IDPAC User's Guide

Revision 1

Wieslander, Johan

1976

Document Version:
Publisher's PDF, also known as Version of record

Link to publication

Citation for published version (APA):
Wieslander, J. (1976). *IDPAC User's Guide: Revision 1*. (Research Reports TFRT-3099). Department of Automatic Control, Lund Institute of Technology (LTH).

Total number of authors:
1

IDPAC

USER'S GUIDE

REVISION I

J. WIESLANDER

Report 7605 April 1976
Department of Automatic Control
Lund Institute of Technology

IDPAC USER's GUIDE

REVISION I.

Johan Wieslander

Lund Institute of Technology
Department of Automatic Control
April 1976

TABLE OF CONTENTS

INTRODUCTION

IDPAC is an interactive program for data analysis and system identification. It is designed to be a powerful tool in the hands of the experienced user.

Thus, talking to the beginner that reads this User's Guide for the first time: You will probably find it somewhat akward to get started using IDPAC. But have hope, it won't take very long until you too are experienced.

IDPAC interacts with the user via a graphic terminal, i e outputs diagrams and text on a screen, and accepts keyboard input in the form of *command lines*. This command method of interaction has many advantages:

a) it is consise,

b) it is predictable, i e the user can prepare himself,

c) it allows the user to improvise at any time,

d) it allows MACROs,

e) it is good for the programmer.

The drawback is that it leaves the beginner without support.

This guide will give some background on how data is organized and used by IDPAC, how the "interactive language" is constructed and interpreted and what method is used in the command routines, where applicable. It will in some cases give a few hints of how things are to be used, but it will *not* teach identification methodology. This is a science and in certain aspects an art that must be studied somewhere else. In doing so, however, IDPAC and maybe this guide will be a good companion.

This report is an extensive revision of I Gustavsson, S Selander, J Wieslander: IDPAC User's Guide, Report 7331, Oct 1973. It describes IDPAC V3A, which actually does not exist at this time (Feb 1976). It is planned for late Spring 1976 and represents an improvement of the commands STRUC, SQR and LS plus a more strict definition of system and frequency response files compared to the current V2D implementation. There are also some minor modifications in command formats. Compared to the version described in the old User's Guide, great changes have been made, eg in command decoding.

IDPAC has since 1973 been used by many people. Their experiences and suggestions have been of great help to thinkers and programmers at the institute who have made this new version of the program and its guide possible. To all there is a heartfelt thank.

Johan Wieslander

I. GENERAL DESCRIPTION OF COMMANDS

IDPAC is an interactive command-driven program. The input of a command and the subsequent decoding is done by a special set of subroutines with the collective name INTRAC. INTRAC is common to several program packages and will be fully described in a separate report. The following is a short introduction.

COMMAND MODES

Normally, a program action is initiated by the user by giving a *command*. When the program is in *normal mode* (or *command mode*), the commands are entered from the keyboard on the user's terminal. A ready sign (>) is output on the left hand margin when the program awaits a new command. In normal mode essentially any command is legal though not necessarily meaningful.

The second program mode is the *MACRO mode*. In this case the commands are read from a special macro file, thus allowing the user easy access to commonly used command sequences. A special set of commands, meaningful only in macro mode, allow looping, testing and jumps to be done.

In the macro mode, a special input command (READ) is available. This command together with the ones previously mentioned, give the possibility of a macro designed to prompt the user to take certain actions or to make certain decisions. In this way the classical "question & answer" dialogue between user and program may be realised.

The macro command mode is more fully described in section III.

Some commands may require a more detailed specification than the one given in the command line. In such a case, a *subcommand sequence* is entered. This is indicated by the ready sign appearing a few steps to the right of the lefthand margin.

In the subcommand sequence, only a restricted special set of commands depending on the main one, and the INTRAC--implemented commands (i e those in section IV), are legal. The commands in the subcommand sequence may be entered either in normal mode or in MACRO mode. The sequence is terminated by a line ending with the "alt mode" ("Escape") character.

GENERIC DESCRIPTION OF COMMANDS

A command has the generic form shown below:

```
CMND LARG1 ... LARGNL←RARG1 ... RARGNR
```

The first item is the command name or the command identifier. After that follows the argument list, separated into two parts by the left arrow (←). The left hand arguments and the right hand arguments represent results and inputs of the command respectively. In some commands one of these parts is missing, then the left arrow is also omitted.

In many commands some of the arguments are *optional*. In the detailed command description later on, this is indicated by enclosing optional arguments in brackets ([ ]). When the number of arguments is optinal, this is indicated with a series of dots (...). Mutually excluding arguments or options are separated by a slash (/). An argument may be replaced by a comma. If so, the corresponding argument in the previous command is used (short-hand feature).

Comments are preceeded by a double quote (") and may end any command line. An empty command line or one containing only a comment is legal.

ARGUMENTS

The *arguments* of a command line can be of different *types*. They may be integer numbers, real numbers, some special characters (+, -, * etc) and Hollerith strings. An argument is recognized as a Hollerith string when its first character is alphabetic, the remaining characters beeing alphanumeric. One form of a Hollerith string is a *flag*. In this case a certain value of the argument specifies a special action to be taken. In the detailed description these values are denoted by quotes (e g 'HP'). An unused flag is omitted or the string 'VOID' is used.

Other forms of Hollerith strings are *names*. The normal use of a name is as a filename. Another instance is as names of *global variables* (see next section). Within a MACRO a name may be used in two other ways: as a name of a formal argument or as a name of a local variable. One or more arguments may be enclosed in parentheses. This indicates that they are attributes to the previous argument. Examples are column numbers in a data file or section name in a system file.

GLOBAL VARIABLES

A reference to a global variable is constructed in the following way:

```
NAME.[EXT]
```

Evidently, it consists of a name followed by a dot. Optionally a second name follows as an extension.

The program package contains a table of values to be associated with these references. These values are of certain types as described in the previous section. Whenever a global variable reference is found in the command line, it is substituted by the corresponding value and type by the INTRAC routines. Thus, e g an integer argument in a command line may be replaced by any global variable reference provided that its corresponding value is of integer type.

Values are assigned to the global variables e g via the LET-command. Also some commands may deliver results to global variables, see e g the command STAT.

On the other hand, some commands make implicit use of a set of predefived (reserved) global variables. See e g the commands PLOT, ASPEC and ML.

RESERVED GLOBAL VARIABLES

In order to make some commands shorter and easier to remember, some variables or flags are left out of the command line. They are instead implemented as a set of reserved and pre-defined global variables. These are in some cases problem dependent, so they should not be regarded as "unimportant". They may at any time be changed via a LET-command.

The names, meaning and default values of the reserved global variables are listed below.

NPLX. number of points (TIME OFF) or number of seconds, minutes or hours per plot page (default 100).

NOF. number of frequency intervals for which spectra will be computed (default 100).

INIML. determines if the ML-identifier shall estimate initial values for the output (default 0).
0 no estimation
1 estimation

PRIML. print parameter for the ML-identifier (default 0).

0 no printout

1 loss function and lambda for starting values are printed as well as the final estimate with derivatives, second derivative matrix and inverse of the second derivative matrix. The final estimate is also displayed.

2 1 + the estimate for each iteration.

3 2 + derivatives, second derivative matrix and inverse of the second derivative matrix for each iteration. Each estimate is displayed.

LIML. determines if the residuals shall be limited (default 0).

0 no limitation

1 the residuals will be limited to 3λ in each iteration, i e if $\varepsilon(t) > 3\lambda$ then $\varepsilon(t) = 3\lambda$.

ITML. maximum number of iterations for the ML-identifier (default 20).

IFP. the point where a data sequence from INSI starts (default 1).

NU. starting value for random number generator. The value should be an odd integer. (starting value 9)

PRINT. print parameter for STAT, TREND and RESID (default 0).

0 no printout

1 STAT: displayed information printed

TREND: coefficients for the correction polynomial printed

RESID: mean, st dev, deg of freedom, test quantity, sqewness and kurtosis from the test of normality + displayed information printed

2 1 + (for RESID) absolute and cumulative frequencies from the test of normality printed

YMIN. minimum value for plots (default 0.0).

YMAX. maximum value for plots (default 0.0).

AMP. amplitude for data sequences from INSI (default 1.0).

DELTA. sample period for data sequences from INSI (seconds) (default 1.0).

WMIN. minimum angular frequency. Used by ASPEC, CSPEC, SPTRF (default 0.01).

WMAX. maximum angular frequency (default 100.).

II. IDPAC FILE HANDLING

IDPAC operates mainly on data stored as files on some form of random access mass memory, typically a disk. The data is referred to by a name given in the command string. IDPAC operates on four different kinds of files:
data files, system files, structure files and macro files. The three latter are implemented as symbolic (or text) files. A frequency response file is a special case of a data file.

DATA FILES

Data files are written in binary format with a so called file head of 10 integers which contain information about the contents of the file.

Data are stored in a matrix form. Each column contains one time series and each row contains one sample from each of the time series in the file. A row in the matrix corresponds to a logical record (i e a FORTRAN READ/WRITE statement).

The number of columns must not exceed 15 (implementation dependent). The number of rows is restricted only by the available disk space.

The parameters in the file head are:

Parameter	Meaning
1	number of rows (number of records)
2	number of columns (record length)
3	third dimension (time) (not used at present)
4	sample interval in ticks (20 ms)
5	date recorded mm/dd/yy
6	time recorded hh/mm
7	indicates constant record length
8	number of generating command
9	0 for a standard data file, 1 for a spectrum file
10	not used

The following conventions apply to *all commands where the output is a data file*:

- If the output file name is omitted the output will be placed where the input was fetched.
- If an output file name but no column number is given a new file is generated.
- If an output file name and a column number are given the new column must replace an old column or be placed immediately to the right of the old ones.

FREQUENCY RESPONSE FILES

A frequency response is in IDPAC stored as a number of triplets (frequency, amplitude, phase). It is required that they are stored so that the frequencies are ordered after increasing values. Frequency values are stored in one column, amplitudes in an other and phase in a third, thus giving a *frequency response file*. The phase is represented in degrees.

For simplicity, commands operating on frequency response files will automatically use columns 1, 2 and 3 for

response 1, columns 4, 5 and 6 for response 2 etc.
E g ASPEC SP(2) ← DATA 20 will compute an autospectrum and store frequency values in column 4, amplitude values in column 5 and phase values (in this example zeroes) in column 6.

A frequency response file is a special case of a data file, and most commands will not be able to distinguish the two. That implies that the user is free to operate on frequency response files with any command operating on data files, he finds useful. Some commands, however, do require some arguments to denote a frequency response file and will give an error message otherwise. Such a file is distinguished by integer 9 in the file head beeing 1 rather than 0. Thus, provided the user knows what he is doing, he can convert a frequency response file to a data file, or vice versa, simply by changing this integer through the command FHEAD.

SYSTEM FILES

A given system may be described or represented in many different ways. Therefore a system file may contain several different sections, each with a different description of the same system. Moreover, as the information may be quite different in structure in different representations a system file is symbolic and the information is grouped into sections by section headings. The information in each section is labeled with keywords. Thus a system file is easily listed on a printer, also it is easily checked, altered or generated with an editor.

A section within a system file has the following format:

```
BEGIN   NAME
SECTION   HEADING
    statements
END
```

The following conventions apply to *all commands where the output is a system file.*

- o If the file already exists a section name must be given.
- o A new section in an existing file will be placed first.
- o If duplicate section names exist, only the first section will be accessible. The others will still remain in the file and can be altered or deleted with the editor.

The section: DISCRETE MISO TRANSFER FUNCTION

This section describes a linear, discrete time, multiple input - single output dynamic system on the general form

$$y(t) = \Sigma \frac{B_i(q^{-1})}{A_i(q^{-1})} \cdot u_i(t) + \Sigma\lambda_i \cdot \frac{C_i(q^{-1})}{D_i(q^{-1})} \cdot e_i(t)$$

The normal form used in IDPAC is, however,

$$y(t) = \Sigma \frac{B_i(q^{-1})}{A(q^{-1})} \cdot u(t) + \lambda \cdot \frac{C(q^{-1})}{A(q^{-1})} \cdot e(t)$$

(Refer to the description of individual commands to see which form is used.)

