132.	63.	80.4
7700	22432.	63.	79.7
7800	22732.	70.	74.0
7900	23032.	82.	74.5
8000	23332.	83.	71.0
8100	23632.	73.	64.7
8200	23932.	73.	63.4
8300	24232.	73.	67.5
8400	24532.	77.	68.0
8500	24832.	73.	67.1
8600	25132.	58.	65.0
8700	25432.	62.	67.1
8800	25732.	66.	74.6
8900	26032.	68.	73.3
9000	26332.	87.	71.4
9100	26632.	104.	70.2
9200	26932.	103.	61.2
9300	27232.	94.	54.3
9400	27532.	64.	52.0
9500	27832.	52.	54.3
9600	28132.	59.	67.4
9700	28432.	47.	75.0
9800	28732.	47.	71.9
9900	29032.	68.	79.8
10000	29332.	59.	81.4
10100	29632.	64.	71.3
10200	29932.	64.	71.0
10300	30232.	64.	73.2
10400	30532.	64.	73.7

10500	30832.	73.	74.2
10600	31132.	72.	74.2
10700	31432.	78.	70.9
10800	31732.	78.	70.7
10900	32032.	73.	67.7
11000	32332.	74.	67.1
11100	32632.	74.	69.4

Appendix B.2: Generalized PLOTXY routine.(PLOT10 library is required also)

```
      SUBROUTINE CURVE(X,Y,NOBS,IPTN,ISYMBL)
      DIMENSION X(NOBS),Y(NOBS)
      CHARACTER        IN*1
*
*          THIS SUBROUTINE JOINS THE LINES BETWEEN THE POINTS
*          (X(I),Y(I)) AND (X(I+1),Y(I+1)) TO FORM A SMOOTH CURVE.
*          NOB IS THE DIMENSION OF THE X AND Y ARRAYS.
*          IOPT DETERMINES WHETHER A LINE OR BAR CHART IS PRODUCED.
*                 IOPT = 0                A DOTTED LINE
*                 IOPT = 1                A SOLID LINE
*                 IOPT = 2                A BAR CHART
*                 IOPT = 2 AND ISYM = 0   PLOT POINTS
*
      NOB=NOBS
      ISYM=ISYMBL
      IOPT=IPTN
      IN=' '
      IF(ISYM .EQ. 0 .AND. IOPT .EQ. 2)IN='X'
      IF(ISYM .EQ. 1) IN='X'
      IF(ISYM .EQ. 2) IN='O'
      IF(ISYM .EQ. 3) IN='I'
      IF(ISYM .EQ. 4) IN='*'
      IF(ISYM .EQ. 5) IN='+'
      IF(IOPT.NE.2) THEN
*
*      PLOT A LINE
*
      IDOT=1
      IF(IOPT.EQ.0)IDOT=2
      CALL MOVEA(X(1),Y(1))
      DO 10 I=2,NOB,IDOT
        CALL MOVEA(X(I-1),Y(I-1))
        CALL DRAWA(X(I),Y(I))
        IF(MOD(I,NOB/10) .NE. 0 .OR. IN.EQ.' ') GO TO 10
        CALL ANMODE
        WRITE(6,1000) IN
 10     CONTINUE
      ELSE
*
*        PLOT A BAR CHART
*
       DO 20 I=1,NOB
        CALL MOVEA(X(I),0.0)
        IF(ISYM.EQ.0) THEN
         CALL MOVEA(X(I),Y(I))
        ELSE
         CALL DRAWA(X(I),Y(I))
        END IF
        CALL ANMODE
        WRITE(6,1000) IN
 20     CONTINUE
```

```
      END IF
1000    FORMAT(1H ,'~STR/',A1,'/')
      RETURN
      END


      SUBROUTINE PLOTXY(X,Y,NOBS,HG,HX,HY,IPTN)
      DIMENSION X(NOBS),Y(NOBS),XS(1000),YS(1000)
      CHARACTER*10   HG(8),HX(8),HY(3)
*
*               SUBROUTINE TO PLOT X VERSUS Y WITH TITLES IN ONE CALL
*
*            IPTN LT 10  PLOT LINE CURVE
*                 GE 10  PLOT BAR CHART
*
        CALL ANMODE
        WRITE(6,*)' ~ERA W  ~GRA 1,35  ~SHRINK YES '
      NOB=NOBS
      IOPT=IPTN
      ISYM1=IOPT
      IF(IOPT.GE.10)ISYM1=IOPT-10
      IOPT1=1
      IF(IOPT.GE.10)IOPT1=2
        CALL SCALE(X,NOB,HX,7,XS)
        CALL SCALE(Y,NOB,HY,3,YS)
      CALL FRAME(HG,HX,HY,XS,YS,NOB)
      CALL CURVE(XS,YS,NOB,IOPT1,ISYM1)
      RETURN
      END
```