The section heading must read:

DISCRETE MISO TRANSFER FUNCTION

The section *must* contain the following statements:

- o sample interval definition
- o one A-polynomial
- o one B- or C-polynomial
- o at least one λ definition if C-polynomials are present.

This section *may* contain:

- several A-, B-, C- and D-polynomials
- initial values for the output
- uncertainties of parameter estimates
- loss function value
- Akaike's test quantity
- covariance matrix of parameter estimates
- comments preceeded by a double quote (")
- blank lines (not between parts of a polynomial description)

If there are several polynomials of the same type, they must be enumerated increasingly from 1 on.

Specifics

Sample interval: SAMPLE INTERVAL t s

where t is the sample interval in seconds (integer or real)

Polynomials: XPOLYNOMIAL j

$Q\uparrow K*(c_1*Q\uparrow i_1 + c_2*Q\uparrow i_2 + \ldots + c_n*Q\uparrow i_n)$

where $X \in \{A,B,C,D\}$; $j = 1,2,..$ or omitted

$K, i_1, i_2, .. \leqq 0$ integers

- the leading power of Q and the parenthesis are optional
- the multiplication signs are optional
- c_i may be written in free format, i e 10, 10. and 1.0E1 are all treated as 10.0
- c_i is treated as $c_i*Q\uparrow 0$
- $Q\uparrow i$ is treated as $1.0*Q\uparrow i$
- terms equal to zero may be omitted
- the order between the terms is not essential

- a polynomial specification may be written over several lines, but there must not be any blank lines or comment lines in between
- there must not be two terms of the same order of Q
- the maximum order of any polynomial is 25 + the order of the time delay
- a C-polynomial definition must be followed by a lambda definition

Noise standard deviation: LAMBDA λ
where λ is the noise standard deviation (integer or real)

Loss function value: LOSS FUNCTION v
where v is the value of a loss function (integer or real)

Akaike's test quantity: AIC v
where vi is a test quantity computed by some identification routines

Initial values for the output:
INITIAL VALUES FOR THE OUTPUT
y_0 + y_{-1}*Q↑-1 + ...
cf polynomial definitions

Standard deviations of parameter estimates:
UNCERTAINTIES
Q↑K*(s_1*Q↑i_1 + s_2*Q↑i_2 + ...)
cf polynomial definitions

Covariance matrix of parameter estimates:

COVARIANCE MATRIX

c_{11} c_{12} .. c_{1m}

c_{21}

.

.

c_{n1} c_{nm}

blank line

c_{1m+1} c_{1n}

.

.

.

c_{nm+1} c_{nn}

Comments: comments must be preceeded by a double quote (")

Example: (This is a system file produced by the ML command.)

```
DISCRETE MISO TRANSFER FUNCTION

"MAXIMUM - LIKELIHOOD ESTIMATION OF ORDER 2
"FROM THE DATA FILE WRK

"INPUT(S): COLUMN(S)  1
"OUTPUT: COLUMN  2

SAMPLE INTERVAL   1.00 S

APOLYNOMIAL
          1.0000   Q↑-0 -  1.3684    Q↑-1 + 0.45712    Q↑-2

BPOLYNOMIAL
Q↑-1 *(  0.85633   Q↑-0 + 0.62352    Q↑-1 )

CPOLYNOMIAL
          1.0000   Q↑-0 + 0.00000    Q↑-1 + 0.00000    Q↑-2


LAMBDA  0.97626    +- 3.98557E-02
LOSS FUNCTION   142.96
AIC   548.95
```

Generation

System files can be generated in the editor (input mode). Example: (user-written lines are underlined)

```
>EDIT SFILE
   FILE SFILE NOT FOUND
   INPUT
   BEGIN
   DISCRETE MISO TRANSFER FUNCTION
   SAMPLE INTERVAL 1.0 S
   APOLYNOMIAL
   1 - 1.5Q↑-1 + 0.7Q↑-2
   BPOLYNOMIAL
   Q↑-1 + 0.5Q↑-2
   CPOLYNOMIAL
   1 - 0.81Q↑-1 + 0.2Q↑-2
   LAMBDA 1
   END
        (carriage return)
   EDIT
   >E
>
```

STRUCTURE FILES

Before an identification is performed the structure (i e the degree of polynomials, number of time delays etc) of the desired model must be specified. This may be a quite lengthy operation and may have to be repeated several times.

A special command (STRUC) has been designed to aquire and check this information and then pass it on to the identification routine. This is done in the form of a symbolic file, a *structure file*. In some cases this information is augmented by other commands, cf the command SQR. Note that the structure file functions as a "pre-system file".

Although it is quite possible to write or alter a structure file via the editor, the proper and safest way of doing it is via the STRUC command.

An example of a structure file is given below: (preliminary)

```
STR2
MAXIMAL VALUES
NAMAX 3
NUMAX 1
NBMAX 3
KBMAX 1
ACIUAL VALUES
NAACT 2
NUACT 1
NBACT 2
KBACT 1
SQFIL R
DATA FROM FILE WRK
COLUMNS  1  2
SAMPLE INTERVAL   1.0000
NAFIX
NBFIX
```

MACRO FILES

MACRO files are symbolic files defining a new command as a sequence of ordinary IDPAC-commands. The MACRO concept is more fully described in the following section.

III. THE MACRO FACILITY

INTRODUCTION

The MACRO facility gives the user the possibility to store a certain series of commands for later and repeated use. This could be of interest in a number of situations.

A. The same sequence of commands is in a given application used several times with only minor modifications. Storing this sequence as a MACRO effectively defines a new special purpose IDPAC command with greater efficiency as a result.

B. By the same token, entire routine data analyses can be performed by a special purpose MACRO, run by unskilled personnel.

C. For the benefit of the unexperienced user a set of argumentless commands may be designed. These would use the WRITE and READ commands to output prompting questions to the user and read his answers. When all necessary information is available, the ordinary IDPAC command is invoked.

A MACRO is implemented as a symbolic file on the mass--storage device. It may be generated either with the editor or with the command MACRO. (In either case, this command is the first one stored in the file.) Whenever a command line contains a command that is not found in the internal table of standard IDPAC commands, it is first assumed to be a MACRO name and the mass memory is searched for a file with that name. If none is found, an error message 'ILLEGAL COMMAND' is given, otherwise execution of commands from that file is started.

FORMAL AND ACTUAL ARGUMENTS

When defining a MACRO, the user often will find that he later on when using the MACRO, he will want to change individual parts of the commands, (viz the arguments of the command or even the command itself). He may do this by giving these parts distinct names and by defining these names to be *formal* arguments of the MACRO. Formal arguments are specified by being included in the MACRO--command or in the special command FORML.

When the MACRO is being called all formal arguments (not the ones specified by FORML) must correspond to an *actual* argument specified in the MACRO call. Then when executing the MACRO all formal arguments are replaced by their actual values. This is done automatically by the MACRO--handler within the command decoding complex, INTRAC.

LOCAL VARIABLES

A local variable has the same form as a formal argument and is in fact treated in very much the same fashion. It is local to the MACRO-level and must be given a value in a READ, FOR or LET command before it is used.

Global variables were discussed in section I.

GENERATION OF MACRO

A MACRO can be generated in some different ways. Naturally, being implemented as a text file, a MACRO can be defined using the text editor. Of course, no checks on the MACRO will be performed. The first command stored must be MACRO, the last one must be END.

A MACRO may also be defined using the *command* MACRO in normal mode. The following actions depend on the value of the switch EXEC ON/EXEC OFF. If the switch is OFF a file will be opened and all following commands up to and including END will be checked for formal errors and (if OK) output to the file, but they are not executed. If on the other hand the switch is ON, actual values for the formal arguments in the MACRO-command will be requested (this is also done for a FORML-command) and all subsequent commands will be checked as well as *executed* before they are stored on the MACRO file. The generation is terminated with the command END, which is also stored.

Naturally some commands, e g GOTO, IF ----, FOR - NEXT etc, will have no effect in the EXEC ON mode.

All MACROs, regardless of their way of generation, can be modified using the text editor. There is no way to check the validity of the modifications.

EXECUTING A MACRO

A MACRO is executed (or called) by giving its name instead of a command on a command line. If the MACRO contains formal arguments their corresponding actual values must follow the MACRO name on the same line. Note that if a delimiter is included in the formal argument list, it must appear in the same relative position among the actual arguments.

MACRO COMMAND ECHOING

All commands in a MACRO may be echoed on the teletype as they are executed. The commands are then preceeded by a reverse ready sign (<) to indicate that they are not user written. When a normal command or a MACRO call is echoed, formal arguments are substituted by their actual values. This doesn't apply, however, to the commands implemented within INTRAC. In these cases substitution might lead to printouts as "IF 3 GT 2 GOTO L1", hence substitution is inhibited.

The echoing may be turned on by the command

```
   TURN MACOM ON
```

and turned off by the command

```
   TURN MACOM OFF.
```

NESTING OF MACROs

A MACRO may contain calls to other MACROs including itself. The number of MACRO levels is, however, restricted to a maximum of five (implementation dependent).
To indicate the current level, the ready sign (or the reverse ready sign) is moved two steps to the right for each level.

SUSPENDING A MACRO

The execution of a MACRO may be suspended by setting data swith 0 on the computer's operator's console. The suspension takes effect when the current command is finished, and may be resumed by the command SWTCH. (This feature is implementation dependent.)

ERRONEOUS COMMANDS

Erroneous commands are not included in a MACRO being generated. Erroneous commands in a MACRO being executed causes it to be suspended and an error message to be typed on the teletype. If MACOM has been turned off, the erroneous command is also printed. The user may then write a correct command on the teletype and resume execution with a SWTCH command or deactivate the MACRO via an END command.

RESTRICTIONS

MACROs may be nested maximum 5 levels deep. The total number of formal arguments used may not exceed 50 (implementation dependent).

IV. SUMMARY OF COMMANDS IMPLEMENTED WITHIN INTRAC

IDPAC relies on a set of subroutines with the collective name INTRAC. Within these routines, some 15 commands of a more general nature are implemented. Examples are generation of a MACRO, controlling the execution of a MACRO, input and output of values for local/global variables and controlling storage of global variables.

A brief description of these commands follows:

1. MACRO NAME [FARG1 FARG2 ...]

In normal mode with EXEC OFF, this command will set a switch internal to the program that causes all subsequent commands up to END to be checked for correctness but not executed. Instead all commands up to and including END will be written onto a file with the name NAME.

In MACRO mode (i e executing a MACRO) this will be the first command encountered. Its effect will be to set up the correspondence between formal and actual arguments.

In normal mode with EXEC ON the effect will be a combination of the two above. In this case, actual values for the formal arguments will immediately be requested.

2. FORML FARG1 [FARG2 ...]

This command offers a possibility to extend the list of formal arguments beyond the one defined by MACRO- or earlier FORML-commands. This is useful when defining a

MACRO with EXEC ON.

Actual values of the arguments defined by FORML are requested when the command is executed, and are not specified in the MACRO call. Cf the command READ.

3. END

This command will end a MACRO definition as well as a MACRO execution. END used in normal mode will deactivate one level of suspended MACROs.

4. EXEC 'ON'/'OFF'

This switch controls the generation of a MACRO as detailed above. The command is illegal during a MACRO generation.

5. LABEL LNAM

This command defines LNAM as the name of the next statement.

It is meaningful only in MACRO mode.

6. GOTO LNAM

Meaningful only in MACRO mode, this statement transfers control to the statement named LNAM. If LNAM is not found, no jump is made.

7. IF ARG1 RELOP ARG2 GOTO LNAM

RELOP ∈{'EQ', 'NE', 'GE', 'GT', 'LE', 'LT'}

(If ARG1 and ARG2 are non-numeric, only 'EQ' or 'NE' are legal.)

This conditional GOTO-statement has the same function as 6 if the relation is true, if it is false no action is taken. ARG1 and ARG2 may be either a Hollerith or numeric constant, a formal argument or a local or global variable.

8. FOR COUNT = BEGIN TO FINISH [STEP INCR]

COUNT - A (local or global) variable or a formal argument
BEGIN, FINISH and INCR - Same as for COUNT or a numeric constant.

This statement, meaningful only in MACRO mode, is associated with the first 'NEXT COUNT' statement following it. Their combined effect is to provide a looping mechanism with the following properties:

a) If 'STEP INCR' is missing, INCR = 1 is assumed.

b) If INCR = 0 transfer control to command following 'NEXT'.

c) Transfer control to command following 'NEXT' *unless*: (with FORTRAN notation)

```
COUNT .LE. FINISH .AND. INCR .GT. 0
.OR.
COUNT .GE. FINISH .AND. INCR .LT. 0
```

in which case the command immediately following 'FOR' is executed.

d) When 'NEXT COUNT' is encountered INCR is added to COUNT. Go to point c).

The values of COUNT, BEGIN, FINISH and INCR may be real or integer.

9. NEXT COUNT

This statement serves as the end of a loop initiated by a 'FOR count = ---' statement. The name COUNT serves to identify the FOR - NEXT pair. The loops may be nested in standard fashion to a maximum of 5 levels.

10. SWTCH ['NRM'/'MAC']

This statement will switch between normal and MACRO mode as indicated by the argument. If the argument is missing the switch is unconditional.

Hint: This command is intended to

a) allow a user of a predefined MACRO to insert his own commands in preprepared places.

b) allow a user to continue execution of a MACRO when remedial action has been taken following an error printout.

11. LET VAR1 [=VAR2 ... VARN] = ARG1 [OP ARG2]

VAR is a formal argument or a local or global variable.
ARG is the above or a constant.
OP ∈{'+', '-', '*', '/'}

The command will evaluate a simple expression of the form above and transfer the value to the variable on the left hand side.

12. READ VAR1 TYPE1 [VAR2 TYPE2 ...]

VAR1, VAR2 ... may be a formal argument or a local or global variable. TYPE1, TYPE2 ... describes the expected type of the corresponding variable (see table below).

This command, meaningful in MACRO mode, will demand an input from the user. The type of the answer is checked against the expected type given in the command.

Hint: Use the WRITE-command to tell the user about the response that is expected from him.

Table of values for TYPE1, TYPE2 ...

Value	Meaning
NAME	Hollerith expected
INT	Integer expected
REAL	Real number expected
DELIM	Delimiter expected
NUM	Number (real or integer) expected
YESNO	'YES' or 'NO' expected

13. WRITE [([DEV][FORM])] [string1/ARG1 [string2/ARG2 ...]]

DEV ∈{'TP', 'DIS', 'LP'} standing for teleprinter, display and lineprinter.
FORM ∈{'FF', 'LF'} (Top of Form, Line Feed)
string = 'Any string of characters'.
ARG = formal argument or local or global variable.

This command will output on the device specified (default: DIS) with the form editing specified the string(s) or value of ARG(s).

When no STRING/ARG is specified the default action is to output the following information after an initial Top of Form.

a) reserved global variables

b) user-defined global variables

c) when applicable: name of executing MACRO including its local variables and formal arguments.

Hint: This command allows a MACRO to put prompting questions to the user (before the READ-command is used).

14. FREE ARG1 [ARG2 ...]

The form of ARG is:

ARG = NAM.[EXT]/NAM.*/*.NAM/*.*

The action of this command is to delete from the list of *non-reserved* global variables the global variable referenced by ARG. In the alternate forms of ARG, the asterisk (*) denotes that all global variables, regardless of first/second name is to be deleted.

E g:	ARG = ADAM.*	will delete all with ADAM as first name.
	ARG = *.*	will delete all global variables.

15. STOP

This command will stop IDPAC and return control to the monitor

EXAMPLES

A few examples will show how a MACRO may be used. (⊛ denotes the 'alt mode' ('Escape') character.)

Example 1. This example shows how a simple and straight-forward command for LS-identification can be constructed, disregarding some of the facilities that STRUC, SQR and LS offer if used separately. The call

```
      LSID ADAM←DATA 2
```

will generate a system file ADAM with a second order single-input single-output model from the data in DATA.

Example 2. The ML-command will optionally give the covariance matrix of the parameter estimates. One way of assessing the uncertainty of a model is to use the covariance matrix to compute a set of perturbed models and then compute their step or impulse responses.

The MACRO RAND in example 2 does this for NL models. The example also shows the corresponding call to ML and INSI followed by the call to RAND.

Example 3. In this case a MACRO is defined with EXEC ON, part I. Note that the actual arguments immediately are requested, and entered (NB1). After the definition the MACRO is called and echoed, because the switch MACOM is ON, part II. At NB2 the command SWTCH NRM is encountered. Thus the MACRO is suspended and normal command mode is initiated. The user gives a subcommand to ML and then SWITCHes back to MACRO-mode.

In part III the switch MACOM is OFF but an error in RESID is detected, (NB3). The offending line is echoed and the MACRO is suspended. The user writes the correct command and reactivates the MACRO with the command SWTCH.

Example 4. Here the editor is used to create a MACRO file. The MACRO is designed to ask the user a few questions concerning a filter he is creating. The idea is that the MACRO is to take the user by the hand and issue a FILT command for him.

The MACRO is input to the editor, part I. It is then invoked and the dialogue, part II, is the result.

```
>MACRO LSID MODEL←DATA NORD
>STRUC STRF
    >NA MAX NORD
    >NU MAX 1
    >NB MAX NORD ⊛
>SQR RMAT←DATA STRF
>DELET MODEL
>LS MODEL←STRF
>END
```

Example 1.

```
>MACRO RAND←MOD U NL
>FOR I = 1 TO NL
>RANPA P←MOD
>DETER Y(I)←P U
>NEXT I
>PLOT Y
>END
>
>ML (SC) MLMOD←WRK 2
    >SAVE COMAT
>LET NPLX. = 40
>INSI STEP NPLX.
    >STEP ⊛
>RAND YSTE←MLMOD STEP 5
```

Example 2.

```
>EXEC ON
>TURN MACOM ON
>MACRO MLTST MODEL RES MERR←DATA(COL1 COL2)NO NOL SWARG        ⎫
      #ML1 RES1 ERR1←WRK(1 3) 1 10 MAC      ←——NB1             ⎪
   >ML (SC) MODEL←DATA(COL1 COL2) NO                           ⎪
      >SWTCH SWARG                                             ⎪
      >⊗                                                       ⎪
   >RESID RES←MODEL DATA (COL1 COL2) NOL                       ⎬ I
   >DETER MERR←MODEL DATA(COL1)                                ⎪
   >VECOP MERR←DATA(COL2)- MERR                                ⎪
   >PLOT MERR DATA(COL2)                                       ⎪
   >END                                                        ⎭
>
>MLTST ML2 RES2 ERR2←WRK(1 3) 2 10 NRM                         ⎫
   <MACRO MLTST MODEL RES MERR←DATA(COL1 COL2) NO NOL SWARG    ⎪
   <ML (SC) ML2←WRK(1 3) 2                                     ⎪
      <SWTCH NRM                                               ⎪
      >SAVE STDEV                          ←——NB2              ⎪
      >SWTCH                                                   ⎬ II
      <                                                        ⎪
   <RESID RES2←ML2 WRK(1 3) 10                                 ⎪
   <DETER ERR2←ML2 WRK(1)                                      ⎪
   <VECOP ERR2←WRK(3) - ERR2                                   ⎪
   <PLOT ERR2 WRK(3)                                           ⎪
   <END                                                        ⎭
>
>TURN MACOM OFF
>
>MLTST ML3 RES3 ERR3←WRK(1 3) 3 30 MAC                         ⎫
<RESID RES3←ML3 WRK(1 3) 30                ←——NB3              ⎪
TOO MANY LAGS                                                  ⎬ III
   >RESID RES3←ML3 WRK(1 3) 10                                 ⎪
   >SWTCH                                                      ⎭
>
```

Example 3.

```
>EDIT FILTR
   NOT FOUND: FILTR
   INPUT
MACRO FILTR
WRITE (TP) 'ENTER NAME OF FILTER'
READ FILTN NAME
WRITE (TP) 'WHAT TYPE (HP,BP,LP) IS ' FILTN '?'
READ FTYP NAME
WRITE (TP) 'WHAT FILTER ORDER?'
READ N INT
IF FTYP EQ BP GOTO L1
WRITE (TP) 'ENTER CUT-OFF FREQUENCY (RAD/S)'                              I
READ CF REAL
FILT FILTN←FTYP N DELTA. CF
GOTO L3
LABEL L1
WRITE (TP) 'ENTER LOW AND HIGH CUT-OFF FREQUENCIES (RAD/S)'
READ LCF REAL HCF REAL
IF LCF LT HCF GOTO L2
FILT FILTN←FTYP N DELTA. LCF HCF
LABEL L3
END
ʅ
   EDIT
   >E
>FILTR
ENTER NAME OF FILTER
TEST
WHAT TYPE (HP,BP,LP) IS TEST?
LP                                                                        II
WHAT FILTER ORDER?
2
ENTER CUT-OFF FREQUENCY (RAD/S)?
3.
>
```

<u>Example 4</u>.

V. COMMANDS AVAILABLE

The following is a structured list of the commands available (Feb 1976), together with a short indication of their functions.

1. INPUT & OUTPUT

CONV	conversion of data from symbolic form to IDPAC-standard binary representation
EDIT	keyboard input or editing of symbolic files eg MACRO- or system files
MOVE	moving files between disk and other storage media
LIST	output of files on printer or display

2. DISPLAY

PLOT	drawing diagrams from data files on display
BODE	drawing frequency response curves on display
PLMAG	drawing a magnified plot of a part of a data file, allowing modification of individual data points
FHEAD	display the file head parameters and allow the user to modify them

3. DATA OPERATIONS

INSI	generate a data file
CUT	extract a part of a data file
CONC	concatenate two data files
PICK	pick out equidistant points
SLIDE	delay signals relative to one another
STAT	compute some scalar values (sum, mean, etc)
SCLOP	do scalar operations (+, -, *, /)
VECOP	do vector operations (element by element) (+, -, *, /)
TREND	remove a trend
ACOF	compute autocorrelation
CCOF	compute crosscorrelation

4. FREQUENCY RESPONSE

FROP	do (+, -, *, /) on frequency responses
ASPEC	compute auto spectrum
CSPEC	compute cross spectrum
SPTRF	compute frequency response of a transfer function

5. SIMULATION & MODEL ANALYSIS

FILT	compute a filter
DSIM	simulate (with noise)
DETER	simulate (deterministic)
RESID	compute residuals with statistic tests
RANPA	compute model with random parameter

6. IDENTIFICATION

ML	Maximum-Likelihood identification
STRUC	define least-squares structure
SQR	perform least squares data reduction
LS	compute least squares parameter estimates

7. (Reserved for later use)

8. (Reserved for later use)

9. OTHER

DELETE	delete a file
TURN	change internal switches

CONV

Purpose

To read a source file (ASCII-code) on mass storage in free format and convert it to an IDPAC - standard binary data-file.

Command

```
CONV DATA ← SFIL (C1 C2 ...) NCOL
```

Arguments

DATA	output file name
SFIL	name of source file
C1,C2 ...	the columns of SFIL that are converted
NCOL	the number of items in SFIL that are to be considered a row

Function

The source (symbolic) file is read in free format, i e strings of characters from the set { 0 1 2 3 4 5 6 7 8 9 + - . E } are converted to real numbers following normal rules. All other characters including carriage return, tab, etc, are treated as delimiters. Thus, the use of formating characters (i e space, tab, CR-LF, etc) is unrestricted. However, the number of data items corresponding to a sampling instant must be constant, specified by NCOL. While there is no limit on NCOL, the number of columns that can be converted at a given instant may be limited and installation dependent.