The generalized plot routines are stored in file PLOTXY.FOR . The subroutines in that file are: CURVE, FRAME, PLOTXY, SCALE, VMAX, VMIN, where FRAME, SCALE, VMAX, and VMIN are the same as the versions in the Time Series Package listed in Appendix A. The new PLOTXY and CURVE routines for this package are listed above. The user should place the same dimension and character definition statements in his program. Note that SCALE used the last 2 locations in the X and Y headings for scaling information. The user must supply the titles in HG, HX, and HY for the plot.

Appendix B.3: Program to Print PLNT.DAT file.

```
      PROGRAM TEST
      CHARACTER*10  FNAM
      CHARACTER*1   NUM(9)
      DIMENSION IPAR(6),PAR(10)
      DATA NUM/'1','2','3','4','5','6','7','8','9'/
      PRINT*,'INPUT PLNT.DAT VERSION NUMBER;'
      ACCEPT*,INUM
      FNAM='PLNT.DAT; '
      FNAM(10:10)=NUM(INUM)
      OPEN(UNIT=1,FILE=FNAM,STATUS='OLD',
     1 FORM='UNFORMATTED')
      REWIND(1)
      CALL RDBF(IPAR,1,6)
      PRINT*,'IPAR. VEC. =',(IPAR(J),J=1,6)
      DO 100 I=1,4
      IF(IPAR(I).NE.0) THEN
       CALL RDBF(PAR,1,IPAR(I))
       PRINT*,'I= ',I
       NJ=IPAR(I)
       PRINT*,'NPAR = ',NJ
       PRINT*,'PARA. VEC. =',(PAR(J),J=1,NJ)
      END IF
  100 CONTINUE
      CLOSE(UNIT=1)
      STOP
      END
      SUBROUTINE RDBF(X,NT,NX)
      DIMENSION X(NX)
      READ(NT)X
      RETURN
      END
```

Note that the version number of the file is the same as the problem number accessed by the time series program. IPAR parameters are model parameters P,Q,R,S,d,b in that order. NPAR is polynomial order in (B). Remember polynomial $\omega(B)$ also has parameter ω_0, which adds one to its NPAR value without increasing its order. To use this program with file BINARY.DAT simply change the filename.

Appendix B.4: Program to Print TSDATA.DAT file.

```
      PROGRAM TEST
      DIMENSION T(1000),U(1000),Y(1000)
      OPEN(UNIT=1,FILE='TSDATA.DAT',STATUS='OLD',
     1 FORM='UNFORMATTED')
      READ(1)NOB
      CALL RDBF(T,1,NOB)
      CALL RDBF(U,1,NOB)
      CALL RDBF(Y,1,NOB)
      I=1
      DO WHILE(I.LT.NOB)
      PRINT*,T(I),U(I),Y(I)
      I=I+1
      END DO
      CLOSE(UNIT=1)
      STOP
      END
      SUBROUTINE RDBF(X,NT,NX)
      DIMENSION X(NX)
      READ(NT)X
      RETURN
      END
```

Note that this program will send the display to the screen immediately. If specific values are desired to be viewed, control the terminal scrolling capability during the program execution.