EDIT

Purpose

To edit, i e create or make changes to, a symbolic (text) file. In IDPAC examples are MACRO-files, system files and symbolic data files from outside, to be converted by CONV.

Command

```
EDIT TFILE
```

Arguments

TFILE	name of symbolic (text) file

Function

The editor works in one of two modes, EDIT-mode and INPUT--mode. In EDIT-mode, the editor will read the text-file from disk back to disk, line by line. At any time, one line is the 'current line', (i e the line about to be written back onto the disk). A number of subcommands are available to control the position of the 'current line' within the text-file, or to modify the 'current line'

In INPUT-mode, lines typed on the keyboard are made the new 'current line', thus forcing the old one to be written onto the disk.

The initial mode of the editor is INPUT if the specified file is not found, otherwise EDIT.

Subcommands

Generals: n — denotes a positive integer, default 1.

/ — denotes any character not included in 'string'.

string — denotes any sequence of printing characters including space.

A string — the string is appended to the current line.

B — the bottom line of the file is made the new current line.

C /string1/string2/ — string 1 in the current line is changed to string 2.

D [n] — n lines are deleted starting with the current line.

E — exit, i e close the file and return.

F string — find the first line after the current line starting with string and make it current.

I string — insert string as the new current line after the old one.

L string — locate the first line after the current line containing string and make it current.

N [n] — make the n:th next line current.

O [n] — overlag the n next lines including the current with keyboard INPUT.

P [n] — print n lines starting with the current line. The last line printed is the new current line.

R string — replace the current line with string.

T — go to the top of the file.

TV 'ON'/'OFF' — enable/disable output on TV (display).

MOVE

Purpose

To move files between different kinds of mass storage and/or rearrange the columns of a data file.

Command

```
MOVE DEV1 [FILE1[(C11 .. C1N)]]['ND']]←DEV2 FILE2[(C21 .. C2N)]
```

Arguments

DEV1 output device ('DK', 'DT' or 'PP')

FILE1 output file name (default FILE2)

C11 .. column numbers in output file (default 1,2,2,..)

'ND' indicates that the old data in columns C11 through C1N of FILE1 shall not be deleted but moved to the right

DEV2 input device ('DK', 'DT' or 'PR')

FILE2 input file name

C21 .. column numbers in input file (default 1,2,3,..)

Function

The columns C21,.. in the data file FILE2 on device DEV2 are moved to the columns C11,.. in the data file FILE1 on device DEV1.

Symbolic (text) files, e g MACRO files, system files etc, are merely copied.

Paper tape format

Input: One line with 10 integers (the file head, see page 2.2). Free format. MOVE uses integer 1,2,3 and 7. One line for each row in the matrix. Free format. Integers are converted to real numbers.

Output: File head in format 10I7.
Data in format 6G13.5. Note that files with more than 6 columns cannot be punched.

Cautions, restrictions

- o DT is dectape unit 2.
- o Column numbers cannot be used for system- and MACRO files.
- o Data files may contain up to 20 columns as input files and up to 15 columns as output files.
- o Files cannot be moved from dectape to dectape. Columns in a data file on dectape cannot be updated, i e the command `MOVE DT FILE1(2 4 6)←DK FILE2(1 2 3)` is illegal.

Examples

```
>MOVE DK←DT DATA
>MOVE DK WORK←DK DATA(2 5 3)
>MOVE DK WORK(1 3) ND←DK DATA(4 1)
```

See the results below.

```
DATA

      11.0000        21.0000        31.0000        41.0000        51.0000
      12.0000        22.0000        32.0000        42.0000        52.0000
      13.0000        23.0000        33.0000        43.0000        53.0000
      14.0000        24.0000        34.0000        44.0000        54.0000
      15.0000        25.0000        35.0000        45.0000        55.0000
      16.0000        26.0000        36.0000        46.0000        56.0000
      17.0000        27.0000        37.0000        47.0000        57.0000
      18.0000        28.0000        38.0000        48.0000        58.0000
      19.0000        29.0000        39.0000        49.0000        59.0000
      20.0000        30.0000        40.0000        50.0000        60.0000

WRK

      21.0000        51.0000        31.0000
      22.0000        52.0000        32.0000
      23.0000        53.0000        33.0000
      24.0000        54.0000        34.0000
      25.0000        55.0000        35.0000
      26.0000        56.0000        36.0000
      27.0000        57.0000        37.0000
      28.0000        58.0000        38.0000
      29.0000        59.0000        39.0000
      30.0000        60.0000        40.0000

WRK

      41.0000        21.0000        11.0000        51.0000        31.0000
      42.0000        22.0000        12.0000        52.0000        32.0000
      43.0000        23.0000        13.0000        53.0000        33.0000
      44.0000        24.0000        14.0000        54.0000        34.0000
      45.0000        25.0000        15.0000        55.0000        35.0000
      46.0000        26.0000        16.0000        56.0000        36.0000
      47.0000        27.0000        17.0000        57.0000        37.0000
      48.0000        28.0000        18.0000        58.0000        38.0000
      49.0000        29.0000        19.0000        59.0000        39.0000
      50.0000        30.0000        20.0000        60.0000        40.0000
```

LIST

Purpose

To output a file on a lineprinter, teleprinter or display.

Command

```
LIST [([DEV]['DATA'])] DNAME [(C1 C2 ...)] [IF NUM]
or
LIST ([DEV] 'TEXT') TNAME [(NAME)]
```

Arguments

DEV	∈{'DIS', 'LP', 'TP'} output device, indicating display, line printer or teleprinter. Default is 'DIS'.
'DATA'	indicates DNAME is a data file
DNAME	data file name
C1, C2	column numbers
IF	first row to be listed
NUM	number of rows to be listed
'TEXT'	indicates that TNAME is a text file
TNAME	name of text file
NAME	name of section in a system file

Function

Data files: If more than 15 rows are to be printed (displayed) format 8G13.5 (6G13.5) is used. Else the data is printed as matrix blocks with NUM lines containing the first 8(6) columns, a blank line, NUM lines containing the next 8(6) columns etc.

Note, frequency response files are special cases of data files.

Text files: The file is directly copied onto the output medium. Text files are:

a) any file created or manipulated by the EDIT command,

b) MACRO files,

c) system files,

d) structure files.

If a section name is given for a system file, only that section is output.

Examples

```
>LIST  DATA
>LIST (LP) DATA(3 4 6) 20 10
>LIST (TEXT) MAC
>LIST (LP TEXT) SYST(NAME)
```

PLOT

Purpose

To plot data vectors on display.

Command

```
PLOT [NP] [DATAX[(C1)]←] [OPT] DATA1 [(C11 ..)] [[OPT]
        DATA2[(C22 ..)]..] [YMIN YMAX]
```

Arguments

NP	number of points or time units per page (default is the reserved global variable NPLX.)
DATAX	optional file containing x-values if plotting versus time not wanted
C1	column number of DATAX file
OPT	option specifying plot mode: 'LI': default. Linear interpolation 'HP': histogram plot 'NL': mark data points with no lines between
DATA1,2 ..	data file names (y-values)
C11, ..	column numbers in data files
YMIN	minimum value on vertical axis
YMAX	maximum value on vertical axis

Function

The indicated column(s) in the data file(s) are plotted on display (regardless of the 'DIS OFF' switch).

NP or NPLX.* rows are plotted per 'page'. A new page is plotted when a rub-out is recieved from the teletype. An alt mode terminates the plotting.
If YMIN* and YMAX* are omitted the vertical scale is determined by the values of the reserved variables YMIN. and YMAX. . If these are equal the program will choose appropriate scales. The curves are marked with integers representing the order of the corresponding file in the PLOT-command.
The marks are omitted if only one curve is plotted.

*

NPLX., YMIN. and YMAX. are global variables given default values at program startup. They may be altered by the LET-command.

Cautions, restrictions

The plotting software will choose a scaling such that the indicated values will safely fall within the screen and such that the scale indices will be in accordance with accepted standards while the axes are divided into an integer number of centimetres (on the hard copy).

Hint

Note that the command line is written on the display above the plot and that comments can be added to the command line (after a double quote (")). The command WRITE(DIS) can also be used to write comments on the display.

Examples

```
>PLOT FILE
>PLOT FILE(1) HP FILE(2) NL FILE(3)
>PLOT FILE -5.5.
```

BODE

Purpose

To plot frequency response files in a Bode-format.

Command

```
BODE [('ONE'/'TWO')] FRF1 [(F11 F12 ...)][FRF2[(F21 F22 ...)]]...
```

Arguments

'ONE'/'TWO'	switch indicating that amplitude and phase is to be plotted: in separate diagrams ('ONE') or together in one diagram ('TWO'). Default is 'TWO'.
FRF1, FRF2 ...	spectrum file names.
F11, F12	spectrum indices within resp files.

Function

The indicated (default all) curves of the frequency response file(s) FRF1 etc are plotted versus angular frequency. The abscissa is a logarithmic axis, while the ordinate is logarithmic for the amplitude and linear for the phase. If all phases are identically zero, as for auto spectrums, the phase plot is omitted.

Amplitude values smaller than 1.E-5 * (largest value) are replaced by the lower limit.

The curves are marked with integers representing the order of the corresponding file in the command.

Hint

Note that the command line is written on the display above the plot and that comments can be added to the command line (after a double quote (")). The command WRITE(DIS) can also be used to write comments on the display.

PLMAG

Purpose

To plot parts of a data vector on display and enable the user to alter data values.

Command

```
PLMAG DNAME[(C)]
```

Function

The Cth (default 1st) column in the data file DNAME may be plotted and data values altered using the following subcommands:

B[LOCK] NB — defines the number of data to be plotted per page (default 50, max 250).

P[LBEG] NR — NB points are plotted from the NRth on.

A[LTER] NR [NUM] — alters the value of NUM (default 1) points starting with point number NR.
If NUM = 1 or missing the old value of the data point is printed on the teletype followed by a value sign (¤) and a new value may be entered.
If no value is entered the old value is retained.
The program writes the next (previous) point if a > (<) - sign is entered.
If NUM > 1 a value sign appears on the teletype and new values may be entered on one or more lines.
The current block is replotted when the altering is finished.

D[ELET] NR Point number NR is deleted. NB, the deletion takes effect when leaving PLMAG. Note also that the entire row NR is deleted and that the number of points in the file is changed.

N[EXT] the next NB points are plotted.

Control is returned to the main program when an alt mode is received.

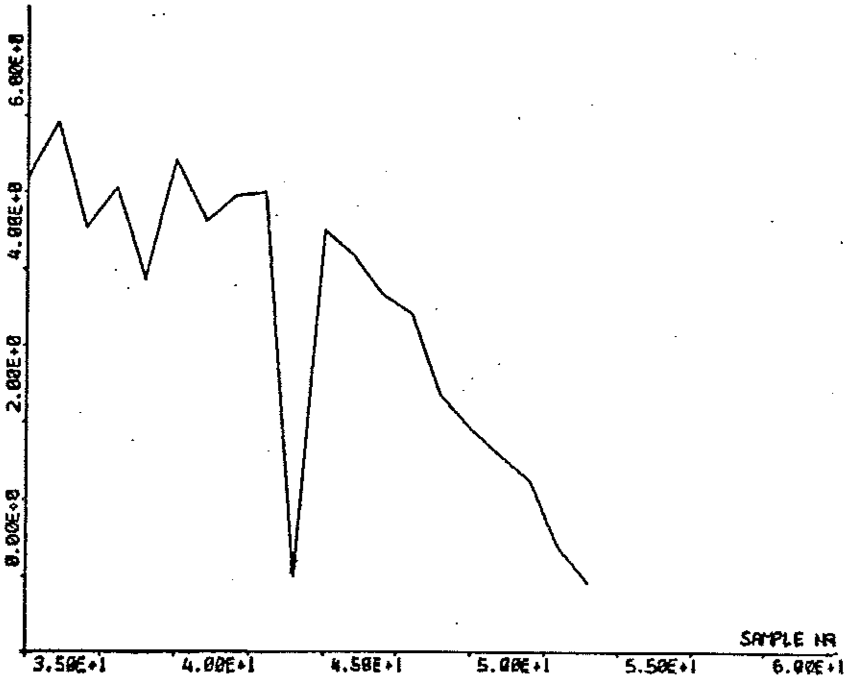
Cautions, restrictions

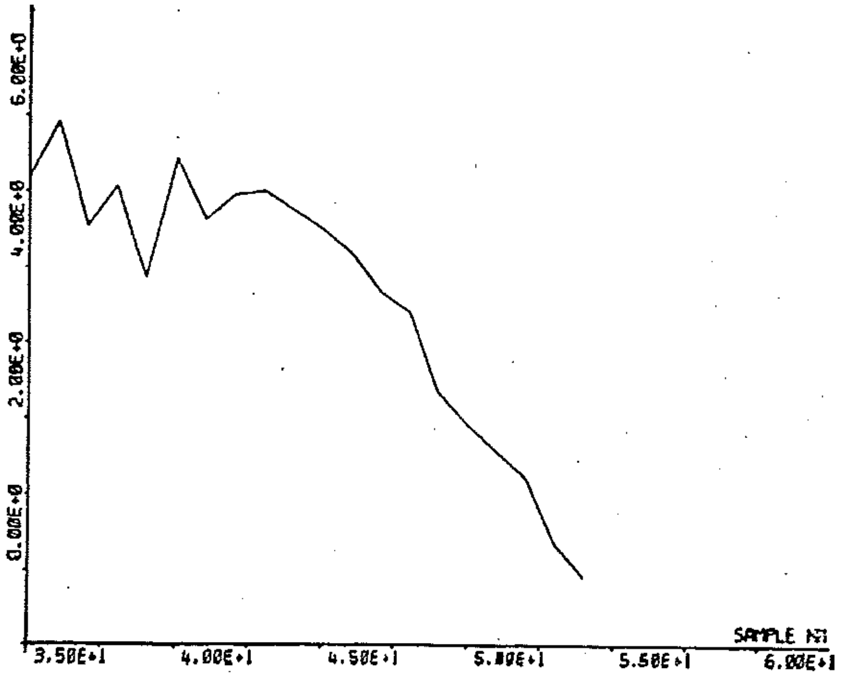
A data point must be plotted before it can be altered.

Example

(See figures below.)

```
>PLMAG FILE(2)
      >B 20
      >P 35
      >A 43
       43 OLD VALUE    5.00000    □>
       44 OLD VALUE    0.00000    □4.75
      >
```


6.00E+0
4.00E+0
2.00E+0
0.00E+0
SAMPLE NR
3.50E+1
4.00E+1
4.50E+1
5.00E+1
5.50E+1
6.00E+1


6.00E+0
4.00E+0
2.00E+0
0.00E+0
SAMPLE NR
3.50E+1
4.00E+1
4.50E+1
5.00E+1
5.50E+1
6.00E+1

FHEAD

Purpose

To display file head parameters of data files and enable the user to change them.

Command

```
FHEAD DATA
```

Function

The file head parameters of the file DATA are written on display with some explanations. The value of parameter i may be altered by the subcommand I VALUE.
Control is returned to the main program when a line is terminated by an alt mode.

Cautions, restrictions

The first and second parameters (i e the number of rows and the number of columns) must not be increased. Note that the file is permanently shortened if the first parameter is changed.

Hint

Use CUT if you want to use only a part of a file.

Example

```
>FHEAD FILE
    >5 070973
    >6 0930
```

INSI

Purpose

To generate data sequences of different types for use as inputs.

Command

```
INSI DATA[(C)] NP
```

Function

A sequence of NP data is generated and placed in the Cth (default 1st) column of the data file DATA.
Type of sequence and the parameters needed for that type are entered as a subcommand. If no parameters are entered default values are used.
If more than one subcommand is given the sequences will be written into successive columns.
The point where the sequence starts, starting value for the random number generator, amplitude and sample period are fetched from the global variables IFP., NU., AMP. and DELTA. resp. These variables are displayed and may be altered before the sequence type is entered via an ordinary LET-command. The subcommand sequence is terminated with an alt mode.

Subcommands:
(default values of the parameters within parenthesis)

```
PRBS [IBP [NBIT ISTART [KNEP]]]
```

IBP - basic period (1)

NBIT - number of bits in the shift register, min 3, max 17 (7)

ISTART - specifies starting point in the sequence
1, 2, 3 or 4 (1)

KNEP - FOA-trick is used (no KNEP)

A pseudo-random binary sequence is generated with the specified characteristics. The FOA-trick implies that the output is the output of a flip-flop that complements when the primary PRBS yields a positive value. Thus the new sequence has asymptotically zero mean value, which the standard PRBS has not.

NORM [MEAN SIGMA]

MEAN - mean value (0.0)

SIGMA - standard deviation (1.0)

A normally distributed (gaussian) random signal with specified mean and sigma is generated.

RECT [A B]

A - lower boundary (0.0)

B - upper boundary (1.0)

A rectangularly distributed random signal is generated (controlled by NU.).

SRTW [PS]

PS - change-of-sign probability (0.5)

A piecewise constant (=AMP.) signal with a given change-of-sign probability is generated.
(SRTW - Sequential Random Telegraph Wave)
(controlled by NU.).

The following are self explanatory.

SINE [OMEGA FI]

OMEGA - angular frequency (1.0 rad/s)

FI - phase (0.0 rad)

ZERO

```
STEP

RAMP [A B]

    A      - constant term (0.0)
    B      - linear term (1.0)

PULSE [LENGTH]

    LENGTH - pulse length (1)
```

Cautions, restrictions

Maximum number of points is 3000.

The global variable NU., giving starting values to the random number generator, is updated after each use. Using the command LET, the value of NU. may be initialized or saved.

Example

```
>INSI FIL 200
            >AMP 2.5
            >PRBS
```

CUT

Purpose

To cut out a part of a data file.

Command

```
CUT [DATA1]←DATA2 IF IL
```

Arguments

DATA1	output file name (default DATA2)
DATA2	input file name
IF	first row in DATA2 to be saved
IL	last row in DATA2 to be saved

Function

The rows IF through IL in the data file DATA2 are moved to the data file DATA1.

Example

```
>CUT SHORT←FILE 2 7

FILE
     11.0000        21.0000        31.0000
     12.0000        22.0000        32.0000
     13.0000        23.0000        33.0000
     14.0000        24.0000        34.0000
     15.0000        25.0000        35.0000
     16.0000        26.0000        36.0000
     17.0000        27.0000        37.0000
     18.0000        28.0000        38.0000
     19.0000        29.0000        39.0000
     20.0000        30.0000        40.0000

SHORT
     12.0000        22.0000        32.0000
     13.0000        23.0000        33.0000
     14.0000        24.0000        34.0000
     15.0000        25.0000        35.0000
     16.0000        26.0000        36.0000
     17.0000        27.0000        37.0000
```

CONC

Purpose

To concatenate two data files.

Command

```
CONC DATA1←DATA2 DATA3
```

Function

The data files DATA2 and DATA3 are concatenated giving the data file DATA1.

PICK

Purpose

To pick out equidistant samples from a data file.

Command

```
PICK DATA1←DATA2 N
```

Function

Each Nth sample in the data file DATA2 is transferred to the data file DATA1.

SLIDE

Purpose

To move the columns of a data file along each other.

Command

```
SLIDE [DATAO]←DATAI K1 K2 ..
```

Arguments

DATAO	output data file name (default DATAI)
DATAI	input data file name
K1,..	the number of steps each column in DATAI will be moved upwards (Ki>0) or downwards (Ki<0). ('Upwards' corresponds to 'to the left' in a plot.)

Function

Row i in the output file will consist of element i + k1, i + k2, i + k3 ... from respective column, where kj is Kj-min(Km). Thus the operation q^{kj} is performed on each column in the file, where q is the forward shift operator.

Cautions, restrictions

The maximum difference between Ki and Kj must not exceed 175. There must be a K for each column in the file.

Hints

SLIDE may be useful when data files are prepared for ML identification if the input directly influences the output or there is a pure time delay in the process. In the first case you must slide the input one step backwards. Differentiation can be performed in the following way: Make a file with two identical columns and slide then one (or more) steps. Then use VECOP to subtract them.

Example

```
>SLIDE ←FILE -1 0 2

FILE

    11.0000        21.0000        31.0000
    12.0000        22.0000        32.0000
    13.0000        23.0000        33.0000
    14.0000        24.0000        34.0000
    15.0000        25.0000        35.0000
    16.0000        26.0000        36.0000
    17.0000        27.0000        37.0000
    18.0000        28.0000        38.0000
    19.0000        29.0000        39.0000
    20.0000        30.0000        40.0000

FILE

    11.0000        22.0000        34.0000
    12.0000        23.0000        35.0000
    13.0000        24.0000        36.0000
    14.0000        25.0000        37.0000
    15.0000        26.0000        38.0000
    16.0000        27.0000        39.0000
    17.0000        28.0000        40.0000
```

STAT

Purpose

To compute some statistical properties of a data vector.

Command

```
STAT DATA[(C)]  [EXT]
```

Function

The sum, mean, variance, standard deviation, minimum and maximum for the Cth (default 1st) column in the data file DATA are computed and displayed. The results will also be printed on line printer if the reserved variable PRINT is nonzero.

Those of the global variables SUM.EXT, MEAN.EXT, VAR.EXT, STDEV.EXT, MIN.EXT and MAX.EXT that are previously defined as real variables will receive the appropriate value, provided that EXT is specified in the command.

Example

```
>STAT UT(2)
                UT   ( 2)

               SUM      =  -216.508
               MEAN     =  -2.16508
               VARIANCE =   424.615
               ST.DEV.  =   20.6062
               MINIMUM  =  -43.0363
               MAXIMUM  =   48.3345
```

SCLOP

Purpose

To perform scalar operations on a data vector.

Command

```
SCLOP [DATA1[(C1)]←DATA2[(C2)]  OP   CONST
```

Arguments

DATA1	output file name (default DATA2)
C1	column number in output file (default 1 or C2)
DATA2	input file name
C2	column number in input file (default 1)
OP	∈{'+', '-', '*', '/'} desired operation
CONST	constant

Function

Each element in column C2 of DATA2 is added, subtracted, multiplied or divided by CONST. The resulting data vector is placed in column C1 of DATA1.

Hint

To subtract the mean value of a data vector use STAT to compute the mean and the SCLOP to subtract it. This can also be done as a 0th order trend correction by TREND.

VECOP

Purpose

To add, subtract, multiply or divide two data vectors element by element.

Command

```
VECOP [DATA1[(C1)]]←DATA2[(C2)] OP DATA3[(C3)]
```

Arguments

DATA1	output file name (default DATA2)
DATA2,3	input file names
C1,2,3	column numbers (default 1)
OP	∈{'+', '-', '*', '/'} desired operation

Function

Each element in column C2 of DATA2 is added, subtracted, multiplied or divided by the corresponding element in column C3 of DATA3. The resulting data vector is placed in column C1 of DATA1.

TREND

Purpose

To estimate and remove a polynomial trend from a part of a data vector.

Command

```
TREND [DATA1[(C1)]←DATA2[(C2)] NO [IF IL]
```

Arguments

DATA1	output file name (default DATA2)
C1	column number in output file (default 1 or C2)
DATA2	input file name
C2	column number in input file (default 1)
NO	polynomial order
IF	first row to be corrected (default 1st)
IL	last row to be corrected (default last in the file)

Function

A polynomial trend of order NO is estimated for the C2th column in the data file DATA2 between the IFth and ILth points. The trend is substracted and the result is placed in the C1th column of the data file DATA1.

If the reserved variable PRINT. is nonzero the parameters (with reference to the left end-point of the interval) will be printed on line printer.

Method

A least squares technique is used where the parameters are estimated with reference to the midpoint of the interval.

Reference

Otnes & Enochson, Digital Time Series Analysis, Wiley, 1972.

Cautions, restrictions

The order of the polynomial must be between 0 and 3.

ACOF

Purpose

To compute autocorrelations of a data vector.

Command

```
ACOF ACF[(C1)]←DATA[(C2)] NOL
```

Arguments

ACF file name for autocorrelations

DATA data file name

C1,2 column numbers (default 1)

NOL number of lags for which the autocorrelation function shall be computed

Function

The autocorrelations of DATA(C2) are computed for 0 through NOL lags and written into ACF(C1).

Method

The autocovariances are computed using

$$R_{xx}(\tau) = \frac{1}{N} \sum_{j=\tau+1}^{N} (x_j - m)(x_{j-\tau} - m) \qquad \tau = 0, 1, .., \text{NOL}$$

where N is the number of data in DNAM2(C2)

x_j is the jth point in DNAM2(C2)

m is the mean value of the data in DNAM2(C2)

Then the autocorrelations are computed from

$$r_{xx}(\tau) = \frac{R_{xx}(\tau)}{R_{xx}(0)} \qquad \tau = \text{NOL}, .., 0$$

Cautions, restrictions

Maximum number of lags is 500.

CCOF

Purpose

To compute cross correlations between two data vectors.

Command

```
CCOF CCF[(C1)]←DATA  (C21 C22) NOL
or
CCOF CCF[(C1)]←DATA1[(C2)] DATA2[(C3)] NOL
```

Arguments

CCF	file name for cross correlations
C1	column number in output file (default 1)
DATA,DATA1,DATA2	input data file names
C21,C22	column numbers in DATA
C2,3	column numbers in DATA2,3 (default 1,1)
NOL	maximum lag for which the cross covariance function shall be computed

Function

The cross correlation function between DATA(C21) and DATA(C22) or between DATA1(C2) and DATA2(C3) is computed for 0 through NOL lags and written into CCF(C1).

Method

The cross covariances are computed using

$$R_{xy}(\tau) = \frac{1}{N} \sum_{j=\max(1,1-\tau)}^{\max(N-\tau,N)} (x_j-m_x)(y_{j+\tau}-m_y) \qquad \tau = -NOL,\dots,NOL$$

where N is the number of data in each input vector

x_j, y_j are the jth data values

m_x, m_y are the mean values of the data in the input vectors.

Then the cross correlations are computed as

$$r_{xy}(\tau) = \frac{R_{xy}(\tau)}{\sqrt{R_{xx}(0) \cdot R_{yy}(0)}}$$

Cautions, restrictions

The maximum number of lags is 500.

Hint

If y is the output of a system for a white noise input u then r_{uy} is the impulse response.

FROP

Purpose

To add, subtract, multiply or divide two frequency response files element by element.

Command

```
FROP [FRF1[(F1)]]←FRF2[(F2)] OP FRF3[(F3)]
```

Arguments

FRF1	output frequency response file (default FRF2)
FRF2,3	input frequency response files
F1,2,3	response numbers
OP	∈{'+', '-', '*', '/'} desired operation

Function

The two frequency response files FRF2 and FRF3 are read, and when two frequency points coincide, the desired operation is performed. The operands are taken as complex numbers and the result, still in amplitude and phase format (i e $z = r \cdot e^{i\varphi}$) is output to FRF1. An effort is made to keep the phase continuous across the 360° boundaries.

Hints

Dividing the cross spectrum φ_{yu} by the autospectrum φ_{u} gives the frequency characteristics of the transfer function H(s), i e $H(j\omega) = \varphi_{yu}(\omega)/\varphi_{u}(\omega)$.

ASPEC

Purpose

To compute the autospectrum of a data vector.

Command

```
ASPEC FRF[(F)]←DATA[(C)] NOL [FREQ]
```

Arguments

FRF name of frequency response file receiving autospectrum

F response number (default 1)

DATA data file name

C column number (default 1)

NOL the spectral estimation shall be based on the autocovariance function up to NOL lags

FREQ file with frequency points

Function

The autospectrum of DATA(C) is computed for NOF. frequency intervals using the autocovariances for up to NOL lags and written into FRF(F). NOF. is a global variable that can be changed by the command LET.
A Tukey window is used for smoothing.

The frequency points are choosen in one of two ways. (NPOI is the number of data points in DATA and T is the sample interval.)

a) NOF. points between Max(WMIN., 2π/NPOI*T) and

Min(WMAX., π/T). WMIN. and WMAX. are two global variables.

b) If the file FREQ has been specified, the frequency points are read from (column 1 of) this file. Still there is a check for the 2π/NPOI·T and π/T limits.

Method

The autocovariances are computed as described for the command ACOF. Then the autospectrum is computed from

$$\Phi(\omega) = 2\Delta T\left\{R_{xx}(0) + \sum_{\tau=1}^{NOL} R_{xx}(\tau)\cos(\omega\Delta T\tau)\left[1 + \cos\left(\frac{\pi\tau}{NOL+1}\right)\right]\right\}$$

where ω is the frequency in rad/s
ΔT is the sample period in s
$R_{xx}(\tau)$ is the autocovariance for lag τ.

The bandwidth $B = 4/\left(3(NOL+1)\Delta T\right)$ and the degrees of freedom $D = 8N/3(NOL+1)$.

Cautions, restrictions

Data files with zero sample period are not accepted as input files.

Maximum number of lags and maximum number of frequencies are 500.

CSPEC

Purpose

To compute amplitude and phase of the cross spectrum between two data vectors.

Command

```
CSPEC FRF[(F)] ←DATA(C11 C12) NOL [IALIGN] [FREQ]
or
CSPEC FRF[(F)] ←DATA1[(C1)] DATA2[(C2)] NOL [IALIGN] [FREQ]
```

Arguments

FRF	frequency response file
F	response number (default 1)
DATAi	input data file names
C11,C12	column numbers in DATA
C1,C2	column numbers in DATA1,2 (default 1,1)
NOL	the spectral estimation shall be based on the cross covariance function up to NOL lags
IALIGN	the numbers of lags necessary to align the two data vectors so that the largest cross covariance is centered at zero (default 0)
FREQ	file with frequency points

Function

The amplitude and phase of the cross spectrum between DATA(C11) and DATA(C12) or between DATA1(C1) and DATA2(C2) are computed for NOF. frequencies using the cross covariances up to NOL lags and written into FRF(S). NOF. is a global variable that can be changed by the command LET.

A Tukey window is used for smoothing.
Frequency points are choosen as in the command ASPEC.

Method

The cross covariances are computed as described for the command CCOF. Then the co- and quadrature spectra are computed from

$$\mathrm{EV}(k) = 0.5\left(R_{xy}(k+\mathrm{IALIGN}) + R_{xy}(-k+\mathrm{IALIGN})\right)$$

$$\mathrm{OD}(k) = 0.5\left(R_{xy}(k+\mathrm{IALIGN}) - R_{xy}(-k+\mathrm{IALIGN})\right)$$

$$k = 0, .., \mathrm{NOL}-|\mathrm{IALIGN}|$$

$$=\mathrm{NOL}'$$

$$\mathrm{CS}(\omega) = 4\Delta T\left\{\mathrm{EV}(0) + 0.5 \sum_{\tau=1}^{\mathrm{NOL}'} \mathrm{EV}(\tau)\cos(\omega\Delta T\tau)\left(1 + \cos\frac{\pi\tau}{\mathrm{NOL}'+1}\right)\right\}$$

$$\mathrm{QS}(\omega) = 2\ T \sum_{\tau=1}^{\mathrm{NOL}'} \mathrm{OD}(\tau)\sin(\omega\Delta T\tau)\left(1 + \cos\frac{\pi\tau}{\mathrm{NOL}'+1}\right)$$

Then the amplitude and phase are computed from

$$\mathrm{AMP}(\omega) = \sqrt{\left(\mathrm{CS}(\omega)^2 + \mathrm{QS}(\omega)^2\right)}$$

$$\mathrm{PHASE}(\omega) = \arctan\left(-\left(\mathrm{QS}(\omega)/\mathrm{CS}(\omega)\right)\right)$$

Cautions, restrictions

Maximum number of time lags and maximum number of frequency intervals are 500.
Data files with zero sample interval will not be accepted by the command.
It is also impossible to compute the cross spectrum between two columns from data files with different sample intervals.

Hints

See command FROP for a possible use of φ_{yu}.

SPTRF

Purpose

To compute the power spetrum or the amplitude and phase of a transfer function. The transfer function may be given in discrete time or continuous time form.

Command

```
SPTRF [('POW'/'AMP' 'DISCR'/'CONT')] FRF[(F)]←SYST[(NAME)]
   TPN NRN TPD NRD [FREQ]
```

Arguments

'POW'/'AMP'	switch choosing a power spectrum or an amplitude and phase computation. Default is 'AMP'.
'DISCR'/'CONT'	switch specifying DISCRete time or CONTinuous time. Default is 'DISCR'.
FRF	output frequency response file
F	response number (default 1)
SYST	name of system file
NAME	section name in system file
TPN	numerator polynomial type (A,B,C or D)
NRN	numerator polynomial number
TPD	denominator polynomial type (A,B,C or D)
NRD	denominator polynomial number
FREQ	file with frequency values

Function

The power spectrum or the amplitude and phase is computed for the transfer function

$$\frac{TPN_{NRN}}{TPD_{NRD}}$$

for NOF. frequencies between WMIN. and Min(WMAX., π/T) where T is the sample interval for a discrete time system. For continuous time systems, the upper limit is WMAX. . WMIN. and WMAX. are global variables.

If FREQ is specified, the frequency points are read from column 1 of this file, but they are still checked against the limits above.

Method

The power spectrum is computed:

$$\varphi(\omega) = / H(e^{i\omega T}) /^{2} \quad \text{or}$$

$$\varphi(\omega) = / G(i\omega)/^{2}$$

in the discrete and continuous time cases respectively.

Hint

The frequency response can be plotted using the command BODE.

Example

```
>SPTRF FRF←SYST  B 1   A 1
>BODE FRF
```

FILT

Purpose

To compute digital low-, band- and high-pass filters of given order with given cut-off frequencies.

Command

```
FILT SYST←TYPE NO T OML [OMH]
```

Arguments

SYST	system file name
TYPE	LP, BP or HP for low-, band- or high-pass filter
NO	filter order
T	sample interval(s)
OML	cut off frequency (rad/s)
OMH	high cut off frequency (for BP filter only) (rad/s)

Function

Parameters for a filter of type TYPE and order NO with cut off frequency(ies) OML (and OMH) are computed and written into the system file SYST as A- and B-polynomials.

Method

A bilinear z-transform method is used which eliminates aliasing effects. The Laplacian variable in the continuous filter transfer function H(s) is substituted by $2(1-z^{-1})/T(1+z^{-1})$ giving the discrete transfer function

H(z). The cut-off frequencies ω_i are substituted by the 'pseudo frequencies' $\nu_i = (2/T)\tan(\omega_i T/2)$. Band-pass filters and high- or low-pass filters of higher order than 1 are computed by multiplication of 1st order filters.

Reference

Roggenbauer, Seifertz, Olsson: Identification and Adjoint Problems of Process Computer Control, Re-7222, Lund Inst of Techn, Dept of Automatic Control, Lund

Cautions, restrictions

Maximum filter order is 10. Cut-off frequencies higher than π/T rad/s will not be accepted by the command.

Hints

A diagram of the amplitude of the filter spectrum may be obtained via the SPTRF and BODE commands.
The filtering is performed with the command DSIM.
Filtering of the type $y_f(t) = y(t) + a \cdot y(t-n)$ can be performed using the SLIDE, SCLOP and VECOP commands.

Example

```
>FILT HPFIL←HP  1  0.5  0.1
>DSIM YF←HPFIL DATA (4)
```

DSIM

Purpose

To simulate multiple input - single output discrete systems of the form

$$y(t) = \Sigma \frac{B_i(q^{-1})}{A_i(q^{-1})} u_i(t) + \Sigma \lambda_i \frac{C_i(q^{-1})}{D_i(q^{-1})} e_i(t)$$

Command

```
DSIM Y[(C1)]←SYST[(NAME)] U1[(C11 C12 ..)][U2[(C21 ..)] ..][NP]
```

Arguments

Y   output data file name
C1   column number in output file (default 1)
SYST   system file name
NAME   section name within SYST
U1,2   input data file names
Ci1,i2,..column numbers in input file Ui (default 1,2,..)
NP   number of points to be simulated (default the number of samples in the shortest input file)

Function

The discrete system SYST is simulated using U1(C11 ..), U2(C21 ..),.. as inputs and noise. Inputs are assumed to appear first and noise last. The output is written into Y(C1).

Method

The simulation is performed as a superposition of simulations of single input - single output systems with the initial values for the output zero and the transient from the initial values if such are present in SYST[(NAME)].
If SYST[(NAME)] contains only one A-polynomial it is used for all inputs.
If no D-polynomial(s) is present, the A-polynomial(s) is used instead.

Cautions, restrictions

The maximum number of points that can be simulated is 3000. Initial values for the output can not be used if there is more than one A-polynomial.

DETER

Purpose

To simulate the deterministic part of a discrete multi-input single output system, i e

$$y(t) = \Sigma \frac{B_i(q^{-1})}{A_i(q^{-1})} u_i(t)$$

Command

```
DETER Y[(C1)]←SYST[(NAME)] U1[(C11 C12 ..)][U2[(C21 ..)]..][NP]
```

Arguments

Y	output data file name
C1	column number in output file (default 1)
SYST	system file name
NAME	section name within SYST
U1,2,..	input data file names
Ci1,i2,	column numbers in input file Ui (default 1,2,..)
NP	the number of points to be simulated (default the number of samples in the shortest input file)

Function

The deterministic part of the system SYST is simulated using U1(C11 ..), U2(C21 ..),.. as inputs. The output is written into Y(C21).

Method

The simulation is performed as a superposition of simulations of single input - single output system with the initial values for the output zero and the transient from the initial values if such are present in SYST[(NAME)]. If SYST[(NAME)] contains only one A-polynomial it is used for all inputs.

Cautions, restrictions

The maximum number of points that can be simulated is 3000. Initial values for the output can not be used if there is more than one A-polynomial.

Hint

The deterministic model error may be obtained if the deterministic output is subtracted from the measured output using the VECOP command.

RESID

Purpose

To test the whiteness of the residuals and independence between the residuals and the inputs of an estimated model.

Command

```
RESID RES[(C11)]←SYST[(NAME)] DATA[(C21 ..)] NOL
```

Arguments

RES	residual file name
SYST	system file name
NAME	section name within SYST
DATA	data file name
C11,C21,..	column numbers (default 1,1,2,...)
NOL	maximum lag when computing autocorrelations of the residuals and cross correlations between residuals and inputs

Function

The residuals are computed using DATA(C21 ..) as inputs and output to SYST. Note that the output must be the last column specified or, if no columns are specified, the last column in DATA.

The normality and whiteness of the residuals are tested, the autocorrelations for the residuals and the cross correlations between the residuals and the input(s) (if any) are computed for up to NOL lags. Then the independence between the residuals and the input(s) is tested.

The autocorrelations and cross correlations are plotted on display along with information about test quantities and number of degrees of freedom. The information is divided into pages. A key on the keyboard (preferably rubout) must be depressed when a new page is wanted. The information displayed as well as more detailed information from the test of normality of the residuals is printed on line printer if the global variable PRINT. is nonzero. See the global variable list.

Method

The residuals are computed using the formula

$$\varepsilon(t) = \frac{A(q^{-1})}{C(q^{-1})} y(t) - \Sigma \frac{B_i(q^{-1})}{C(q^{-1})} u_i(t)$$

The normality is tested by a chi-square goodness-of-fit test. The observations are grouped into K intervals forming a frequency histogram. The observed frequency in the ith class interval is called f_i and the expected one F_i if the true distribution was normal. The quantity

$$x^2 = \sum_i^K \frac{(f_i - F_i)^2}{F_i}$$

is approximately $\chi^2(K-3)$. The number of class intervals is chosen depending on the number of observations.

The number of changes of sign of the residuals is computed and is

$$N\left(\frac{M-1}{2}, \frac{\sqrt{M-1}}{2}\right)$$

where M is the number of observations. To test the independence of the residuals

$$\sum_1^5 r_\varepsilon^2(\tau)$$

is computed. This quantity is $\chi^2(5)$.

To test the independence between the residuals and the input(s) $x^T P_x^{-1} x$ is computed where

$$x = \begin{vmatrix} r_{\varepsilon u}(j) \\ \vdots \\ r_{\varepsilon u}(j+m-1) \end{vmatrix}$$

and

$$P_x = \frac{1}{N} \begin{vmatrix} r_{uu}(0) & \dots\dots\dots\dots & & r_{uu}(m-1) \\ r_{uu}(1) & r_{uu}(0) & \dots\dots & r_{uu}(m-2) \\ & & & \\ r_{uu}(m-1) & \dots\dots\dots\dots & & r_{uu}(0) \end{vmatrix}$$

This quantity is asymptotically $\chi^2(m)$.
The cross correlation function is defined by

$$r_{\varepsilon u}(\tau) = E[\varepsilon(t) \cdot u(t+\tau)]$$

For positive τ, j is chosen equal n+1 where n is the order of the system description in SYST[(NAME)]. m is chosen to 5. The test quantity is also computed for negative $\tau = -4, \dots, 0$.
For both the autocorrelation and cross correlation functions the two sigma limits $\left(= 1.96(1/\sqrt{M})\right)$ are plotted, indicating the region inside which the estimates of the correlations should be with 95 % probability if the residuals (and the input(s)) are independent.

Reference

J S Bendat and A G Piersol: Measurement and analysis of random data, Wiley, New York, 1966.

Cautions, restrictions

The length of the input file must not exceed 3000 samples.

Hints

The test quantity for negative τ can indicate whether the original system is under feedback or not. A large test quantity may indicate feedback.
If the system file describes a unit system, i e contains only one A-parameter and one C-parameter (both equal to 1.0), the independence and normality of a time series may be tested.

Some (approximate) values for the CHI-SQUARE statistic follows: (Definition $P(\chi^2 > \ell) = 0.05$.)

Degrees of freedom	ℓ
5	11
10	18
15	25
20	31
25	38
30	44

RANPA

Purpose

To generate a gaussian random vector with given covariance matrix and add it to the parameters in a system description.

Command

```
RANPA DSYST←OSYST[(NAME)]
```

Arguments

DSYST file name for disturbed parameter system
OSYST file name for original system description
NAME section name

Function

A gaussian random vector with a covariance matrix given in OSYST is generated and added to the parameters in OSYST. The new parameters are written into DSYST. Initial values for the output of the system will not be changed.

Caution

The random number generator is controlled by the global variable NU., which is updated after each call to RANPA. You can save or initialize NU. via LET.

Hint

RANPA can be used to determine the effect of uncertainties in an identification result. E g step or impulse response for different sets of parameters give an indication of which of the DC gain and the high frequency properties of the model is the most accurate.

Cautions, restrictions

At present the maximum number of parameters is 16.

ML

Purpose

To perform maximum likelihood identification on multiple input - single output systems on the form

$$A(q^{-1})y(t) = B_1(q^{-1})u_1(t) + \dots + B_m(q^{-1})u_m(t) + \lambda C(q^{-1})e(t)$$

Command

```
ML[(SW)] SYST[(NAME)]←DATA (C1 ..) NO
```

Arguments

SW	switch telling if subcommands are wanted. If so, SW = 'SC', otherwise SW = 'VOID' or missing.
SYST	system file name
NAME	name of section in SYST
DATA	data file name
C1,..	column numbers in data file (default 1,2,..)
NO	model order

Function

A maximum likelihood model of order NO is estimated from DATA(C1 ..) and written into the system file SYST. Note that the output of the system must be the last column specified or, if no column numbers are specified, the last column in DATA.

Subcommands

INVAL ABC/C SYST[(NAME)]

Fetches starting values for parameters from the system file SYST, section NAME.
ABC indicates that all parameters shall have starting values.
C indicates that only C-parameters shall have starting values, meaning that the first iteration will be a least squares estimation of the A- and B-parameters.
If INVAL is not used all parameters will have the starting value 0.0.

FIX A(1) [VA1] (2) [VA2] B(22) [VB22]..

The indicated parameters are fixed to the values VA1, .. (default 0.0 or the values given by INVAL).

SAVE [COMAT] [STDEV] [GRAD] [EVALS]

Indicates that the covariance matrix of the parameter estimates, the standard deviations of the parameter estimates, the gradient of the loss function and/or the eigenvalues of the second derivative matrix of the loss function shall be saved in the parameter file.

EXIT

No identification is performed but control is returned to the main program.

The esimation is started when a line is terminated by an alt mode.

Global variables used (default values underlined):

INIML. 1 initial values for the output will be estimated.
       0 no estimation
PRIML. 0 no printout

1 loss function and lambda for starting values are printed as well as the final estimate with derivatives, second derivative matrix and inverse of the second derivative matrix. The final estimate is also displayed.

2 1 + the estimate printed for each iteration

3 2 + derivatives, second derivative matrix and inverse of the second derivative matrix printed for each iteration. Each estimate is displayed.

LIML. 1 the residuals will be limited to 3·lambda in each iteration, i e if $\varepsilon(t)>3\lambda$, then $\varepsilon(t) = 3\lambda$.

<u>0</u> no limitation

ITML. maximum number of iterations (default 20).

With the SC option these variables are displayed. They may any time be modified with a LET command.

Method

A maximum likelihood estimate of the parameters in the model

$$A(q^{-1})y(t) = B_1(q^{-1})u_1(t) +\ldots+ B_m(q^{-1})u_m(t) + \lambda C(q^{-1})e(t)$$

is obtained by minimizing the loss function

$$V(\theta) = \frac{1}{2}\Sigma\varepsilon^2(t)$$

where

$$C(q^{-1})\varepsilon(t) = A(q^{-1})y(t) - B_1(q^{-1})u_1(t) - \ldots - B_m(q^{-1})u_m(t)$$

and

$$\theta = (a_1,\ldots,a_n,b_{11},\ldots,b_{1n},\ldots,b_{m1},\ldots,b_{mn},c_1,\ldots,c_n)$$

The maximum likelihood estimate of λ will be

$$\hat{\lambda}^2 = \frac{2}{N}V(\hat{\theta})$$

where $\hat{\theta}$ is the minimum point of V.

The minimization is performed iteratively by a combined Gauss-Newton and Newton-Raphson algorithm.

The parameter accuracy is estimated by

$$\sigma^2_{\theta_i} = \lambda^2[V_{\theta\theta}(\hat{\theta})]^{-1}_{ii}$$

Akaike's test quantity (n- is the number of parameters):

$$AIC = N*(\ln 2\pi * 2 \ln \hat{\lambda}) + 2\ np$$

is expected to have a minimum when the number of parameters is correct.

The convergence criteria are

$$\max_i \left|\frac{\Delta\theta_i}{\theta_i}\right| \leq 10^{-4} \quad \text{or} \quad |\Delta V| \leq 10^{-6}$$

The first step of the iteration is a least squares estimate of the parameters of $A(q^{-1})$ and $B_i(q^{-1})$ provided that the subcommand INVAL ABC is not used.

References

Åström, K J, Bohlin, T, and Wensmark, S: Automatic Construction of Linear Stochastic Dynamic Models for Stationary Processes with Random Disturbances using Operating Records, Report TP 18.150 (1965), IBM Nordic Laboratory, Sweden.

Gustavsson, I: Parametric Identification of Multiple Input, Single Output Linear Dynamic Systems, Report 6907 (1969), Dept of Automatic Control, Lund Inst of Techn, Lund, Sweden.

Almqvist, R: Program för maximum-likelihood identifiering på PDP-15 (Program for Maximum Likelihood Identification on PDP-15), Report RE-103(1972), Master Thesis, Dept of Automatic Control, Lund Inst of Techn, Lund, Sweden.

Cautions, restrictions

Maximum order is 9, maximum number of input is 8 and maximum total number of estimates parameters is 25 including fixed parameters and initial values for the output. There is no maximum for the number of input data.

Hints

1. If just a least squares estimation is wanted, use no initial values for the A and B parameters and set PRIML, and ITML, to 1.

2. Time series analysis can be carried out if only one column is indicated since the number of inputs is always assumed to be one less than the number of columns specified and the output is always the last column specified.

3. A moving average model $y = Ce$ can only be estimated in the following way:
   Fix all a_i, $i = 1, \ldots, n$ to zero.
   Give any nonzero initial values to c_i, $i = 1, \ldots, n$.

4. Known relations between parameters can be introduced as indicated by the following examples:

   Ex 1. Known relation: $b_1u_1(t-1) + 1.5b_1u_2(t-1)$
   Develop a signal: $\tilde{u}(t-1) = u_1(t-1) + 1.5u_2(t-1)$
   and then estimate $b_1\tilde{u}(t-1)$

   Ex 2. Known relation: $a_1y(t-1) + 3a_1y(t-2)$
   Develop a signal: $\tilde{y}(t-1) = y(t-1) + 3y(t-2)$
   and use this signal as an input signal and estimate the coefficient a_1.

5. Notice that the same technique can be used for e g the estimation of the coefficients a_{12} and a_{13} in the model

$$y(t) + a_1y(t-1) + a_{12}y(t-12) + a_{13}y(t-13) = \\ = b_1u(t-1) + e(t) + c_1e(t-1)$$

In this case develop a signal $\tilde{u}(t-1) = y(t-12)$ and use this signal as an input signal to estimate a_{12} and a_{13}.

6. If the experiment/simulation is not started in steady state with the levels subtracted from the data it might be useful also to estimate initial values of the difference equation

   $$A(q^{-1})y(t) = B_1(q^{-1})u_1(t) +\ldots+ B_m(q^{-1})u_m(t) + \lambda C(q^{-1})e(t)$$

   i e the values y(0), y(-1), ..., y(1-n), where n is the order of the system.

7. The computations may be interrupted with the data switch # 3 (PDP-15).

8. During the minimization, it can happen that the algorithm converges to a local minimum instead of a global one. To overcome this problem, start the algorithm with different initial values (subcommand INVAL). The system file used by INVAL may be created in one of two ways:

   a) Via the editor, maybe by altering some coefficients in an existing system description.

   b) By fixing some coefficients to values different from the current (local) minimum, and then performing a new identification. The resultant system description is used as a new starting point, now with all parameters free.

STRUC

Purpose

To create a file (a structure file) containing information on the structure of a desired polynomial model. This information is then via the resulting file passed on to a model matching command (e g SQR and LS).

Command

```
STRUC STRF
or
STRUC [STRF1]←STRF2
```

Arguments

STRF, STRF1 and STRF2: Names of structure files.

Function

In the first form of the command, a new structure file is created. In the second form the file STRF2 is updated giving a new file STRF1 or a new version of STRF2 if STRF1 is missing.
STRUC assumes a model on polynomial form

$$A(q^{-1})y(t) = \sum_{i=1}^{NU} q^{-ki} B_i(q^{-1})u_i(t) + e(t).$$

The structural quantities NA NU NB(i) K(i) can be given both maximum and actual values.

STRUC will receive information through subcommands and will perform various checks for consistency, eg:

a) New maximum values are legal only for a new structure file or after the REVRT subcommand.

b) New actual values must be less or equal to maximum values.

c) The number of inputs (NU) must be specified prior to the commands NB or KB.

Subcommands

In some commands below a switch exists, here designated sw. Its value is 'MAX'/'ACT', default value being 'ACT' (cf subcommand KB). It function is to indicate that a maximum or actual value is beeing specified for the variable in question.

REVRT

This command will remove all information other than maximum and actual structure variables (cf command SQR).

NA sw N

NA, i e the degree of the A-polynomial is given the maximum/actual value N.

NU sw N

NU, i e the number of inputs is given maximum/actual value N.

NB sw N1 N2 ... NNU

NBi, i e the degrees of the B-polynomials are given maximum/actual values N1, N2, etc.

KB sw N1 N2 ... NNU

KBi, i e the delay in each input are given maximum/ /actual values N1, N2 etc. Here a third alternative, 'MIN' is allowed for sw. The default is 1, 0 is allowed, negative values are illegal. (In such a case, use command SLIDE.)

FIX A(N) [VN] ... B(M) [VM]...

The parameter $a_n \dots b_m$ etc are given fixed values v_n, v_m etc. If no value is given, zero is assumed.

UNFIX [A(N1 N2 ...)] [B(M1 M2 ...)]

The specified parameter(s) (default all) is unfixed.

SQR

Purpose

To compute the square-root matrix R of the least-squares identification algorithm. See the command LS !

Command

```
SQR RMAT←DATA [(C1 C2 ...)] STRF
```

Arguments

RMAT	the name of the R-matrix
DATA	the data file name
C1, ...	column numbers in data file
STRF	name of structure file

Function

The R-matrix is computed recursively from the data file. The output signal of the system is assumed to be the last column specified or, if no column numbers are given, the last column.

SQR will update the structure file with a notation of the name of the R-matrix and the number of data points used, as well as the name and column numbers of the data file. This serves two purposes:

a) This information is used by LS.

b) The existence of this information prevents STRUC to alter the maximum values, which are used by LS to properly interpret the R-matrix.

Hint

For a given set of maximum values, a single call to SQR is sufficient, still allowing repeated calls to STRUC, changing actual values, and to LS, computing parameter estimates.

LS

Purpose

To compute a least-squares model using the square-root algorithm. The model is a multiple input - single output model of the form:

$$A(q^{-1})y(t) = \sum_{i}^{NU} q^{-ki} B_i(q^{-1})\ u_i(t) + e(t)$$

with deg A = NA, deg B_i = NB(i) and K_i = KB(i). Cf the command STRUC !

Command

```
LS [(SW)] SYST[(NAME)]←STRF
```

Arguments

SW switch telling if subcommands are wanted. If so, SW = 'SC', otherwise SW = 'VOID' or missing.

SYST system file receiving the result

NAME optional name of section within SYST

STRF structure file prepared by commands STRUC and SQR

Function

The input structure file contains:

a) The name of the file containing the R-matrix computed by SQR. See also equation (3) below.

b) The maximum structure variables defining the meaning of the elements in the R-matrix.

c) Information in the form of actual polynomial degrees or indices of fixed parameters telling which parameters that are not to be estimated.

Substituting backwards in equation (4) below, it is then possible to compute the desired parameters, with the additional precaution described.

The estimated parameter values are output to the system file together with some extra information. Some is optional, controlled by subcommands, while other is always given, e g the value of the loss function and Akaike's test quantity. This is also displayed together with the percentage contribution to the loss function from the unestimated parameters. The displayed information is also printed depending on the reserved global variable PRINT..

Subcommands

SAVE STDEV	estimated uncertainties of parameters are output to the system file
SAVE COMAT	the covariance matrix of the parameter estimates is output to the system file

Method

For each measurement point (less some points at the beginning of the series) the quantity e(t) is observed:

$$y(t) + \sum_{i} a_i y(t-i) - \sum_{ij} b_{ij}\ u_i(t-j-k_i) = e(t)$$

Using normal symbols, the complete set of observations can be expressed in the equation (see the reference below)

$$Y - \phi\Theta = E \qquad (1)$$

where Θ contains the unknown parameters while Y and ϕ contains the observations. The parameters Θ are now to be computed so as to minimize the loss function $V = \frac{1}{2} E^T E$. If Q is an orthogonal matrix, i e $Q^T = Q^{-1}$, we have

$$E'^T E' = (QE)^T \, QE = E^T Q^{-1} QE = E^T E$$

Now, rewriting (1) using partitioned matrices and multiplying by Q from the left we have:

$$Q[\phi \;\; Y]\begin{bmatrix} -\Theta \\ 1 \end{bmatrix} = QE = E' \tag{2}$$

Q is then chosen so that

$$Q[\phi \;\; Y] = \begin{bmatrix} R \\ O \end{bmatrix} \tag{3}$$

The upper-right triangular matrix R of order n+1, n being the number of parameters, is computed using Householder transformations. The steps between 1 and 3 are performed by the command SQR.

Using (3) equation (2) can now be rewritten omitting the zeroes (i e rows n+2, n+3,...)

$$\left(\begin{array}{c|c} R_n & \begin{array}{c} r_{1\; n+1} \\ r_{2\; n+1} \\ \vdots \\ r_{n\; n+1} \\ r_{n+1\; n+1} \end{array} \end{array}\right) \begin{pmatrix} -\Theta_1 \\ -\Theta_2 \\ \vdots \\ -\Theta_n \\ 1 \end{pmatrix} = \begin{pmatrix} e'_1 \\ e'_2 \\ \vdots \\ e'_n \\ e'_{n+1} \end{pmatrix} \tag{4}$$

Now, by direct backwards substitution Θ_i can be computed yielding $e'_1 = e'_2 = \dots = e'_n = 0$. Thus the minimal value of the loss function is given by

$$V_n = \frac{1}{2} E^T E = \frac{1}{2} E'^T E' = \frac{1}{2} {e'_{n+1}}^2 = \frac{1}{2} r_{n+1\; n+1}^{\;2} \; .$$

The heavy part of the computations, viz the triangularization (3) is done in the command SQR allowing for the maximum number of parameters. If, however, actual values in the structure file indicates that some parameters are to be omitted, the corresponding columns of R_n is omitted followed by a new triangularization giving a new R-matrix, R_m, $m < n$. For each omitted parameter, the corresponding loss function $V_m = \frac{1}{2} r_{m+1\ m+1}^2$ and the value of AIC is computed and displayed.

Then, for the required model, the noise intensity, the covariance matrix and the parameter accuracies are computed as well as Akaike's test quantity (AIC). Cf the command ML.

$$\hat{\lambda}^2 = \frac{2}{N} V_m$$

$$\sigma^2_{\theta_i} = \hat{\lambda}^2 \, [R_m^T \cdot R_m]^{-1}_{ii}$$

$$AIC = N(\ln 2\pi + 2 \ln \hat{\lambda}) + 2n$$

Hints

a) Note that the time-consuming command SQR has to be performed only once, for a given set of max-values.

b) Note the easy way of estimating values for the KB_i.

c) Compare example 1 in the chapter on MACROs.

Reference

Åström, K J: Lecture notes on identification.

DELET

Purpose

To delet data, system and macro files from disk.

Command

```
DELET FILE1 [FILE2 ..]
```

Function

The files FILE1 .. are deleted from disk.

TURN

Purpose

To manipulate the switches for macro command, line printer and display output and plotting versus time/ sample number.

Command

TURN SWITCH STATE

Arguments

SWITCH	'MACOM', 'LPCOM', 'DIS' or 'TIME'
STATE	'ON' or 'OFF' for MACOM, LPCOM and TV 'OFF', 'S', 'M', or 'H' for TIME

Function

'MACOM' enables/disables echoing of commands from an executing macro.

'LPCOM' enables/disables echoing of correct commands on line printer.

'DIS' enables/disables all output on display.

'TIME' determines whether data shall be plotted versus sample number (OFF) or time in seconds (S), minutes (M) or hours (H).

Default values as IDPAC is started are MACOM and DIS ON, LPCOM and TIME OFF.